Factoring nonnegative matrices with linear programs

Victor Bittorf[*], Benjamin Recht[*], Christopher Ré[*] and Joel A. Tropp[†]
[*] Computer Sciences Department, University of Wisconsin-Madison
[†] Computational and Mathematical Sciences, California Institute of Technology

June 2012

Abstract

This paper describes a new approach for computing nonnegative matrix factorizations (NMFs) with linear programming. The key idea is a data-driven model for the factorization, in which the most salient features in the data are used to express the remaining features. More precisely, given a data matrix $\boldsymbol{X}$, the algorithm identifies a matrix $\boldsymbol{C}$ that satisfies $\boldsymbol{X} \approx \boldsymbol{C}\boldsymbol{X}$ and some linear constraints. The matrix $\boldsymbol{C}$ selects features, which are then used to compute a low-rank NMF of $\boldsymbol{X}$. A theoretical analysis demonstrates that this approach has the same type of guarantees as the recent NMF algorithm of Arora et al. (2012). In contrast with this earlier work, the proposed method (1) has better noise tolerance, (2) extends to more general noise models, and (3) leads to efficient, scalable algorithms. Experiments with synthetic and real datasets provide evidence that the new approach is also superior in practice. An optimized C++ implementation of the new algorithm can factor a multi-Gigabyte matrix in a matter of minutes.

Keywords. Nonnegative Matrix Factorization, Linear Programming, Stochastic gradient descent, Machine learning, Parallel computing, Multicore.

1 Introduction

Nonnegative matrix factorization (NMF) is a popular approach for selecting features in data [13–15, 20]. Many machine learning and data-mining software packages (including Matlab [3], R [10], and Oracle Data Mining [1]) now include heuristic computational methods for NMF. Nevertheless, we still have limited theoretical understanding of when these heuristics are correct.

The difficulty in developing rigorous methods for NMF stems from the fact that the problem is computationally challenging. Indeed, Vavasis has shown that NMF is NP-Hard [24]; see [4] for further worst-case hardness results. As a consequence, we must instate additional assumptions on the data if we hope to compute nonnegative matrix factorizations in practice.

In this spirit, Arora, Ge, Kannan, and Moitra (AGKM) have exhibited a polynomial-time algorithm for NMF that is provably correct—provided that the data is drawn from an appropriate model [7]. Their result describes one circumstance where we can be sure that NMF algorithms are capable of producing meaningful answers. This work has the potential to make an impact in machine learning because proper feature selection is an important preprocessing step for many other techniques. Even so, the actual impact is diminished by the fact that the AGKM algorithm is too computationally expensive for large-scale problems and is not tolerant to departures from the modeling assumptions. Thus, for NMF, there remains a gap between the theoretical exercise and the actual practice of machine learning.

The present work presents a scalable, robust algorithm that can successfully solve the NMF problem under appropriate hypotheses. Our first contribution is a new formulation of the nonnegative feature selection problem that requires only the solution of a single linear program. Second, we provide a theoretical analysis of this algorithm based on the theory of linear programming. This argument shows that our method succeeds under the same modeling assumptions as the AGKM algorithm, but our approach is substantially more tolerant to noise. We additionally prove that if there exists a unique, well-defined model, then we can recover this model with much higher noise than the AGKM results suggest. One could argue that NMF only "makes sense" (i.e., is well-posed) when such a unique solution exists, and so we believe this result is of independent interest. Furthermore, our algorithm and analysis extend in a transparent way to cover a wider class of noise models.

In addition to these theoretical contributions, our work also includes a major algorithmic and experimental component. Our formulation of NMF allows us to exploit methods from operations research and database systems to design solvers that scale to extremely large datasets. We develop an efficient stochastic gradient descent (SGD) algorithm that is (at least) two orders of magnitude faster than the approach of AGKM when both are implemented in Matlab. We describe a parallel implementation of our SGD algorithm that can robustly factor matrices with 10^5 features and 10^6 examples in a few minutes on a multicore workstation.

Our formulation of NMF uses a data-driven modeling approach to simplify the factorization problem. More precisely, we search for a small collection of rows from the data matrix that can be used to express the other rows. This type of approach appears in a number of other factorization problems, including rank-revealing QR [12], interpolative decomposition [17], subspace clustering [8, 21], dictionary learning [9], and others. Our computational techniques can be adapted to address large-scale instances of these problems as well.

2 Separable Non-negative Matrix Factorizations and Hott Topics

Notation. For a matrix $\boldsymbol{X}$ and indices i and j, we write $\boldsymbol{X}_{i\cdot}$ for the ith row of $\boldsymbol{X}$ and $\boldsymbol{X}_{\cdot j}$ for the jth column of $\boldsymbol{X}$.

Let $\boldsymbol{X}$ be a nonnegative $f \times n$ data matrix with columns indexing examples and rows indexing features. The goal of NMF is to to find a factorization $\boldsymbol{X} = \boldsymbol{F}\boldsymbol{W}$ where $\boldsymbol{F}$ is $f \times r$, $\boldsymbol{W}$ is $r \times n$, and both factors are nonnegative. $\boldsymbol{F}$ is a *feature matrix* and $\boldsymbol{W}$ is a *weight matrix.*

Unless stated otherwise, we assume that each row of $\boldsymbol{X}$ is normalized so it sums to one. Under this assumption, we may also assume that each row of $\boldsymbol{F}$ and of $\boldsymbol{W}$ also sums to one [4]. To verify this point, express $\boldsymbol{X} = (\boldsymbol{F}\boldsymbol{D})(\boldsymbol{D}^{-1}\boldsymbol{W})$ where $\boldsymbol{D}$ is a diagonal, nonnegative $r \times r$ matrix. We may choose $\boldsymbol{D}$ such that the rows of $\boldsymbol{W}$ all sum to one. Under this choice, we have

$$1 = \sum_{j=1}^{n} X_{ij} = \sum_{j=1}^{n} \sum_{k=1}^{r} F_{ik} W_{kj} = \sum_{k=1}^{r} F_{ik} \quad \text{for each index } i.$$

It is notoriously difficult to solve the NMF problem. Vavasis showed that it is NP-complete to decide whether a matrix admits a rank-r nonnegative factorization [24]. AGKM proved that an exact NMF algorithm can be used to solve $\mathsf{3\text{-}SAT}$ in subexponential time [4].

The literature contains some mathematical analysis of NMF that can be used to motivate algorithmic development. Thomas [22] developed a necessary and sufficient condition for the existence

Algorithm 1: AGKM: Approximably Separable Non-negative Matrix Factorization [4]

```
Initialize $R = \emptyset$.
Compute the $f \times f$ matrix $\boldsymbol{D}$ with $D_{ij} = \|\boldsymbol{X}_{i\cdot} - \boldsymbol{X}_{j\cdot}\|_1$.
for $k = 1, \dots f$ do
    Find the set $\mathcal{N}_k$ of rows that are at least $5\epsilon/\alpha + 2\epsilon$ away from $\boldsymbol{X}_{k\cdot}$.
    Compute the distance $\delta_k$ of $\boldsymbol{X}_{k\cdot}$ from $\operatorname{conv}(\{\boldsymbol{X}_{j\cdot} : j \in \mathcal{N}_k\})$.
    if $\delta_k > 2\epsilon$, add $k$ to the set $R$.
end for
Cluster the rows in $R$ as follows: $j$ and $k$ are in the same cluster if $D_{jk} \leq 10\epsilon/\alpha + 6\epsilon$.
Choose one element from each cluster to yield $\boldsymbol{W}$.
$\boldsymbol{F} = \arg\min_{\boldsymbol{Z} \in \mathbb{R}^{f \times r}} \|\boldsymbol{X} - \boldsymbol{Z}\boldsymbol{W}\|_{\infty,1}$
```

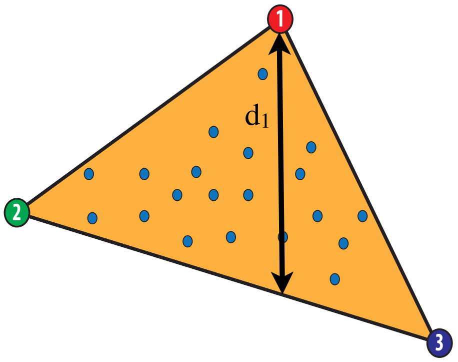

Figure 1: Numbered circles are hott topics. Their convex hull (orange) contains the other topics (small circles), so the data admits a separable NMF. The arrow d_1 marks the ℓ_1 distance from hott topic (1) to the convex hull of the other hott topics; d_2 and d_3 are similar. The hott topics are α-robustly simplicial when each $d_i \geq \alpha$.

of a rank-r NMF. More recently, Donoho and Stodden [7] obtained a related sufficient condition for uniqueness. AGKM exhibited an algorithm that can produce a nonnegative matrix factorization under a weaker sufficient condition. To state their results, we need the following

Definition 2.1 *A set of vectors* $\{\boldsymbol{x}_1, \dots, \boldsymbol{x}_r\} \subset \mathbb{R}^d$ *is* simplicial *if no vector* $\boldsymbol{x}_i$ *lies in the convex hull of* $\{\boldsymbol{x}_j : j \neq i\}$*. The set of vectors is* α-robust simplicial *if, for each* i*, the* ℓ_1 *distance from* $\boldsymbol{x}_i$ *to the convex hull of* $\{\boldsymbol{x}_j : j \neq i\}$ *is at least* α*. Figure 1 illustrates these concepts.*

These ideas support the uniqueness results of Donoho and Stodden and the AGKM algorithm. Indeed, we can find an NMF of $\boldsymbol{X}$ efficiently if $\boldsymbol{X}$ contains a set of r rows that is simplicial and whose convex hull contains the remaining rows.

Definition 2.2 *A NMF* $\boldsymbol{X} = \boldsymbol{F}\boldsymbol{W}$ *is called* separable *if the rows of* $\boldsymbol{W}$ *are simplicial and there is a permutation matrix* Π *such that*

$$\Pi \boldsymbol{F} = \begin{bmatrix} \boldsymbol{I}_r \\ \boldsymbol{M} \end{bmatrix} . \tag{1}$$

To compute a separable factorization of $\boldsymbol{X}$, we must first identify a simplicial set of rows from $\boldsymbol{X}$. Then we can compute weights that express the remaining rows as convex combinations of this distinguished set. We call the simplicial rows *hott* and the corresponding features the *hott topics*.

Observe that this model allows us to express all the features for a particular instance if we know the values of the instance at the simplicial rows. This assumption can be justified in a variety of applications. For example, in text, knowledge of a few keywords may be sufficient to reconstruct counts of the other words in a document. In vision, localized features can be used to predict gestures. In audio data, a few bins of the spectrogram may allow us to reconstruct the remaining bins.

While the non-negative matrices one may encounter in practice might not admit a separable factorization, they may be *well-approximated* by one. AGKM derived an algorithm for non-negative matrix factorization when the matrix was either separable or well-approximated by a separable factorization. To state their result, we introduce a $(\infty, 1)$-norm on $f \times n$ matrices:

$$\|\boldsymbol{\Delta}\|_{\infty,1} := \max_{1 \le i \le f} \sum_{j=1}^{n} |\Delta_{ij}| \, .$$

Theorem 2.3 ([4]) *Suppose ϵ and α are nonnegative constants satisfying $\epsilon \le \frac{\alpha^2}{20+13\alpha}$, and $\hat{\boldsymbol{X}} = \boldsymbol{X} + \boldsymbol{\Delta}$ where $\boldsymbol{X}$ admits a separable factorization $\boldsymbol{F}\boldsymbol{W}$, $\boldsymbol{W}$ are α-robustly simplicial, and $\|\boldsymbol{\Delta}\|_{\infty,1} \le \epsilon$. Then Algorithm 1 finds a rank-r factorization $\hat{\boldsymbol{F}}\hat{\boldsymbol{W}}$ such that $\left\|\hat{\boldsymbol{X}} - \tilde{\boldsymbol{F}}\tilde{\boldsymbol{W}}\right\|_{\infty,1} \le 10\epsilon/\alpha + 7\epsilon$.*

Note that in particular, this algorithm computes the factorization exactly when $\epsilon = 0$. Although this method is guaranteed to run in polynomial time, it has a number of undesirable features. First, the algorithm requires a priori knowledge of the parameters α and ϵ. It may be possible to calculate ϵ, but we can only estimate α if we know which rows are hott. Second, the algorithm computes all ℓ_1 distances between rows at a cost of $O(f^2 n)$. Third, for every row in the matrix, we must determine its distance to the convex hull of the rows that lie at a sufficient distance; this step requires us to solve a linear program for each row of the matrix at a cost of $\Omega(fn)$. Finally, this method is intimately linked to the choice of the error norm $\|\cdot\|_{\infty,1}$. It is not obvious how to adapt the algorithm for other noise models. We present a new approach, based on linear programming, that overcomes all these deficiencies.

3 Main Theoretical Results: NMF by Linear Programming

The main theoretical result of this paper shows that matrices that admit or are well approximated by separable factorizations can be factored by solving linear programs. A major advantage of this linear programming formulation is that it will enable scaling to very large data sets.

Here is the key observation. If we pad $\boldsymbol{F}$ with zeros to be an $f \times f$ matrix, we have

$$\boldsymbol{X} = \boldsymbol{\Pi}^T \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{M} & \boldsymbol{0} \end{bmatrix} \boldsymbol{\Pi} \boldsymbol{X} =: \boldsymbol{C}\boldsymbol{X}$$

We call the matrix $\boldsymbol{C}$ *factorization localizing.* Note that any factorization localizing matrix $\boldsymbol{C}$ is an element of the polyhedral set

$$\Phi(\boldsymbol{X}) := \{\boldsymbol{C} \ge 0 \ : \ \boldsymbol{C}\boldsymbol{X} = \boldsymbol{X}, \mathrm{Tr}(\boldsymbol{C}) = r, \mathrm{diag}(\boldsymbol{C}) \le 1, C_{ij} \le C_{jj} \text{ for all } i, j\} \, .$$

Thus, to find a factorization of $\boldsymbol{X}$, it suffices to find a feasible element of $\boldsymbol{C} \in \Phi(\boldsymbol{X})$ whose diagonal is integral. This can be accomplished by linear programming. With such a $\boldsymbol{C}$, we can define $\boldsymbol{W}$ to be the rows of $\boldsymbol{X}$ corresponding to the indices i where $C_{ii} = 1$. Then $\boldsymbol{F}$ consists of the nonzero columns of $\boldsymbol{C}$. This approach is summarized in Algorithm 2. In turn, we can prove the following

Proposition 3.1 *Suppose $\boldsymbol{X}$ admits a separable factorization $\boldsymbol{F}\boldsymbol{W}$. Then Algorithm 2 constructs a nonnegative matrix factorization of $\boldsymbol{X}$.*

As the proposition suggests,we can isolate the rows of $\boldsymbol{X}$ that yield a simplicial factorization by solving a single linear program. The factor $\boldsymbol{F}$ can be found by selecting columns of $\boldsymbol{C}$.

Algorithm 2 Separable Non-negative Matrix Factorization by Linear Programming

```
Require: An $f \times n$ matrix $\boldsymbol{X}$ that has an $\alpha$-simplicial factorization.
Ensure: An $f \times r$ matrix $\boldsymbol{F}$ and $r \times n$ matrix $\boldsymbol{W}$ with $\boldsymbol{F} \geq \boldsymbol{0}$, $\boldsymbol{W} \geq \boldsymbol{0}$, and $\boldsymbol{X} = \boldsymbol{F}\boldsymbol{W}$.
Compute the $\boldsymbol{C} \in \Phi(X)$ that minimizes $\boldsymbol{p}^T \operatorname{diag} \boldsymbol{C}$ where $\boldsymbol{p}$ is any vector with distinct values.
Let $I = \{i \ : \ C_{ii} = 1\}$ and set $\boldsymbol{W} = \boldsymbol{X}_{I\cdot}$ and $\boldsymbol{F} = \boldsymbol{C}_{\cdot I}$.
```

3.1 Robustness to Noise

For the robust case, define the polyhedral set

$$\Phi_\tau(\boldsymbol{X}) := \left\{ \boldsymbol{C} \geq 0 \ : \ \|\boldsymbol{C}\boldsymbol{X} - \boldsymbol{X}\|_{\infty,1} \leq \tau, \operatorname{Tr}(\boldsymbol{C}) = r, \operatorname{diag}(\boldsymbol{C}) \leq 1, C_{ij} \leq C_{jj} \text{ for all } i,j \right\}$$

The set $\Phi(\boldsymbol{X})$ consists of matrices $\boldsymbol{C}$ that *approximately* locate a factorization of $\boldsymbol{X}$. We can prove the following

Proposition 3.2 *Suppose that ϵ and α are positive constants satisfying $\epsilon \leq \frac{\alpha^2}{8+4\alpha}$. Suppose $\hat{\boldsymbol{X}} = \boldsymbol{X} + \boldsymbol{\Delta}$ where $\boldsymbol{X}$ admits a separable factorization $\boldsymbol{F}\boldsymbol{W}$, $\boldsymbol{W}$ are α-robustly simplicial, and $\|\boldsymbol{\Delta}\|_{\infty,1} \leq \epsilon$. Then Algorithm 3 constructs a rank-r nonnegative factorization $\hat{\boldsymbol{F}}\hat{\boldsymbol{W}}$ such that $\left\|\hat{\boldsymbol{X}} - \tilde{\boldsymbol{F}}\tilde{\boldsymbol{W}}\right\|_{\infty,1} \leq 4\epsilon$.*

Algorithm 3 requires the solution of two linear programs. The first minimizes a cost vector over $\Phi_{2\epsilon}(\hat{\boldsymbol{X}})$. This lets us find $\hat{\boldsymbol{W}}$. Afterward, the matrix $\hat{\boldsymbol{F}}$ can be found by setting

$$\hat{\boldsymbol{F}} = \arg\min_{\boldsymbol{Z} \geq 0} \ \left\|\hat{\boldsymbol{X}} - \boldsymbol{Z}\hat{\boldsymbol{W}}\right\|_{\infty,1} . \tag{2}$$

Algorithm 3 Approximably Separable Nonnegative Matrix Factorization by Linear Programming

```
Require: An $f \times n$ matrix $\boldsymbol{X}$ that is within $\epsilon$ of an $\alpha$-robustly simplicial factorization.
Ensure: An $f \times r$ matrix $\boldsymbol{F}$ and $r \times n$ matrix $\boldsymbol{W}$ with $\boldsymbol{F} \geq \boldsymbol{0}$, $\boldsymbol{W} \geq \boldsymbol{0}$, and $\|\boldsymbol{X} - \boldsymbol{F}\boldsymbol{W}\|_{\infty,1} \leq 4\epsilon$.
Compute the $\boldsymbol{C} \in \Phi_{2\epsilon}(X)$ that minimizes $\boldsymbol{p}^T \operatorname{diag} \boldsymbol{C}$ where $\boldsymbol{p}$ is any vector with distinct values.
Let $I = \{i \ : \ C_{ii} = 1\}$ and set $\boldsymbol{W} = \boldsymbol{X}_{I\cdot}$.
Set $\boldsymbol{F} = \arg\min_{\boldsymbol{Z} \in \mathbb{R}^{f \times r}} \|\boldsymbol{X} - \boldsymbol{Z}\boldsymbol{W}\|_{\infty,1}$
```

Not only are our algorithms simpler than the AGKM algorithm, but they are also significantly more robust to perturbations. Unlike the AGKM algorithm, our algorithm does not need to know the parameter α. And, in the case that the factorization of $\boldsymbol{X}$ is unique, we can prove a stronger result.

Proposition 3.3 *Suppose $\boldsymbol{X}$ admits a rank-r separable factorization $\boldsymbol{F}\boldsymbol{W}$ with $\boldsymbol{W}$ α-robustly simplicial and $\boldsymbol{F}$ has the form (1) with $\|\boldsymbol{M}\|_\infty \leq \beta$. Suppose $\epsilon \leq \frac{1}{2}\alpha(1-\beta)$. Suppose $\hat{\boldsymbol{X}} = \boldsymbol{X} + \boldsymbol{\Delta}$ with $\|\Delta\|_{\infty,1} \leq \epsilon$. Then Algorithm 3 constructs a nonnegative factorization satisfying $\left\|\hat{\boldsymbol{X}} - \hat{\boldsymbol{F}}\hat{\boldsymbol{W}}\right\|_{\infty,1} \leq 4\epsilon$. Moreover, in this case, we need not enforce the constraint that $C_{ij} \leq C_{jj}$ for all i,j.*

The proof of Propositions 3.1, 3.2, and 3.3 can be found in the appendix. The main idea is to use LP duality to certify that the diagonal entries of $\boldsymbol{C}$ identify hott topics. For the noisy case, the structure of the dual norm of $(\infty, 1)$ is used to bound the error tolerance. In this regard, we can analyze robust recovery in any norm with minor modifications of our analysis.

Having established these theoretical guarantees, it now remains to find an algorithm to solve the LP. Off the shelf LP solvers may suffice for moderate-size problems, but for large scale matrix factorization problems, we show in Section 5 that their running time is prohibitive. In Section 4, we turn to describe how to solve Algorithm 3 efficiently for large data sets.

3.2 Related Work

Localizing factorizations via column or row subset selection is a popular alternative to direct factorization methods such as the SVD. Interpolative decomposition such as Rank-Revealing QR [12] and CUR [17] have favorable efficiency properties as compared to their exact counterparts. Factorization localization has been used in subspace clustering, and has been shown to be robust to outliers [8, 21].

In recent work on dictionary learning, Esser et al. proposed a factorization localization solution to nonnegative matrix factorization [9]. They showed excellent empirical performance and proved asymptotic exact recovery in a restricted noise model. Their work also requires preprocessing to remove duplicate or near duplicate rows. Our work here improves upon this work in several aspects, enabling finite sample error bounds, the elimination of any need to preprocess the data, and algorithmic implementations that scale to very large data sets.

4 Incremental Gradient Algorithms for NMF

The rudiments of our fast implementation rely on two standard optimization techniques: dual decomposition and incremental gradient descent. Both techniques are described in depth in Chapters 3.4 and 7.8 of Bertsekas and Tstisklis [5].

We aim to minimize $\boldsymbol{p}^T \operatorname{diag} \boldsymbol{C}$ subject to $\boldsymbol{C} \in \Phi_\tau(\boldsymbol{X})$. To proceed, form the Lagrangian

$$\mathcal{L}(\boldsymbol{C}, \beta, \boldsymbol{w}) = \boldsymbol{p}^T \operatorname{diag} \boldsymbol{C} + \beta(\operatorname{Tr}(\boldsymbol{C}) - r) + \sum_{i=1}^{f} w_i \left(\|\boldsymbol{X}_{i\cdot} - [\boldsymbol{C}\boldsymbol{X}]_{i\cdot}\|_1 - \tau \right)$$

with multipliers β and $\boldsymbol{w} \geq 0$. Note that we do not dualize out all of the constraints. We leave those in the constraint set $\Phi_0 = \{\boldsymbol{C} \ : \ \boldsymbol{C} \geq 0, \, , \operatorname{diag}(\boldsymbol{C}) \leq 1, \, , \text{and} \, , C_{ij} \leq C_{jj} \text{ for all } i, j\}$.

Dual subgradient ascent solves this problem by alternating between minimizing the Lagrangian over the constraint set Φ_0, and then taking a subgradient step with respect to the dual variables

$$w_i \leftarrow w_i + s \left(\|\boldsymbol{X}_{i\cdot} - [\boldsymbol{C}^{\boldsymbol{*}}\boldsymbol{X}]_{i\cdot}\|_1 - \tau \right) \quad \text{and} \quad \beta \leftarrow \beta + s(\operatorname{Tr}(\boldsymbol{C}^{\boldsymbol{*}}) - r)$$

where $\boldsymbol{C}^*$ is the minimizer of the Lagrangian over Φ_0. The update of w_i makes very little difference with respect to the solution quality and we typically only update β.

We minimize the Lagrangian using projected incremental gradient descent. Note that we can rewrite the Lagrangian as

$$\mathcal{L}(\boldsymbol{C}, \beta, \boldsymbol{w}) = -\tau \mathbf{1}^T \boldsymbol{w} - \beta r + \sum_{k=1}^{n} \left(\sum_{j \in \operatorname{supp}(\boldsymbol{X}_{\cdot k})} w_j \|\boldsymbol{X}_{jk} - [\boldsymbol{C}\boldsymbol{X}]_{jk}\|_1 + \mu_j (p_j + \beta) C_{jj} \right) .$$

Algorithm 4 HOTTOPIXX: Approximate Separable NMF by Incremental Gradient Descent

```
Require: An $f \times n$ nonnegative matrix $\boldsymbol{X}$. Primal and dual stepsizes $s_p$ and $s_d$.
Ensure: An $f \times r$ matrix $\boldsymbol{F}$ and $r \times n$ matrix $\boldsymbol{W}$ with $\boldsymbol{F} \geq \boldsymbol{0}$, $\boldsymbol{W} \geq \boldsymbol{0}$, and $\|\boldsymbol{X} - \boldsymbol{F}\boldsymbol{W}\|_{\infty,1} \leq 4\epsilon$.
Pick a cost $\boldsymbol{p}$ with distinct entries.
Initialize $\boldsymbol{C} = 0$, $\beta = 0$
for $t = 1, \ldots, N_{epochs}$ do
    for $i = 1, \ldots n$ do
        Choose $k$ uniformly at random from $[n]$.
        $\boldsymbol{C} \leftarrow \boldsymbol{C} + s_p \cdot \operatorname{sign}(\boldsymbol{X}_{\cdot k} - \boldsymbol{C}\boldsymbol{X}_{\cdot k})\boldsymbol{X}_{\cdot k}^T - s_p \operatorname{diag}(\boldsymbol{\mu} \circ (\beta \boldsymbol{1} - \boldsymbol{p}))$.
    end for
    Project $\boldsymbol{C}$ onto $\Phi_0$.
    $\beta \leftarrow \beta + s_d(\operatorname{Tr}(\boldsymbol{C}) - r)$
end for
Let $I = \{i \ : \ C_{ii} = 1\}$ and set $\boldsymbol{W} = \boldsymbol{X}_{I\cdot}$.
Set $\boldsymbol{F} = \arg\min_{\boldsymbol{Z} \in \mathbb{R}^{f \times r}} \|\boldsymbol{X} - \boldsymbol{Z}\boldsymbol{W}\|_{\infty,1}$
```

Here, $\operatorname{supp}(\boldsymbol{x})$ is the set indexing the entries where $\boldsymbol{x}$ is nonzero, and μ_j is the number of nonzeros in row j divided by n. The incremental gradient method chooses one of the n summands at random and follows its subgradient. We then project the iterate onto the constraint set Φ_0. The projection onto Φ_0 can be performed in the time required to sort the individual columns of $\boldsymbol{C}$ plus a linear time operation. The full procedure is described in Appendix B. In the case where we expect a unique solution, we can drop the constraint $C_{ij} \leq C_{jj}$, resulting in a simple clipping procedure: set all negative items to 0 and set any diagonal entry greater than 1 to 1. In practice, we perform a tradeoff. Since the constraint $C_{ij} \leq C_{jj}$ is used solely for symmetry breaking, we have found empirically that we only need to project onto Φ_0 every n iterations or so.

This incremental iteration is repeated n times in a phase called an *epoch*. After each epoch, we update the dual variables, and quit after we believe we have identified the large elements of the diagonal of $\boldsymbol{C}$. Just as before, once we have identified the hott rows, we can form $\boldsymbol{W}$ by selecting these rows of $\boldsymbol{X}$. We can find $\boldsymbol{F}$ just as before, by solving (2). Note that this minimization can also be computed by incremental subgradient descent. The full procedure, called HOTTOPIXX, is described in Algorithm 4.

4.1 Sparsity and Computational Enhancements for Large Scale.

For small-scale problems, HOTTOPIXX can be implemented in a few lines of Matlab code. But for the very large data sets studied in Section 5, we take advantage of natural parallelism and a host of low-level optimizations that are also enabled by our formulation. As in any numerical program, memory layout, and cache behavior can be critical factors for performance. We use standard techniques: in-memory clustering to increase pre-fetching opportunities, padded data structures for better cache alignment, and compiler directives to allow the Intel compiler to apply vectorization.

Note that the incremental gradient step (step 6 in 4), only modifies the entries of $\boldsymbol{C}$ where $\boldsymbol{X}_{\cdot k}$ is nonzero. Thus, we can parallelize the algorithm either with respect to updating the rows or the columns of $\boldsymbol{C}$. We store $\boldsymbol{X}$ in large contiguous blocks of memory to encourage hardware prefetching. In contrast, we choose a dense representation of our localizing matrix, $\boldsymbol{C}$; this choice trades space for runtime performance.

Each worker thread is assigned a number of rows of $\boldsymbol{C}$ so that all rows fit in the shared L3 cache. Then, each worker thread repeatedly scans $\boldsymbol{X}$ while marking updates to multiple rows of $\boldsymbol{C}$. We repeat this process until all rows of $\boldsymbol{C}$ are scanned, similar to the classical block-nested loop join in relational databases [19].

5 Experiments

Except for the speedup curves, all of the experiments were run on an identical configuration: a dual Xeon X650 (6 cores each) machine with 128GB of RAM. The kernel is Linux 2.6.32-131.

In small-scale, synthetic experiments, we compared Hottopixx to the AGKM algorithm and the linear program formulation of Algorithm 3 in Matlab. Both AGKM and Algorithm 3 were run using CVX [11] coupled to the SDPT3 solver [23]. We ran Hottopixx for 50 epochs with primal stepsize 1e-1 and dual stepsize 1e-2. Once the hott topics were identified, we fit $\boldsymbol{F}$ using two cleaning epochs of incremental gradient descent for all three algorithms.

To generate our instances, we sampled r hott topics uniformly from the unit simplex in $\mathbb{R}^n$. These topics were duplicated d times. We generated the remaining $f - r(d+1)$ rows to be random convex combinations of the hott topics, with the combinations selected uniformly at random. We then added noise with $(\infty, 1)$-norm error bounded by $\eta \cdot \frac{\alpha^2}{20+13\alpha}$. Recall that AGKM algorithm is only guaranteed to work for $\eta < 1$. We ran with $f \in \{40, 80, 160\}$, $n \in \{400, 800, 1600\}$, $r \in \{3, 5, 10\}$, $d \in \{0, 1, 2\}$, and $\eta \in \{0.25, 0.95, 4, 10, 100\}$. Each experiment was repeated 5 times.

Because we ran over 2000 experiments with 405 different parameter settings, it is convenient to use the *performance profiles* to compare the performance of the different algorithms [6]. Let $\mathcal{P}$ be the set of experiments and $\mathcal{A}$ denote the set of different algorithms we are comparing. Let $Q_a(p)$ be the value of some performance metric of the experiment $p \in \mathcal{P}$ for algorithm $a \in \mathcal{A}$. Then the performance profile at τ for a particular algorithm is the fraction of the experiments where the value of $Q_a(p)$ is within a factor of τ of the minimal value of $\min_{b \in \mathcal{A}} Q_b(p)$. That is

$$P_a(\tau) = \frac{\#\left\{p \in \mathcal{P} \;:\; Q_a(p) \leq \tau \min_{a' \in \mathcal{A}} Q_{a'}(p)\right\}}{\#(\mathcal{P})} \,.$$

In a performance profile, the higher a curve corresponding to an algorithm, the more it is outperforming the other algorithms. This provides a convenient way to visually contrast algorithms.

Our performance profiles are shown in Figure 2. The first two figures corresponds to experiments with $f = 40$ and $n = 400$. The third figure is for the synthetic experiments with all other values of f and n. In terms of $(\infty, 1)$-norm error, the linear programming solver typically achieves the lowest error. However, using SDPT3, it is prohibitively slow to factor larger matrices. On the other hand, Hottopixx achieves better noise performance than the AGKM algorithm in much less time. Moreover, the AGKM algorithm must be fed the values of ϵ and α in order to run. Hottopixx does not require this information and still achieves about the same error performance.

We also display a graph for running only four epochs (hott (fast)). This algorithm is by far the fastest algorithm, but does not achieve as optimal a noise performance. For very high levels of noise, however, it achieves a lower reconstruction error than the AGKM algorithm, whose performance degrades once η approaches or exceeds 1 (Figure 2(f)). We also provide performance profiles for the root-mean-square error of the nonnegative matrix factorizations (Figure 2 (d) and (e)). The performance is qualitatively similar to that for the $(\infty, 1)$-norm.

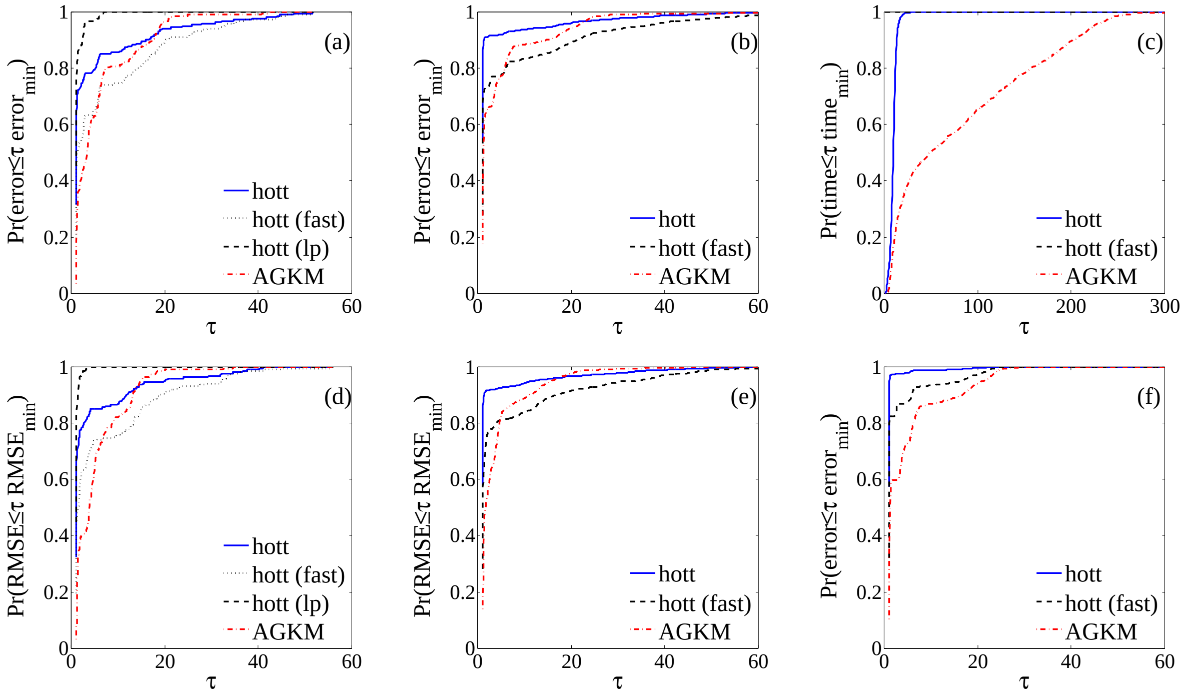

Figure 2: Performance profiles for synthetic data. (a) $(\infty, 1)$-norm error for 40×400 sized instances and (b) all instances. (c) is the performance profile for running time on all instances. RMSE performance profiles for the (d) small scale and (e) medium scale experiments. (f) $(\infty, 1)$-norm error for the $\eta \geq 1$. In the noisy examples, even 4 epochs of Hottopixx is sufficient to obtain competitive reconstruction error.

data set	features	documents	nonzeros	size (GB)	time (s)
jumbo	1600	64000	1.02e8	2.7	338
clueweb	44739	351849	1.94e7	0.27	478
RCV1	47153	781265	5.92e7	1.14	430

Table 1: Description of the large data-sets. Time is to find 100 hott topics on the 12 core machines.

We also coded Hottopixx in C++, using the design principles described in Section 4.1, and ran on three large data sets. We generated a large synthetic example (jumbo) as above with $r = 100$. We generated a co-occurrence matrix of people and places from the ClueWeb09 Dataset [2], normalized by TFIDF. We also used Hottopixx to select features from the RCV1 data set to recognize the class CCAT [16]. The statistics for these data sets can be found in Table 1.

In Figure 3 (left), we plot the speed-up over a serial implementation. In contrast to other parallel methods that exhibit memory contention [18], we see superlinear speed-ups for up to 20 threads due to hardware prefetching and cache effects. All three of our large data sets can be trained in minutes, showing that we can scale Hottopixx on both synthetic and real data. Our algorithm is able to correctly identify the hott topics on the jumbo set. For clueweb, we plot the RMSE Figure 3 (middle). This curve rolls off quickly for the first few hundred topics, demonstrating that our algorithm may be useful for dimensionality reduction in Natural Language Processing applications. For RCV1, we trained an SVM on the set of features extracted by Hottopixx and

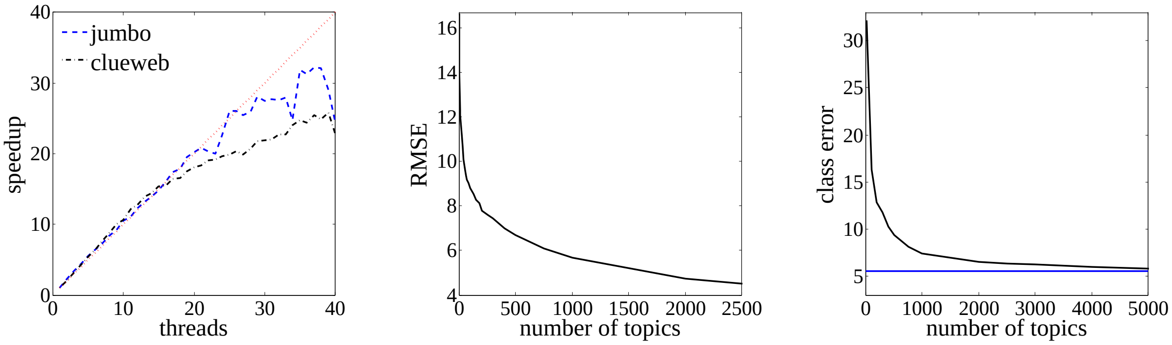

Figure 3: (left) The speedup over a serial implementation for Hottopixx on the jumbo and clueweb data sets. Note the superlinear speedup for up to 20 threads. (middle) The RMSE for the clueweb data set. (right) The test error on RCV1 CCAT class versus the number of hott topics. The blue line here indicates the test error achieved using all of the features.

plot the misclassification error versus the number of topics in Figure 3 (right). With 1500 hott topics, we are able to achieve 7% misclassification error as compared to 5.5% with the entire set of features.

6 Discussion

This paper provides an algorithmic and theoretical framework for analyzing and deploying any factorization problem that can be posed as a linear (or convex) factorization localizing program. Future work should investigate the applicability of Hottopixx to more data sets and factorization localizing algorithms, and should revisit earlier theoretical bounds on such prior art.

Our hope is that our results indicate how to translate and extend the beautiful theoretical insights of prior work into practical machine learning algorithms. Said another way, while AGKM aim to *"bring rigor to machine learning"*, our aim is to bring implementable algorithms to Theoretical Computer Science.

Acknowledgments

The authors would like to thank Michael Ferris and Steve Wright for helpful suggestions. BR is generously supported by ONR award N00014-11-1-0723 and NSF award CCF-1139953. CR is generously supported by the Air Force Research Laboratory (AFRL) under prime contract no. FA8750-09-C-0181, the NSF CAREER award under IIS-1054009, ONR award N000141210041, and gifts or research awards from Google, Greenplum, Johnson Controls, Inc., LogicBlox, and Oracle. JAT was generously supported by ONR awards N00014-08-1-0883 and N00014-11-1002, AFOSR award FA9550-09-1-0643, DARPA award N66001-08-1-2065, and a Sloan Research Fellowship. Any opinions, findings, and conclusion or recommendations expressed in this work are those of the authors and do not necessarily reflect the views of any of the above sponsors including DARPA, AFRL, or the US government.

References

[1] docs.oracle.com/cd/B28359_01/datamine.111/b28129/algo_nmf.htm.

[2] lemurproject.org/clueweb09/.

[3] www.mathworks.com/help/toolbox/stats/nnmf.html.

[4] S. Arora, R. Ge, R. Kannan, and A. Moitra. Computing a nonnegative matrix factorization – provably. To appear in STOC 2012. Preprint available at \arxiv.org/abs/1111.0952, 2011.

[5] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, Belmont, MA, 1997.

[6] E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming, Series A*, 91:201–213, 2002.

[7] D. Donoho and V. Stodden. When does non-negative matrix factorization give a correct decomposition into parts? In *Advances in Neural Information Processing Systems*, 2003.

[8] E. Elhamifar and R. Vidal. Sparse subspace clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[9] E. Esser, M. Möller, S. Osher, G. Sapiro, and J. Xin. A convex model for non-negative matrix factorization and dimensionality reduction on physical space. *IEEE Transactions on Image Processing*, 2012. To appear. Preprint available at arxiv.org/abs/1102.0844.

[10] R. Gaujoux and C. Seoighe. NMF: A flexible R package for nonnegative matrix factorization. *BMC Bioinformatics*, 11:367, 2010. doi:10.1186/1471-2105-11-367.

[11] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 1.21. http://cvxr.com/cvx, May 2010.

[12] M. Gu and S. C. Eisenstat. Efficient algorithms for computing a strong rank-revealing QR factorization. *SIAM Journal on Scientific Computing*, 17:848–869, 1996.

[13] T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd Annual International SIGIR Conference on Research and Development in Information Retrieval*, 1999.

[14] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.

[15] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems*, 2001.

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

[17] M. W. Mahoney and P. Drineas. CUR matrix decompositions for improved data analysis. *Proceedings of the National Academy of Sciences*, 106:697–702, 2009.

[18] F. Niu, B. Recht, C. Ré, and S. J. Wright. HOGWILD!: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems*, 2011.

[19] L. D. Shapiro. Join processing in database systems with large main memories. *ACM Transactions on Database Systems*, 11(3):239–264, 1986.

[20] P. Smaragdis. Non-negative matrix factorization for polyphonic music transcription. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 177–180, 2003.

[21] M. Soltanolkotabi and E. J. Candès. A geometric analysis of subspace clustering with outliers. Preprint available at arxiv.org/abs/1112.4258, 2011.

[22] L. B. Thomas. Problem 73-14, rank factorization of nonnegative matrices. *SIAM Review*, 16(3):393–394, 1974.

[23] K. C. Toh, M. Todd, and R. H. Tütüncü. *SDPT3: A MATLAB software package for semidefinite-quadratic-linear programming.* Available from `http://www.math.nus.edu.sg/~mattohkc/sdpt3.html`.

[24] S. A. Vavasis. On the complexity of nonnegative matrix factorization. *SIAM Joural on Optimization*, 20(3):1364–1377, 2009.

A Proofs

A.1 Dual Certificates

In all of our proofs, we will make use of the following linear program

$$\begin{array}{ll} \text{minimize}_{\boldsymbol{C},\boldsymbol{Q}} & \boldsymbol{t}^T \operatorname{diag}(\boldsymbol{C}) \\ \text{subject to} & \boldsymbol{C}\boldsymbol{X} - \boldsymbol{X} + \boldsymbol{Q} = 0 \\ & \|\boldsymbol{Q}\|_{\infty,1} \leq \tau \\ & \boldsymbol{C} \geq 0 \\ & \operatorname{diag}(\boldsymbol{C}) \leq \mathbf{1} \\ & \operatorname{Tr}(\boldsymbol{C}) = r \\ & C_{ij} \leq C_{jj} \text{ for all } i,j \end{array} \tag{3}$$

where the cost $\boldsymbol{t} \in \mathbb{R}^f$ will vary for different purposes. The dual problem of (3) is given by

$$\begin{array}{ll} \text{maximize}_{\boldsymbol{R},\boldsymbol{\gamma},\eta} & \operatorname{Tr}(\boldsymbol{R}\boldsymbol{X}^T) - \tau \|\boldsymbol{R}\|_{\infty,1}^* - \sum_{i=1}^f \gamma_i + r\eta \\ \text{subject to} & \boldsymbol{R}\boldsymbol{X}^T - \operatorname{diag}(\boldsymbol{\gamma}) + \operatorname{diag}(\mathbf{1}^T\boldsymbol{M}) - \boldsymbol{M} + \eta\boldsymbol{I} \leq \operatorname{diag}(\boldsymbol{t}) \\ & \boldsymbol{\gamma} \geq 0,\ \boldsymbol{M} \geq 0 \end{array} \tag{4}$$

This duality can be verified by forming the Lagrangian

$$\mathcal{L} = \boldsymbol{t}^T \operatorname{diag}(\boldsymbol{C}) - \operatorname{Tr}(\boldsymbol{R}^T(\boldsymbol{C}\boldsymbol{X} - \boldsymbol{X} + \boldsymbol{Q})) + \boldsymbol{\gamma}^T(\operatorname{diag}(\boldsymbol{C}) - \mathbf{1}) - \eta(\operatorname{Tr}(\boldsymbol{C}) - r) + \sum_{i,j} M_{ij}(C_{ij} - C_{jj})$$

with $\boldsymbol{\gamma} \geq 0$ and $\boldsymbol{M} \geq 0$. Minimizing with respect to $\boldsymbol{C} \geq 0$ and $\boldsymbol{Q}$ with $\|\boldsymbol{Q}\|_{\infty,1} \leq \tau$ yields the dual problem.

The KKT conditions for this problem are primal and dual feasibility along with the complementary slackness conditions

$$\begin{array}{rl} 0 \leq \boldsymbol{C} \perp & \boldsymbol{R}\boldsymbol{X}^T - \operatorname{diag}(\boldsymbol{\gamma}) + \operatorname{diag}(\mathbf{1}^T\boldsymbol{M}) - \boldsymbol{M} + \eta\boldsymbol{I} - \operatorname{diag}(\boldsymbol{t}) \leq 0 \\ 0 \leq C_{jj} - C_{ij} \perp & M_{ij} \geq 0 \qquad \text{for all } i,j \\ 0 \leq 1 - \operatorname{diag}(C) \perp & \boldsymbol{\gamma} \geq 0\,. \end{array} \tag{5}$$

A.2 Separators

All of our proofs will make use of *separators* $\boldsymbol{\rho}_i$ which will have positive dot product with one hott topic and negative dot products with the others. Since we assume that there is a set of r hott topics that form a simplex, this will always be possible to find.

For a hott topic i, let $\boldsymbol{t} = \boldsymbol{e}_i$. Let I denote a set of simplicial rows of $\boldsymbol{X}$. Then we can find a collection of vectors $\boldsymbol{\rho}_i$, for $i \in I$, satisfying

$$\begin{array}{ll} \boldsymbol{X}_{i\cdot}\boldsymbol{\rho}_i = u_i & \\ \boldsymbol{X}_{j\cdot}\boldsymbol{\rho}_i \leq -v_i & \text{for } j \in I \setminus \{i\} \end{array} \tag{6}$$

for any u and v greater than or equal to zero.

To see how to apply these separators, we first prove the following useful Lemma. Let L be the set of rows that cannot form a simplicial decomposition of $\boldsymbol{X}$. That is, L is the set of *not hott* topics.

Lemma A.1 *If $\ell \in L$, $C_{\ell\ell} = 0$ for all $\boldsymbol{C} \in \Phi(\boldsymbol{X})$.*

Proof For $\ell \in L$, consider the linear program

$$\underset{\boldsymbol{C}\in\Phi(\boldsymbol{X})}{\text{minimize}} \;\; -\boldsymbol{e}_\ell^T \operatorname{diag}(\boldsymbol{C}) \tag{7}$$

Proving that the optimal value of this problem is zero proves the lemma. For each $i \in I$, let $D(i)$ denote the set of rows of $\boldsymbol{X}$ that are equal to $\boldsymbol{X}_i$. Let $\{\boldsymbol{\rho}_i\}$ denote a set of separators satisfying (6) with $u_i = 1/|D(i)|$ and $v_i = 0$. Then set $\boldsymbol{R}_{i\cdot} = \boldsymbol{\rho}_i^T$ for $i \in I$, $\boldsymbol{R}_{i\cdot} = 0$ otherwise. Let $\eta = -1$, $\gamma_i = 0$ and $M_{ij} = 1/|D(i)|$ for $j \in D(i)$. Let all other $M_{ij} = 0$. Then we have found a dual feasible solution with dual cost 0. ■

A.3 Proof of Proposition 3.1

Let $\boldsymbol{t}$ be any vector with distinct, positive entries. For example, $t_i = i$ for $i = 1, \ldots, f$. Consider the linear program

$$\underset{\boldsymbol{C}\in\Phi(\boldsymbol{X})}{\text{minimize}} \;\; \boldsymbol{t}^T \operatorname{diag}(\boldsymbol{C}) \tag{8}$$

Let $\boldsymbol{C}_0$ denote the factorization localizing matrix which identifies the factorization with lowest cost $\boldsymbol{t}^T \operatorname{diag}(\boldsymbol{C})$ and has either ones or zeros on the diagonal. Then $\boldsymbol{C}_0$ is the unique optimal solution of (8).

To see this, let I denote the set of hott topics of minimum cost. For each $i \in I$, let $D(i)$ denote the set of indices k where $\boldsymbol{X}_{k\cdot} = \boldsymbol{X}_{i\cdot}$. Suppose the jth column of $\boldsymbol{C}_1$ is not zero for some i and $j \in D(i)$. Then we can find a solution of lower cost by adding the weights to the ith column, zeroing out the jth diagonal entry. Thus we can assume that $\boldsymbol{C}_1$ has zeros on the diagonal unless $i \in I \cup L$. But Lemma A.1 shows that all entries in L must be zero.

A.4 Proof of Proposition 3.3

Suppose $\boldsymbol{C}$ satisfies $\left\|\boldsymbol{C}\hat{\boldsymbol{X}} - \hat{\boldsymbol{X}}\right\|_{\infty,1} \leq \epsilon$. Then we have

$$\|\boldsymbol{C}\boldsymbol{X} - \boldsymbol{X}\|_{\infty,1} \leq \left\|\boldsymbol{C}\hat{\boldsymbol{X}} - \hat{\boldsymbol{X}}\right\|_{\infty,1} + \|\boldsymbol{C}\boldsymbol{\Delta} - \boldsymbol{\Delta}\|_{\infty,1} \leq 2\epsilon$$

and a similar inequality holds swapping the roles of $\boldsymbol{X}$ and $\hat{\boldsymbol{X}}$.

The goal is now to show that if $\boldsymbol{C}$ is feasible for

$$\left\|\boldsymbol{C}\hat{\boldsymbol{X}} - \hat{\boldsymbol{X}}\right\|_{\infty,1} \leq 2\epsilon\,, \boldsymbol{C} \geq 0,\;\; \mathrm{diag}(\boldsymbol{C}) \leq 1,\;\; \mathrm{Tr}(\boldsymbol{C}) = r \tag{9}$$

then the matrix $\boldsymbol{C}$ identifies the hott topics. In fact, we will show that the hott topics will correspond to diagonal entries of $\boldsymbol{C}$ that are greater than 1/2, and all other topics will have diagonal entries less than 1/2.

To prove this, first consider the hott topics. Let $\boldsymbol{t} = \boldsymbol{e}_i$ in (3). Suppose we can find a set of separators $\boldsymbol{\rho}_i$ with $u_i = 1$, $v_i = \frac{-\beta}{1-\beta}$, and $\|\boldsymbol{\rho}_i\|_\infty \leq \frac{1}{4\epsilon}$ for all i. Then if we set $\boldsymbol{R}$ to be equal to zero everywhere except in row i where we let $\boldsymbol{R}_{i\cdot} = \boldsymbol{\rho}_i^T$, we set $\gamma_j = 0$ for all i and $\eta = 0$, we see that we have a feasible assignment for (4) with cost 1/2. To verify feasibility just note that $\boldsymbol{X}_{j\cdot}\boldsymbol{\rho}_i < 0$ for all $j \neq i$ because if $\boldsymbol{X}_{j\cdot}$ is a hott topic, $\boldsymbol{X}_{j\cdot}\boldsymbol{\rho}_i < 0$ by construction. If $\boldsymbol{X}_{j\cdot}$ is not hott, then it is a convex combination of hott topics: $\boldsymbol{X}_{j\cdot} = \delta\boldsymbol{X}_{i\cdot} + (1-\delta)\boldsymbol{z}$ with $\boldsymbol{z}$ in the convex hull of $\boldsymbol{X}_{k\cdot}$ with $k \in [r] \setminus \{i\}$. By assumption, $\delta \leq \beta$ and

$$(\delta\boldsymbol{X}_{i\cdot} + (1-\delta)\boldsymbol{z})\boldsymbol{\rho}_i \leq \beta - (1-\beta)\frac{\beta}{1-\beta} \leq 0\,.$$

Together, this proves that is no feasible assignment for (3) where $C_{ii} \leq 1/2$.

The existence of the $\boldsymbol{\rho}_i$ with controlled ℓ_∞ norms follows because we assumed that $\boldsymbol{X}$ was robustly separable. Thus, there exists a hyperplane ρ that strictly separates $\boldsymbol{X}_{i\cdot}$ from the convex hull of $\boldsymbol{X}_{k\cdot}$ with $k \in [r] \setminus i$. To estimate the ℓ_∞ norm of ρ, consider the linear program

$$\text{minimize } \;\|\boldsymbol{\rho}\|_\infty \quad \text{subject to } (\boldsymbol{X}_{j\cdot} - \boldsymbol{X}_{i\cdot})\boldsymbol{\rho} \leq \frac{-1}{1-\beta} \quad \text{for } \; j \in [r] \setminus \{i\}$$

The dual problem is

$$\text{maximize } \;(1-\beta)^{-1}\sum_{i=1}^{r-1} u_i \quad \text{subject to } \|\sum_j u_i(\boldsymbol{X}_{j\cdot} - \boldsymbol{X}_{i\cdot})\|_1 \leq 1, \quad u \geq 0$$

which has a feasible assignment $u_i = \frac{1}{(r-1)\alpha}$. Thus, we can find a ρ which robustly separates hott topic $\boldsymbol{X}_{i\cdot}$ from the rest with infinity-norm at most $\frac{1}{\alpha(1-\beta)}$.

To show the not hott topics have bounded diagonals, set $\boldsymbol{t} = -\sum_{j \notin I} \boldsymbol{e}_j$. For the hott topics, set $\boldsymbol{R}_{i\cdot} = \boldsymbol{\rho}_i^T$. Set all of the remaining $\boldsymbol{R}_{j\cdot} = 0$. Set $\eta = -1$ and $\gamma_i = 0$ for all i. Then the dual cost is at least $-1/2$, and hence the largest a non-hott topic diagonal can be is 1/2.

Once we have identified the hott topics, I, we simply solve the second linear program

$$\underset{\boldsymbol{B}}{\text{minimize}} \left\|\hat{\boldsymbol{X}} - \begin{bmatrix} \boldsymbol{I} \\ \boldsymbol{B} \end{bmatrix} \hat{\boldsymbol{X}}_I\right\|_{\infty,1} \tag{10}$$

to find an ϵ-accurate factorization.

A.5 Proof of Proposition 3.2

This proposition more or less follows from the proof of Proposition 3.2, but we have to again rule out the case of duplicate or near-duplicate rows in the noiseless $\boldsymbol{X}$.

In this case, we minimize a linear cost $\boldsymbol{t}^T \operatorname{diag}(\boldsymbol{C})$, where $\boldsymbol{t}$ has distinct values, over the constraint set (9). Define $\beta = 4\epsilon/\alpha$. Then all vectors which have simplicial decompositions with ℓ_∞ norm at most β are guaranteed to have total diagonal mass of at most $1/2$. It remains to show that the LP will assign at least $1 - \frac{1}{2r}$ weight to r rows that form the basis for an accurate factorization of $\hat{\boldsymbol{X}}$.

To proceed, let I again denote a set of indices for r distinct hott topics and let

$$\Omega_i = \left\{ j \ : \ \left\| \hat{\boldsymbol{X}}_{j\cdot} - \hat{\boldsymbol{X}}_{j\cdot} \right\|_{\infty,1} \leq 2\epsilon \right\} .$$

We now show that the LP will assign large mass to r diagonal entries, each corresponding to an index in one of the Ω_i.

Suppose that the algorithm fails to assign mass to all of the indices in Ω_i for some $i \in I$. Pick any $j \in \Omega_i$, and let K denote the set of indices with non-zero diagonal entries in the optimal $\boldsymbol{C}$. Then for any vector $\boldsymbol{p}$ with $\boldsymbol{p} \geq 0$ and $\mathbf{1}_{|K|}^T \boldsymbol{p} = 1$, we have

$$\begin{aligned} \left\| \hat{\boldsymbol{X}}_{j\cdot} - \sum_{k \in K} p_k \hat{\boldsymbol{X}}_{k\cdot} \right\|_1 &\geq \left\| \boldsymbol{X}_{j\cdot} - \sup_{\boldsymbol{p} \geq 0,\ \mathbf{1}_{|K|}^T \boldsymbol{p} = 1} \sum_{k \in K} p_k \boldsymbol{X}_{k\cdot} \right\|_1 - \left\| \boldsymbol{\Delta}_{j\cdot} - \sum_{k \in K} p_k \boldsymbol{\Delta}_{k\cdot} \right\|_1 \\ &\geq \alpha - \left\| \sum_{k \in K} p_k (\boldsymbol{\Delta}_{j\cdot} + \boldsymbol{\Delta}_{k\cdot}) \right\|_{\infty,1} \\ &\geq \alpha - 2(\beta + \epsilon) = \alpha - 8\epsilon/\alpha - 2\epsilon \geq 2\epsilon \, . \end{aligned}$$

And hence the assignment must be infeasible. Therefore, one element of each Ω_i has positive mass on the diagonal. For any choice of representatives J with a unique element from each Ω_i, we can find a feasible $\boldsymbol{C}$ with $C_{jj} \geq 1 - \frac{1}{2r}$ for $j \in J$. Therefore, the lowest cost assignment will choose exactly one member from each cluster and assign it mass $1 - \frac{1}{2r}$. Once we have identified the large diagonal entries of $\boldsymbol{C}$, we can again recover a full factorization by solving (10).

B Projection onto Φ_0

To project onto the set Φ_0, note that we can compute the projection one column at a time. Moreover, the projection for each individual column amounts to (after permuting the entries of the column),

$$\{ \boldsymbol{x} \in \mathbb{R}^f \ : \ 0 \leq x_i \leq x_1 \ \forall i \, , x_1 \leq 1 \} \, .$$

Assume, again without loss of generality, that we want to project a vector $\boldsymbol{z}$ with $z_2 \geq z_3 \geq \ldots \geq z_n$. Then we need to solve the quadratic program

$$\begin{array}{ll} \text{minimize} & \frac{1}{2} \| \boldsymbol{z} - \boldsymbol{x} \|^2 \\ \text{subject to} & 0 \leq x_i \leq x_1 \ \forall i \, , x_1 \leq 1 \end{array} \tag{11}$$

The optimal solution can be found as follows. Let k_c be the largest $k \in \{2, \ldots, f\}$ such that

$$z_{k_c + 1} \leq \Pi_{[0,1]} \left(\sum_{k=1}^{k_c} z_k \right) =: \mu$$

where $\Pi_{[0,1]}$ denotes the projection onto the interval $[0,1]$. Set

$$\hat{x}_i = \begin{cases} \mu & i \leq k_c \\ (z_i)_+ & i > k_c \end{cases} .$$

Then $\hat{\boldsymbol{x}}$ is the optimal solution. A linear time algorithm for computing $\hat{x}$ is given by Algorithm 5

To prove that $\hat{\boldsymbol{x}}$ is optimal, define

$$y_i = \begin{cases} z_i - \mu & i \leq k_c \\ \min(z_i, 0) & i > k_c \end{cases} .$$

y_i is the gradient of $\frac{1}{2}\|\boldsymbol{x} - \boldsymbol{z}\|^2$ at $\hat{x}$. Consider the LP

$$\begin{array}{ll} \text{minimize} & -\boldsymbol{y}^T\boldsymbol{x} \\ \text{subject to} & 0 \leq x_i \leq x_1 \ \forall i \,, x_1 \leq 1 \end{array} .$$

$\hat{\boldsymbol{x}}$ is an optimal solution for this LP because the cost is negative on the negative entries, 0 on the nonnegative entries that are larger than k_c, positive for $2 \leq k \leq k_c$, and nonpositive for $k = 1$. Hence, by the minimum principle, $\hat{\boldsymbol{x}}$ is also a solution of (11).

Algorithm 5 Column Squishing

```
Require: A vector $\boldsymbol{z} \in \mathbb{R}^f$ with $z_2 \geq z_3 \geq \ldots \geq z_n$.
Ensure: The projection of $\boldsymbol{z}$ onto $\{\boldsymbol{x} \in \mathbb{R}^f \ : \ 0 \leq x_i \leq x_1 \ \forall i \,, x_1 \leq 1\}$.
$\mu \leftarrow z_1$.
for $k = 2, \ldots, f$ do
    if $z_k \leq \Pi_{[0,1]}(\mu)$, Set $k_c = k - 1$ and break
    else set $\mu = \frac{k-1}{k}\mu + \frac{1}{k}z_k$.
end for
$x_1 \leftarrow \Pi_{[0,1]}(\mu)$
for $k = 2, \ldots, k_c$ set $x_k = \Pi_{[0,1]}(\mu)$.
for $k = (k_c + 1), \ldots, f$ set $x_k = (z_i)_+$.
return $\boldsymbol{x}$
```