Convex NMF on Non-Convex Massiv Data

Kristian Kersting[1] and **Mirwaes Wahabzada**[1] and **Christian Thurau**[2] and **Christian Bauckhage**[2]
[1]Knowledge Discovery Department, [2]Vision and Social Media Group
Fraunhofer IAIS, Schloss Birlinghoven, 53754 Sankt Augustin, Germany
firstname.lastname@iais.fraunhofer.de

Abstract

We present an extension of convex-hull non-negative matrix factorization (CH-NMF) which was recently proposed as a large scale variant of convex non-negative matrix factorization (C-NMF) or Archetypal Analysis (AA). CH-NMF factorizes a non-negative data matrix V into two non-negative matrix factors $V \approx WH$ such that the columns of W are convex combinations of certain data points so that they are readily interpretable to data analysts. There is, however, no free lunch: imposing convexity constraints on W typically prevents adaptation to intrinsic, low dimensional structures in the data. Alas, in cases where the data is distributed in a non-convex manner or consists of mixtures of lower dimensional convex distributions, the cluster representatives obtained from CH-NMF will be less meaningful. In this paper, we present a hierarchical CH-NMF that automatically adapts to internal structures of a dataset, hence it yields meaningful and interpretable clusters for non-convex datasets. This is also conformed by our extensive evaluation on DBLP publication records of 760,000 authors, 4,000,000 images harvested from the web, and 150,000,000 votes on World of Warcraft guilds.

1 Introduction

Modern applications of data mining and machine learning in computer vision, natural language processing, computational biology, and other areas often consider massive datasets and we need to run expensive algorithms such as principle component analysis (PCA), latent Dirichlet allocation (LDA), or non-negative matrix factorization (NMF) to extract meaningful, low-dimensional representations.

If the data are words contained in documents, these methods yield topic models representing each document as a mixture of a small number of topics and each word is attributable to one of the topics. In computer vision, where it is common to represent images as vectors in a high-dimensional space, they extract visual words and have been used for face and object recognition, or color segmentation. Social networks such as Flickr, Facebook, or Myspace allow for a wide range of interactions amongst their members, resulting in massive, temporal datasets relating users, media objects, and actions. Here, low-dimensional representations may identify and summarize common social activities.

Therefore, given massive matrices of hundreds of millions of entries, how can we *efficiently* factorize them? How can we create *meaningful*, low-dimensional representations? How can we gain inside into the dataset? These are precisely the questions which we address in this paper.

A recent positive development in data mining and machine learning has been the realization that massive datasets are not only challenging but may as well be viewed as an opportunity [Torralba *et al.*, 2008; Talwalkar *et al.*, 2008]. Machine learning and data mining techniques typically consist of two parts: the model and the data. Most effort in recent years has gone into the modeling part. Massive datasets, however, allow one to move into the opposite direction: *how much can the data itself help us to solve the problem*? Halevy *et al.* [2009] even speak of the "*the unreasonable effectiveness of data*". Massive datasets are likely to capture even very rare aspects of the problem at hand. Along this line, Thurau *et al.* [2009] have recently introduced a data-driven NMF approach, called convex-hull NMF, that is fast and scales extremely well: it can efficiently factorize gigantic matrices and in turn extract meaningful "clusters" from massive datasets containing millions of images and ratings. The key idea is to restrict the "clusters" to be combinations of vertices of the convex hull of the dataset; thus directly exploring the data itself to solve the convex NMF problem.

There is, however, no free lunch: by restricting the "clusters" to combinations of vertices of the convex hull of the dataset, convex-hull NMF cannot adapt to the intrinsic, low dimensional structure of the data anymore. Intuitively, for data modeled by Gaussians, i.e., as a combination of convex sets, convex-hull NMF will assign clusters to the "extreme" Gaussians and not to each Gaussian. Our main contribution is a simple and, hence, powerful and scalable generalization of convex hull NMF that automatically adapts to the intrinsic (low dimensional) structure in the data. The main insight is that one can use FastMap due to Faloutsos and Lin [1995] within convex-hull NMF to compute the convex hull vertices on massive datasets. Consequently, we can still solve the convex NMF problem but additionally adapt to the intrinsic structure in the data, when we split the data along each 1D FastMap line recursively, stop when some minimum node size is reached, and apply a post-pruning step. This way, we get meaningful clusters that respect the structure of the data, i.e., that diversify the results even better than convex-hull NMF. Our extensive experimental evaluation shows that the method, called hierarchical convex-hull NMF, achieves similar reconstruction quality as convex-hull NMF with only a small overhead while it produces more diversified clusters on DBLP publication records of 760.000 authors, 4 million tiny im-

Figure 1: The basis vectors resulting from different NMF variants applied to the CBCL Face Database 1. (a) Standard NMF results in part-based, sparse representations. Data points cannot be expressed as convex combinations of these basis elements. (b) Convex NMF (C-NMF) yields basis elements that allow for convex combinations. Moreover, the basis vectors are "meaningful" since they closely resemble given data points. They are, however, not indicative of characteristic variations among individual samples. (c) Convex-Hull NMF (CH-NMF) diversifies the basis vectors resulting in pale faces, faces with glasses, faces with beards, and so on. (d) Hierarchical CH-NMF (HCH-NMF) as proposed in the current paper also diversifies the results. Additionally, it automatically groups them, i.e., it identifies structure within the data. The induced hierarchical decomposition is shown together with the number of images falling into the corresponding subtrees.

ages, and 150 million votes on World of Warcraft guilds.

We proceed as follows. We start off by briefly reviewing non-negative matrix factorization (NMF) in Section 2, including convex NMF and convex-hull NMF. Then, we introduce hierarchical convex-hull NMF in Section 3. Before concluding, we present our extensive experimental evaluation in Section 4.

2 Non-Negative Matrix Factorization

Assume an $m \times n$ input data matrix $\mathbf{V} = (\mathbf{v_1}, \ldots, \mathbf{v_n})$ consisting of n column vectors of dimensionality m. We consider factorizations of the form $\mathbf{V} \approx \mathbf{W}^{m \times k}\mathbf{H}^{k \times n}$. The resulting matrix $\mathbf{W}$ contains a set of $k \ll n$ basis vectors which are linearly combined using the coefficients in $\mathbf{H}$ to represent the data. Common approaches to achieve such a factorization include Principal Component Analysis (PCA) [Jolliffe, 1986], Singular Value Decomposition (SVD) [Golub and van Loan, 1996], Vector Quantization (VQ), or non-negative Matrix Factorization (NMF) [Lee and Seung, 1999].

Various variants and improvements to NMF have been introduced in recent years. For example, Cai *et al.* [2008] presented a matrix factorization that obeys the geometric data structure. In [Kim and Park, 2008], a speed improvement to NMF is achieved using a novel algorithm based on an alternating nonnegative least squares framework. Another interesting variation is presented in [Suvrit, 2008] where optimization is based on a block-iterative acceleration technique. Recently, Mairal *et al.* [2010] have presented a very elegant online NMF approach based on sparse coding that also scales to large matrices (for additional related work, please see reference in [Mairal *et al.*, 2010]). In this work, however, we build on *Convex-NMF (C-NMF)* recently introduced by Ding *et. al* [2009], and it is not clear how to adapt these advanced NMF techniques to it as Convex-NMF represents the data matrix $\mathbf{V}$ as a convex combination of data points, i.e. $\mathbf{V} = \mathbf{VGH}^T$ where each column i of $\mathbf{G}$ is a stochastic vector that obeys $\|\mathbf{g}_i\|_1 = 1, \mathbf{g}_i \geq \mathbf{0}$. This is akin to *Archetypal Analysis* according to Cutler and Breiman [1994] where both matrices $\mathbf{G}$ and $\mathbf{H}^T$ are to be stochastic. Convex-NMF yields interesting interpretations of the data because each data point is now expressed as a weighted sum of certain data points (see Fig. 1).

Convex NMF Convex non-negative matrix factorization (C-NMF) was introduced by Ding *et al.* [2009] and minimizes $J = \|\mathbf{V} - \mathbf{VGH}^T\|^2$, where $\mathbf{V} \in \mathbb{R}^{m \times n}, \mathbf{G} \in \mathbb{R}^{n \times k}, \mathbf{H} \in \mathbb{R}^{n \times k}$. The matrices $\mathbf{G}$ and $\mathbf{H}$ are updated iteratively until convergence using the following update rules

$$G_{ik} = G_{ik}\sqrt{\frac{(\mathbf{Y}^+\mathbf{H})_{ik} + (\mathbf{Y}^-\mathbf{GH}^T\mathbf{H})_{ik}}{(\mathbf{Y}^-\mathbf{H})_{ik} + (\mathbf{Y}^+\mathbf{GH}^T\mathbf{H})_{ik}}} \quad (1)$$

and

$$H_{ik} = H_{ik}\sqrt{\frac{(\mathbf{Y}^+\mathbf{G})_{ik} + (\mathbf{HG}^T\mathbf{Y}^-\mathbf{G})_{ik}}{(\mathbf{Y}^-\mathbf{G})_{ik} + (\mathbf{HG}^T\mathbf{Y}^+\mathbf{G})_{ik}}} \quad (2)$$

where $\mathbf{Y} = \mathbf{V}^T\mathbf{V}$, and the matrices $\mathbf{Y}^+$ and $\mathbf{Y}^-$ are given by $Y^+_{ik} = \frac{1}{2}|Y_{ik}| + Y_{ik}$ and $Y^-_{ik} = \frac{1}{2}|Y_{ik}| - Y_{ik}$, respectively.

For the initialization of $\mathbf{G}$ and $\mathbf{H}$ two methods are proposed. The first initializes to (almost) unary representations based on a k-means clustering of $\mathbf{V}$. The second assumes a given NMF or Semi-NMF solution. For further details on the algorithm and its initializations we refer to [Ding *et al.*, 2009].

Recall, however, that our goal is to analyse massive, high-dimensional datasets. Unfortunately, the C-NMF update rules (1) and (2) have a time complexity of $O(n^2)$. Moreover, although the iterative algorithm comes down to simple matrix multiplications, the size of the involved matrices quickly becomes another limiting factor (similar to the intermediate blowup problem in tensor decomposition [Kolda and Sun, 2008]), since $\mathbf{V}^T\mathbf{V}$ results in an $n \times n$ matrix. Switching to an online update rule would avoid memory issues but it would at the same time introduce additional computational overhead. Overall, we can say that C-NMF does not scale to large datasets. In the following, we will review *Convex-Hull NMF (CH-NMF)*, which is a recent C-NMF method that is well suited for for large-scale data analysis.

Convex-Hull NMF Convex-Hull NMF aims at a data factorization based on the data points residing on the data convex hull. Such a data reconstruction has two interesting properties: first, the basis vectors are real data points and mark, unlike in most other clustering/factorization techniques, the most extreme and not the most common data points. Second, any data point can be expressed as a convex and meaningful combination of these basis vectors. This

Algorithm 1: CH-NMF

Input: Data matrix $\mathbf{V}^{m \times n}$
Output: Matrices $\mathbf{X}$ and $\mathbf{H}$

1. Compute k eigenvectors $\mathbf{e}_l$, $l = 1 \ldots k$ of the covariance matrix of $\mathbf{V}^{m \times n}$;
2. Project $\mathbf{V}$ onto the 2D-subspaces $\mathbf{E}_{o,q}^{2 \times n} = \mathbf{V}^T[\mathbf{e}_o, \mathbf{e}_q]$, $o = 1 \ldots k$, $q = 1 \ldots k, o \neq q$;
3. Compute and mark convex hull data points $H_{cvx}(\mathbf{E}_{o,q})$ for each 2D projection;
4. Combine marked convex hull data points (using the original data dimensionality m) $\mathbf{S}^{m \times p} = \{H_{cvx}(\mathbf{E}_{1,2}), \ldots, H_{cvx}(\mathbf{E}_{k-1,k})\}$;
5. Optimize $J_S = \|\mathbf{S} - \mathbf{S}\mathbf{I}^{p \times k}\mathbf{J}^{k \times p}\|^2$ such that $\|\mathbf{i}_i\|_1 = 1$ for $\mathbf{i}_i \geq \mathbf{0}$ and $\|\mathbf{j}_i\|_1 = 1$ for $\mathbf{j}_i \geq \mathbf{0}$;
6. Optimize $J = \|\mathbf{v}_i - \mathbf{X}\mathbf{h}_i^T\|^2$ for $i = 1 \ldots n$ where $\mathbf{X} = \mathbf{S}^{m \times p}\mathbf{I}^{p \times k}$ such that $\|\mathbf{h}_i\|_1 = 1$ and $\mathbf{h}_i \geq \mathbf{0}$;

offers interesting new opportunities for data interpretation as indicated in Fig. 1 and demonstrated in [Thurau *et al.*, 2009].

More precisely, following Ding *et al.* [2009], one seeks a factorization of the form $\mathbf{V} = \mathbf{V}\mathbf{G}\mathbf{H}^T$, where $\mathbf{V} \in \mathbb{R}^{m \times n}, \mathbf{G} \in \mathbb{R}^{n \times k}, \mathbf{H} \in \mathbb{R}^{n \times k}$. One further restricts the columns of $\mathbf{G}$ and $\mathbf{H}$ to convexity, i.e., $\|\mathbf{g}_i\|_1 = 1, \mathbf{g}_i \geq \mathbf{0}$ and $\|\mathbf{h}_j\|_1 = 1, \mathbf{h}_j \geq \mathbf{0}$. Indeed, Ding *et al.* [2009] also consider convex combinations but not for the matrix $\mathbf{H}$. In other words, CH-NMF aims at factorizing the data such that each data point is expressed as a convex combination of convex combinations of specific data points. The task now is to minimize

$$J = \|\mathbf{V} - \mathbf{V}\mathbf{G}\mathbf{H}^T\|^2 \tag{3}$$

such that $\|\mathbf{g}_i\|_1 = 1, \mathbf{g}_i \geq \mathbf{0}$ and $\|\mathbf{h}_j\|_1 = 1, \mathbf{h}_j \geq \mathbf{0}$. To do so, one sets $\mathbf{X} = \mathbf{V}^{d \times n}\mathbf{G}^{n \times k}$. The intuition is as follows. Since we assume a convex combination for $\mathbf{X}$, and by definition of the convex hull, the convex hull $H_{cvx}(\mathbf{V})$ of $\mathbf{V}$ must contain $\mathbf{X}$. Obviously, we could achieve a perfect reconstruction, giving $J = 0$, by setting $\mathbf{G}$ so that it would contain exactly one entry equal to 1 for each convex hull data point while all other entries were set to zero. Or more informal: *following the definition of the convex hull we can perfectly reconstruct any data point by a convex combination of convex hull data points.* Therefore, our goal becomes to solve Eq. (3) by finding k appropriate data points on the convex hull:

$$J = \|\mathbf{V} - \mathbf{X}\mathbf{H}^T\|^2 \tag{4}$$

such that $\mathbf{x}_i \in H_{cvx}(\mathbf{V}),\ i = 1, \ldots, k$.

Finding a solution to Eq. (4), however, is not necessarily straight forward. It is known that the worst case complexity for computing the convex hull of n points in m dimensions is $\Theta(n^{\frac{m}{2}})$. Moreover, the number of convex hull data points may tend to n for high dimensional spaces, see e.g. [Donoho and Tanner, 2005; Hall *et al.*, 2005] so that computing the convex hull of large data-sets quickly becomes practically infeasible. CH-NMF therefore seeks an approximate solution by subsampling the convex hull. It exploits the fact that any data point on the convex hull of a linear lower dimensional projection of the data also resides on the convex hull in the original data dimension. Since $\mathbf{V}$ contains finitely many points and therefore forms a polytope in $\mathbb{R}^m$, we can resort to the *main theorem of polytope theory*, see e.g. [Ziegler, 1995]. In our context, it says that every vertex of an affine image of P, i.e., every point of the convex hull of the image of P, corresponds to a vertex of P. Therefore computing the convex hull of several 2D affine projections of the data offers a way of subsampling $H_{cvx}(\mathbf{V})$. This is an efficient way as computing the convex hull of a set of 2D points can be done in $O(n \log n)$ time, [de Berg *et al.*, 2000].

This subsampling strategy is the main idea underlying CH-NMF and, indeed, various methods can be used for linearly projecting the data to a 2D space. Thurau *et al.* [2009] proposed to use PCA, i.e., projecting the data using pairwise combinations of the first d eigenvectors of the covariance matrix of $\mathbf{V}$ as summarized in Alg. 1. For massive, high-dimensional data, however, computing the covariance matrix may take a lot of time. Therefore, we propose to use instead Faloutsos and Lin's FastMap [1995], but please see next section.

The main point here is, triggered by the idea that the expected size of the convex hull of n Gaussian data points in the plane is $\Omega(\sqrt{\log n})$ [Hueter, 1999], CH-NMF extracts only approximately $p = j\sqrt{\log n}$ candidate points. This candidate set grows much slower than n. Given a candidate set of p convex hull data points $\mathbf{S} \in H_{cvx}(\mathbf{V})$, we now select those k convex hull data points that yield the best reconstruction of the remaining subset $\mathbf{S}$. This, again, can be formulated as a convex NMF optimization problem. We now have to minimize the following reconstruction error

$$J_S = \|\mathbf{S}^{m \times p} - \mathbf{S}^{m \times p}\mathbf{I}^{p \times k}\mathbf{J}^{k \times p}\|^2 \tag{5}$$

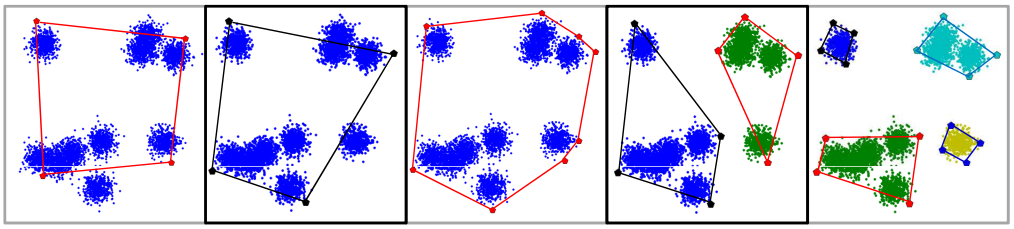
under the convexity constraints $\|\mathbf{I}_i\|_1 = 1, \mathbf{I}_i \geq \mathbf{0}$ and $\|\mathbf{J}_i\|_1 = 1, \mathbf{J}_i \geq \mathbf{0}$. Since $p \ll n$, solving (5) can be done efficiently using a quadratic programming solver. Note that the data dimensionality is m. The convex hull projection only served to determine a candidate set; all further computations are carried out in the original data space.

By obtaining a sufficient reconstruction accuracy for $\mathbf{S}$, we can set $\mathbf{X} = \mathbf{S}^{m \times p}\mathbf{I}^{p \times k}$ and thereby select k convex hull data points for solving Eq. (4). Typically, $\mathbf{I}$ results in unary representations. If this is not the case, we simply map $\mathbf{S}\mathbf{I}$ to their nearest neighboring data point in $\mathbf{S}$.

Given $\mathbf{X}$, the computation of the coefficients $\mathbf{H}$ is straight forward. For smaller data-sets it is possible to use the iterative update rule from Eq. (2). However, since we do not further modify the basis vectors $\mathbf{X}$, we can also find an optimal solution for each data point $\mathbf{v}_i$ individually $J_i = \|\mathbf{v}_i - \mathbf{X}\mathbf{h}_i\|^2$ using common solvers. Obviously, this can be parallelized.

3 Hierarchical CH-NMF

Modern datasets are not only massive, but often very complex. This makes it challenging to find useful information in the data. Fortunately, most datasets typically have a low intrinsic dimension. That is, the data lay on a smooth, structured low-dimensional manifold. As an illustrative (already low-dimensional) example consider Fig. 2. It depicts a typical "non-convex data" situation. We have drawn data points from 10 randomly positioned Gaussians in $2D$ and were interested in computing overcomplete representations, i.e., the number of basis vectors is greater than the dimensionality of the input. Overcomplete representations have been advocated because they have greater robustness in the presence of noise, can be sparser, and can have greater flexibility in matching structure in the data. By design, CH-NMF assigns clusters to the "extreme" Gaussians and not to "inner" Gaussians, cf. Figs. 2 **(a,c)**. Although we can still reconstruct each data point perfectly, this is discouraging. The intrinsic (low dimensional) structure of the data is not captured and, in turn, the representation of the data found is not as meaningful as it could be.



(a) CH-NMF bv=4 **(b)** HCH-NMF bv=4 **(c)** CH-NMF bv=8 **(d)** HCH-NMF bv=8 **(e)** HCH-NMF bv=16

Figure 2: Resulting basis vectors of CH-NMF **(a,c)** for 4 resp. 8 basis vectors (bv) and of HCH-NMF **(b,d,e)** for 4, 8, resp. 16 basis vectors. The data samples are drawn from 10 randomly placed Gaussian distributions in $2D$, 500 samples per Gaussian. For 4 basis vectors **(b)**, HCH-NMF essentially mimics CH-NMF. For more basis vectors **(d,e)**, however, it starts to adapt to the structure of the data: the basis vectors reside on the convex hulls of the Gaussians. CH-NMF's basis vectors **(a,c)**, in contrast, remain residing on the convex hull. By design, it considers the "extreme" Gaussians only and does not adapt to the structure of the data. (Best viewed in color.)

Algorithm 2: HCH-NMF: Hierarchical Convex-Hull NMF

Input: Data, i.e., the set of rows $\mathbf{v}_i$ of $\mathbf{V}^{m \times n}$; the pairwise distances $\mathbf{D}$; the minimal size *MinSize* of a leave
Output: A hierarchical decomposition of D represented as tree T

1 **if** $|\mathbf{V}| <$ *MinSize* **then**
2 **return** CH-NMF(*Leaf*) for l (even multiple of k) basis vectors
3 **else**
4 *Rule* $\leftarrow$CHOOSERULE($\mathbf{V}$);
5 $\mathbf{V}_s \leftarrow \{\mathbf{v}_i \in \mathbf{V} \mid \mathit{Rule}(\mathbf{v}_i) = \mathit{true}\}$;
6 $\mathbf{V}_f \leftarrow \{\mathbf{v}_i \in \mathbf{V} \mid \mathit{Rule}(\mathbf{v}_i) = \mathit{false}\}$;
7 *LeftTree* $\leftarrow$MAKETREE($\mathbf{V}_s$);
8 *RightTree* $\leftarrow$MAKETREE($\mathbf{V}_f$);
9 **return** (*Rule*, *LeftTree*, *RightTree*)

Hierarchical convex-hull NMF (HCH-NMF) is a convex NMF approach that automatically adapts to the low intrinsic dimensionality of data as illustrated in Figs. 2 **(b,d,e)**. The elegance of HCH-NMF stems from two facts:

- It naturally falls out of running CH-NMF using FastMap [Faloutsos and Lin, 1995] for efficiently computing and marking convex hull data points.
- In turn, it provably solves the convex NMF problem as it directly makes CH-NMF manifold-adaptive.

The latter point is difficult to prove for a two-steps approach: run any clustering approach, then run CH-NMF on the clusters. Also employing any of the existing large-scale manifold learning methods, see e.g. [Talwalkar *et al.*, 2008] is difficult. As Talwakar *et al.* [2008], argue they require a $\mathcal{O}(n^3)$ spectral decomposition of matrices where n is the number of samples. When the matrix is sparse, these techniques can be implemented relatively efficiently. However, when dealing with a large, dense matrix, the involved matrix products become expensive to compute.

As summarized in Alg. 2, HCH-NMF is based on a hierarchical decomposition of $\mathbb{R}^{\mathbb{D}}$ in form of a tree[1]. That is, it starts with the empty tree and repeatedly searches for the best test for a node according to some splitting criterion such as weighted variance along the FastMap dimension. Next, the examples $\mathbf{V}$ in the node are split into $\mathbf{V}_s$ (success) and $\mathbf{V}_f$ (failure) according to the test. For each split, the procedure is recursively applied, obtaining subtrees for the respective splits. We stop splitting if a minimum number *MinSize* of examples is reached or the variance in one node is small enough. In the leaves, we run CH-NMF on the examples falling into the leaves to find l basis vectors. Finally, we may run a post-processing step to find the best k basis vectors. In other words, HCH-NMF is conceptually easy, yet scalable to massive datasets and powerful as our experimental results will demonstrate.

Let us briefly review FastMap. FastMap computes a u-dimensional Euclidean embedding and proceeds as follows. Given pairwise distances among objects, in our case the rows of the data matrix $\mathbf{V}$, we select a pair of distant objects called *pivot objects*. Then, we draw a line between the pivot objects. Essentially, it serves as the first coordinate axis. For each object o, we determine the coordinate value $fm(o)$ along this axis by projecting o onto this line. Next, the pairwise distances of all objects are updated to reflect this projection, i.e., we compute the pairwise distances among the objects in the subspace orthogonal to the line. This process is repeated until, after u iterations, we get the u coordinates as well as the u-dimensional representation of all objects. Ostrouchov and Samatova [2005] have shown that the pivots are taken from the faces, usually vertices, of the convex hull of the data points in the original implicit Euclidean space. This justifies the idea to employ FastMap in step (3) "compute and mark convex hull data points" of CH-NMF as already mentioned in the last section. More important for HCH-NMF, it suggests a natural splitting criterion to produce a hierarchical decomposition: *Split the data according to the weighted variance along the 1D FastMap line.* More precisely, we compute the splitting rule as summarized in Alg. 3. Essentially, we run one iteration of FastMap and split along the "FastMap" line w.r.t. weighted variance. That is, we pick a pair of distant objects $\mathbf{x}$ and $\mathbf{y}$ (lines 1-3). Then, we project the data onto the line and compute for each data point its 1D coordinate value (lines 4-6). Now, we compute the split variable θ that minimizes the weighted variance (lines 8-12) and return the corresponding splitting rule (lines 13-14). Thus, for each splitting variable, determining the split point s can be done very quickly. In turn, by scanning all of the inputs, determining the best split is feasible and scales as $\mathcal{O}(n \log n)$.

How large should we grow the tree? Clearly, a very large tree might produce too many basis vectors. The basis vectors are likely to be not meaningful and the tree essentially "overfits" the data. A small tree, however, might not cap-

[1] In this paper, we follow Dasgupta and Freund's random projection trees RPTs) [Dasgupta and Freund, 2009a]. We do not split along a random direction but note that this could easily be achieved.

Algorithm 3: CHOOSERULE based on FastMap

Input: Data, i.e., the set of rows $\mathbf{v}_i$ of $\mathbf{V}^{m \times n}$; the minimal size of a leave *MinSize*
Output: A splitting rule *Rule* for the data

Pick any data point $\mathbf{t} \in \mathbf{V}$;
Let $\mathbf{x}$ be the farthest point from t in $\mathbf{V}$;
Let $\mathbf{y}$ be the farthest point from x in $\mathbf{V}$;
for $i = 1, 2, \ldots n$ **do**
    Project $\mathbf{v}_i$ onto the line spanned by $\mathbf{x}$ and $\mathbf{y}$;
    Let $fm(\mathbf{v}_i)$ be $\mathbf{v}_i$'s 1D coordinate value;
Sort the values $fm(\mathbf{v}_i)$ generating the list $s_1 \leq s_2 \leq \ldots \leq s_n$;
for $i = 1, 2, \ldots n$ **do**
    $\mu_1 = \frac{1}{i} \sum_{j=1}^{i} s_j$;
    $\mu_2 = \frac{1}{n-i} \sum_{j=i+1}^{n} s_j$;
    $c_i = \sum_{j=1}^{i} (s_j - \mu_1)^2 + \sum_{j=i+1}^{n} (s_j - \mu_2)^2$;
Find i that minimizes c_i and set $\theta = (s_i + s_{i+1})/2$;
$Rule(\mathbf{v}) := fm(\mathbf{v}) \leq \theta$;
return (*Rule*)

ture the important structure in the data at all. Reconsider standard CH-NMF. It corresponds to a tree of depth zero. In other words, the tree size is a tuning parameter governing HCH-NMF's complexity and one of its key advantages: *the diversified meaningfulness of its factorization*. The preferred strategy, as for example explained in [Hastie *et al.*, 2001], proceeds as follows. We grow a large tree T_0, stop the splitting process only when some minimum leaf size, say 200, is reached. Then, this tree is pruned in a post-processing step. We omit a detailed algorithmic description but rather briefly describe it now. The goal is to compute $k \geq l$ many basis vectors that reconstruct the data the best. We achieve this by successively merging neighboring leafs until we get k basis vectors. In each step, we find the two neighboring leafs that — if merged — produce the lowest reconstruction error over the covered examples. As this can be time consuming, we efficiently approximate it by always selecting the two neighboring leaves with highest resulting cluster accuracy when merging them.

In conclusion, HCH-NMF with a tree of depth 0 essentially coincides with CH-NMF. Moreover, the well known fact that if Z is any convex set that contains a convex hull $\mathbf{conv}(U)$ of a set U, in particular $Z = \mathbf{conv}(U \cup V)$, then $\mathbf{conv}(U) \subseteq Z$, see e.g. [Boyd and Vandenberghe, 2004], essentially proves the following correctness theorem.

Theorem 1 *HCH-NMF solves the convex NMF problem. It produces convex combinations of the data points in terms of basis vectors that minimize* (3) *but reside on convex hulls of clusters of data points.*

Due to the hierarchical decomposition of the data, HCH-NMF can adapt to the structure underlying the data as illustrated in Figs. 2 **(b,d,e)**. Indeed, it is akin to k-means. Because k-means clustering is a NP-hard optimization problem, see e.g. [Dasgupta and Freund, 2009b], this suggests that it is very unlikely that there exists an efficient algorithm for it.

4 Experiments

Our intention here is to investigate whether HCH-NMF can indeed find meaningful basis vectors in massive datasets and how it compares to CH-NMF. To this aim, we implemented both in Python using FastMap and the h5py HDF5 interface to deal with massive data. For optimization we used the cvxopt library by Dahl and Vandenberghe[2]. (H)CH-NMF offers many opportunities for parallelization.

[2] http://abel.ee.ucla.edu/cvxopt/

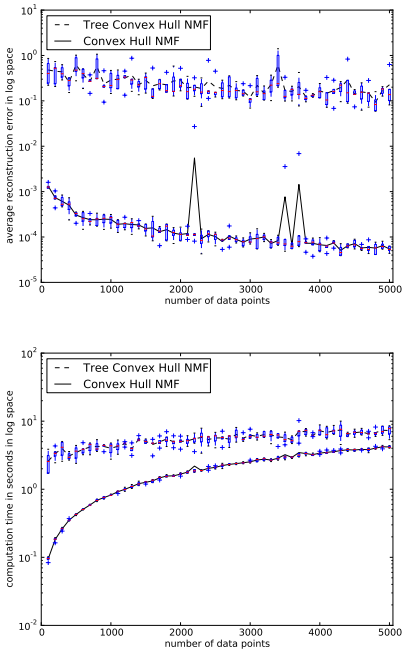


Figure 3: Boxplots of reconstruction errors **(top)** and computation times [sec.] **(bottom)** (both in log-space) of HCH-NMF and CH-NMF for varying numbers of synthetically generated data averaged over five reruns. As mentioned before, the Frobenius norm for CH-NMF has to be lower as CH-NMF approximates the convex hull of the complete data distribution and ignores the intrisic structure. Thus, for the purpose of interpretation and diversity this is probably not the right metric. (Best viewed in color.)

In the experiments, we only distributed the final reconstructions equally among all available cores. All experiments were ran on a standard Intel 3GHz computer with two cores. We report running times only for comparison of CH-NMF and HCH-NMF. Clearly, a C/C++ implementation would run several orders of magnitude faster.

We conducted four different experiments. To compare running time and reconstruction performance in a controlled setup, we compared (H)CH-NMF on synthetically generated data. Our main focus, however, are three additional experiments on massive real-world datasets, namely, publication histories of $760,000$ DBLP authors, 1.4 million activity profiles of guilds, and 4 million images of the Tiny image data-set [Torralba *et al.*, 2008]. For the sake of a better visualization, we show small trees.

Synthetic Data: Along the lines of [Ding *et al.*, 2009; Thurau *et al.*, 2009], we evaluated the mean reconstruction error and run-time performance using a varying number of data points sampled from three randomly positioned Gaussians in 2D. As already shown in [Thurau *et al.*, 2009], CH-NMF outperforms C-NMF for larger numbers of samples: it is several orders of magnitude faster while achieving competitive reconstruction errors. Therefore, we only compared HCH-NMF against CH-NMF. We varied the number of sampled data points ranging from 100 to 5000 in steps of 100. The maximum number of iterations for any numerical optimization was 100. The number of basis vectors was set to 12, i.e., we searched for overcomplete representations. The results averaged over 5 reruns are summarized in Fig. 3. As one can see, HCH-NMF produces an overhead

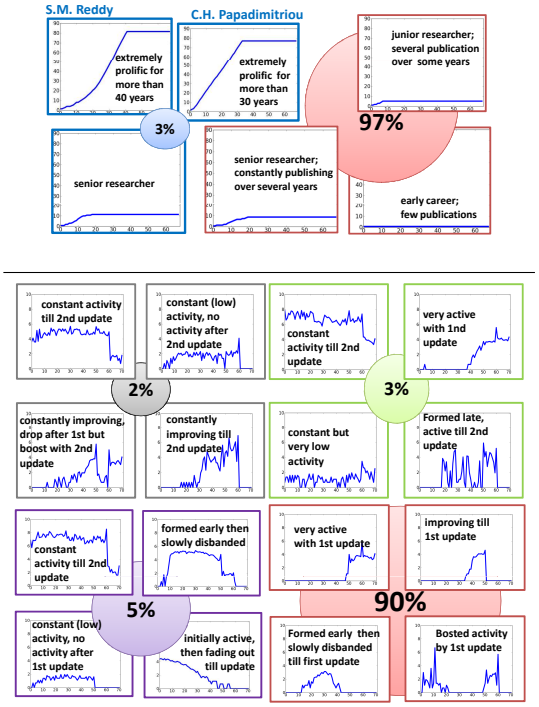


Figure 4: **(Top)** Clusters and corresponding basis vectors (histograms) found by HCH-NMF on the DBLP dataset. By describing the basis vectors, we gain an intuitively understandable description of academic careers: 97% of all authors are best described by the typical phases of an academic career. Indeed, there are renown, extremely prolific exceptions. Because the basis vectors are actual data points, we can identify them as Papadimitriou and Reddy. **(Bottom)** Clusters and corresponding basis vectors found by HCH-NMF on the World of Warcraft® dataset. (Best viewed in color.)

in running time for smaller sample sizes. For larger sample sizes, HCH-NMFs catches up. The reconstruction error is slightly higher than for CH-NMF but still lower than for k-means (as reported in [Thurau *et al.*, 2009]). The higher reconstruction error is due to the natur of convex hulls: any point within the convex hull can perfectly reconstructed. Hence, CH-NMF is a lower bound of HCH-NMF in terms of reconstruction error.

Bibliographic Analysis based on DBLP: Bibliographic databases such as DBLP[3] are a rich source of information. Here, we are interested in the question whether there are common patterns in the development of academic careers. To this aim, we extracted from DBLP the cumulative publication histograms of $757,368$ authors, cf. Fig. 4**(Top)**. A publication histogram consists of the number of publications listed in DBLP in her first year, second year, and so on. We have cumulated the publications numbers of the years. The longest histogram we found spanned 68 years. To get equal length curves we filled missing years with 0. Following [Aitchison, 1982], we use logarithmic histogram values in our analysis. The idea is that the publication histogram of an author is a good descriptor for her activity and also to some extend for her success (but of course has not to imply impact/quality). A senior researcher, for example, is likely to have contributed over several years but there are exceptionally prolific authors. PhD students, on the other hand, may not have published many papers. We expected HCH-NMF to discover these variations. This was indeed the case as shown in Fig. 4 **(Top)**. The patterns found can be summarized as "*the majority of authors fall into one of the phases of a regular academic career (student, junior, senior) but of course there are illustrious exceptions.* It took 1 hour to compute the model, i.e., growing the tree and pruning it. CH-NMF took 45 minutes essentially yielding the union of all shown basis vectors, hence, giving the impression "*there is a Papadimitriou in all of us*".

Social Network World of Warcraft®: This dataset consists of recordings of the online appearance of characters in the computer game World of Warcraft®. It is assumed that World of Warcraft® has about 12 million p(l)aying customers. The game takes place in a virtual medieval fantasy environment. World of Warcraft® is often considered one large social platform which is used for chatting, team-play, and gathering. Compared to well known virtual worlds that mainly serve as chat platforms such as Second-life ®[4], World of Warcraft® is probably the *real* second life as it has a larger and more active (paying) user base. Moreover, a whole industry is developing around World of Warcraft®. It is estimated that 400.000 people world-wide are employed as gold-farmers, i.e. collecting virtual goods for online games and selling them over the Internet.

Players organize in groups, which are called guilds. Unlike groups known from other social platforms, such as Flickr, membership in a guild is exclusive. Obviously, the selection of a guild influences with whom players frequently interact. It also influences how successful players are in terms of game achievements. We assume that the level distribution among a guild is a good descriptor for its success and activity. For example, a guild of very experienced level 80 characters has a higher chance for achievements than a guild of level 10 players. Also, a level histogram gives an indicator for player activity over time. If players are continuously staying with a particular guild, we expect an equally distributed level histogram, as the characters are continuously increasing their level over time. The data was crawled from the publicly accessible site `www.warcraftrealms.com`. We viewed each character online appearance as a vote. Characters observations span a period of 4 years. Every time a character is seen online, he votes for the guild he is a member of according to his level. We accumulate the votes into a level-guild histogram, going from level 10 (level 1-9 are excluded) to level 80 (the highest possible level). Players advance in level by engaging in the game, i.e. completing quests or other heroic deeds. Following [Aitchison, 1982], we use logarithmic histogram values in our analysis. In total, we collected 150 million votes of 18 million characters belonging to 1.4 million guilds.

Running HCH-NMF took about 2.5 hours and revealed some very interesting patterns as shown in Fig. 4 **(Bottom)**. As for CH-NMF, see [Thurau *et al.*, 2009], we can also spot singular events, in our case large *updates* to the game content (this is a regular procedure that makes novel content available and also allows a further advancement in character level). Apparently, large updates to the game can result in a restructuring of social groups. More interestingly, HCH-NMF allows us to descriptions of clusters. For instance, 90% of the data can be described in terms of just

[3] `http://www.informatik.uni-trier.de/~ley/db/`

[4] `http://secondlife.com/`

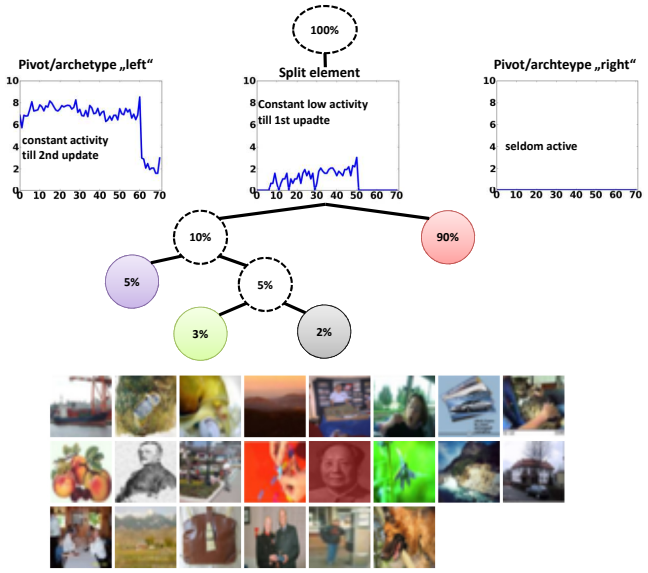


Figure 5: **(Top)** HCH-NMF tree as well as the pivot and split elements for the first split on the World of Warcraft® dataset. The "90%" cluster in Fig. 4 captures the data "between" the split element and the "right" pivot element. Thus, the majority of guilds is actually close to the "low constant activity" or "seldom activity" guilds. Both patterns are well captured by combining the corresponding four basis vectors in Fig. 4. **(Bottom)** HCH-NMF's split elements for the 4 million tiny images are natural images. (Best viewed in color.)

four basis vectors: "*formed early then slowly disbanded till 1st update*", "*improving till 1st update*", "*very active with 1st update*", and "*Boosted activity with 1st update*". That was surprising. We knew that most guilds are rather *seldom active*, see [Thurau *et al.*, 2009]. Therefore, we "zoomed" into the model, a feature of HCH-NMF not supported by CH-NMF. Specifically, we had a look at the pivot and split elements of the first level of the induced tree, see Fig. 5 **(Top)**. This revealed that most of the 90% are actually "*seldom active*": they lay between the "constantly active till 2nd update" and "seldom active" guild on the FastMap line. The four basis vectors of the 90% cluster together are archetypical guilds to reconstruct this pattern. Running CH-NMF took about 2 hours and produced essentially the same basis vectors. The major difference was that it used the "*seldom active*" guild directly and did not factorize it.

Massive Image Collection: Our final experiment applies HCH-NMF to a subset of 4 million images of Torralba *et al.*'s 80 million tiny images [Torralba *et al.*, 2008]. The images are represented as 384 dimensional GIST feature vector. The result of running HCH-NMF is shown in Fig. 6. As already reported in [Thurau *et al.*, 2009], some of the basis vectors discovered show a geometric similarity to Walsh filters that are found among the principal components of natural images [Heidemann, 2006]. This suggests that the extremal points in this large collection of natural images are located close to the principal axes of the data. On the other hand, the split elements as shown in Fig. 5 **(Bottom,)** are mostly "realistic" images. This suggests that they are located in the center of the data/clusters. In contrast to CH-NMF, HCH-NMF grouped the basis vectors into meaningful clusters. From left to right, the basis vectors, i.e., the columns, capture different aspects of the images such as dark-light-dark, horizontal-cross/circle-vertical, complex-plain-complex. Overall, running HCH-NMF took only 23 hours and 37 minutes. This is remarkable as we used an external USB hard disk and the running time can be considerably reduced using an internal hard disk and implementing HCH-NMF fully in C/C++.

5 Conclusion

We have introduced hierarchical convex-hull NMF. It seeks to leverage convex NMF by automatically adapting to the geometric structure of the data. It is fast and straightforward to implement, provably solves the convex NMF problem, combines the interpretability of both convex NMF as well as hierarchical decomposition methods, and scales well to massive, high-dimensional datasets. These contributions advance the understanding of descriptive analytics of massive, high-dimensional datasets and is an encouraging sign that applying NMF techniques in the wild, i.e., on hundreds of millions of data points may not be insurmountable.

There are several interesting avenues for future work. One is the application of HCH-NMF to other challenging datasets, such as Wikipedia, Netflix, Facebook, or the blogsphere, and to use it for applications such as collaborative filtering. For the latter case, it is interesting to develop bottom up HCH-NMF variants and to investigate the missing values case. Another important avenue is parallelization. HCH-NMF suggests a natural data-driven parallelization: the training set is partitioned into subsets associated with separate processors. Finally, HCH-NMF is highly relevant for high-dimensional classification problems. Here, it is infeasible to include enough training samples to cover the class regions densely. As Cevikalp *et al.* [2008] have recently pointed out, irregularities in the resulting sparse sample distributions cause local classifiers such as nearest neighbors and kernel methods to have irregular decision boundaries. One solution is to "fill in the holes" by building a convex model of the regions spanned by the training samples. Using HCH-NMF, we even take the geometric structure of each class into account.

Acknowledgment The authors would like to thank A. Torralba, R. Fergus, and W. T. Freeman for making the tiny images freely available and S. Zayakh for preparing the DBLP dataset. M. Wahabzada and K. Kersting were supported by the Fraunhofer ATTRACT Fellowship STREAM.

References

[Aitchison, 1982] J. Aitchison. The Statistical Analysis of Compositional Data. *J. of the Royal Statistical Society B*, 44(2):139–177, 1982.

[Boyd and Vandenberghe, 2004] S. Boyd and L. Vandenberghe. *Convex Optimization*. Camb. Univ. Press, 2004.

[Cai *et al.*, 2008] Deng Cai, Xiaofei He, Xiaoyun Wu, and Jiawei Han. Non-negative matrix factorization on manifold. In *International Conference on Data Mining*, pages 63–72. IEEE, 2008.

[Cevikalp *et al.*, 2008] H. Cevikalp, B. Triggs, and R. Polikar. Nearest hyperdisk methods for high-dimensional classification. In *Proceedings of the 25th International Conference on Machine Learning (ICML-08)*, 2008.

Figure 6: HCH-NMF clusters (columns) and corresponding basis vectors (images in the columns) of the 4 million tiny images. From left to right, the basis vectors capture different aspects of the images such as dark-light-dark, horizontal-cross/circle-vertical, complex-plain-complex. (Best viewed in color.)

[Cutler and Breiman, 1994] A. Cutler and L. Breiman. Archetypal Analysis. *Technometrics*, 36(4):338–347, 1994.

[Dasgupta and Freund, 2009a] S. Dasgupta and Y. Freund. Random projection trees and low dimensional manifolds. In R.E. Ladner and C. Dwork, editors, *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC-08)*, pages 537–546, 2009.

[Dasgupta and Freund, 2009b] S. Dasgupta and Y. Freund. Random projection trees for vector quantization. *IEEE Transactions on Information Theory*, 55:3229–3242, 2009.

[de Berg *et al.*, 2000] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry*. Springer, 2000.

[Ding *et al.*, 2009] C.H.Q. Ding, T. Li, and M.I. Jordan. Convex and Semi-Nonnegative Matrix Factorizations. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2009. Accepted for publication.

[Donoho and Tanner, 2005] D.L. Donoho and J. Tanner. Neighborliness of Randomly-Projected Simplices in High Dimensions. *Proc. of the Nat. Academy of Sciences*, 102(27):9452–9457, 2005.

[Faloutsos and Lin, 1995] C. Faloutsos and K.-I. Lin. FastMap: A Fast Algorithm for Indexing, Data-mining and Visualization of Traditional and Multimedia Datasets. In *Proc. ACM SIGMOD*, 1995.

[Golub and van Loan, 1996] G.H. Golub and J.F. van Loan. *Matrix Computations*. Johns Hopkins University Press, 3rd edition, 1996.

[Halevy *et al.*, 2009] A.Y. Halevy, P. Norvig, and F. Pereira. The unreasonable effectiveness of data. *IEEE Intelligent Systems*, 24:8–12, 2009.

[Hall *et al.*, 2005] P. Hall, J. Marron, and A. Neeman. Geometric Representation of High Dimension Low Sample Size Data. *J. of the Royal Statistical Society B*, 67(3):427–444, 2005.

[Hastie *et al.*, 2001] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2001.

[Heidemann, 2006] G. Heidemann. The principal components of natural images revisited. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(5), 2006.

[Hueter, 1999] I. Hueter. Limit Theorems for the Convex Hull of Random Points in Higher Dimensions. *Trans. of the American Mathematical Society*, 351(11):4337–4363, 1999.

[Jolliffe, 1986] I.T. Jolliffe. *Principal Component Analysis*. Springer, 1986.

[Kim and Park, 2008] Jingu Kim and Haesun Park. Toward faster nonnegative matrix factorization: A new algorithm and comparisons. In *International Conference on Data Mining*, pages 353–362. IEEE, 2008.

[Kolda and Sun, 2008] Tamara G. Kolda and Jimeng Sun. Scalable tensor decompositions for multi-aspect data mining. In *International Conference on Data Mining*, pages 363–372. IEEE, 2008.

[Lee and Seung, 1999] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–799, 1999.

[Mairal *et al.*, 2010] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *JMLR*, 11:19–60, 2010.

[Ostrouchov and Samatova, 2005] G. Ostrouchov and N.F. Samatova. On FastMap and the convex hull of multivariate data: toward fast and robust dimension reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1340–1434, 2005.

[Suvrit, 2008] Sra Suvrit. Block-iterative algorithms for non-negative matrix approximation. In *ICDM*, pages 1037–1042. IEEE, 2008.

[Talwalkar *et al.*, 2008] Ameet Talwalkar, Sanjiv Kumar, and Henry Rowley. Large-scale manifold learning. In *Computer Vision and Pattern Recognition*. IEEE, 2008.

[Thurau *et al.*, 2009] C. Thurau, K. Kersting, and C. Bauckhage. Convex non-negative matrix factorization in the wild. In H. Kargupta and W. Wang, editors, *Proceedings of the 9th IEEE International Conference on Data Mining (ICDM-09)*, 2009.

[Torralba *et al.*, 2008] Antonio Torralba, Rob Fergus, and William T. Freeman. 80 Million Tiny Images: A Large Data Set for Nonparametric Object and Scene Recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 30(11):1958–1970, 2008.

[Ziegler, 1995] G.M. Ziegler. *Lectures on Polytopes*. Springer, 1995.