Frame-based Data Factorizations

Sebastian Mair [1] **Ahcène Boubekki** [1 2] **Ulf Brefeld** [1]

Abstract

Archetypal Analysis is the method of choice to compute interpretable matrix factorizations. Every data point is represented as a convex combination of factors, i.e., points on the boundary of the convex hull of the data. This renders computation inefficient. In this paper, we show that the set of vertices of a convex hull, the so-called *frame*, can be efficiently computed by a quadratic program. We provide theoretical and empirical results for our proposed approach and make use of the frame to accelerate Archetypal Analysis. The novel method yields similar reconstruction errors as baseline competitors but is much faster to compute.

1. Introduction

Matrix factorizations are used in many different learning tasks, including clustering, classification, recommendation, text and social network analysis, image denoising and hyperspectral imaging; consequently, a great deal of approaches to factor a matrix into two or more matrices have been proposed.

A common limitation of matrix factorization techniques, however, is a lack of interpretability of the resulting factors. This observation also holds for non-negative matrix factorization (Paatero & Tapper, 1994; Lee & Seung, 1999) (NMF). Although NMF constrains data, matrices, and factors to be non-negative, the remaining factors may lie far from other data points and do not necessarily contribute to interpretability. Recently, the focus on interpretable factors led to stochastic NMFs for clusterings (Arora et al., 2011; 2013) and convex constraints (Ding et al., 2010).

However, computing interpretable factors is actually an "old" problem that has been introduced by Cutler and Breiman (1994). Their idea is to represent every data point as a convex combination of factors, the so-called archetypes. Since the archetypes are themselves restricted to be convex combinations of data points, they need to lie on the boundary of the convex hull of the data; hence, they are either close to actual data points or mixtures thereof, see Figure 1. An archetype-based factorization yields two stochastic matrices and naturally interpretable factors for arbitrary data matrices. Variants of Archtypal Analysis (AA) are for instance introduced in Mørup and Hansen (2010), Chen et al. (2014) and Bauckhage et al. (2015).

The computation of convex hulls is generally demanding. For AA, a straight forward idea is to make use of (e.g., precomputed) vertices of the convex hull of the data. We will refer to the set of vertices as the *frame*. Given the frame, computing the archetypes reduces to a search within the frame. We propose to go one step further and approximate AA by restricting the whole optimization to the frame: firstly, we compute the frame and secondly, we compute the archetypes on the frame (and just on the frame). Naturally, the result is a factorization of the frame itself. The factorization can then be extended to the all data points by recomputing the weights in hindsight. The idea is similar to Thurau et al. (2011) who propose to approximate the frame using two-dimensional projections; unfortunately, their approach adds an unnecessary layer of approximation to the problem: Ideally, given an appropriate approximation of the frame, the bulk of the remaining data points should be reconstructed either lossless or at only a small cost.

Standard approaches to compute the convex hull like Quickhull (Barber et al., 1996) become infeasible for dimensionalities larger than three because of dispensable triangulations. Discarding the triangulations leads to linear programming (LP)-based solutions (Dulá & Helgason, 1996; Ottmann et al., 2001; Dulá & López, 2012) which test whether a point at-hand is part of the frame. These approaches require adequate preprocessing as duplicates may cause false negatives.

In this paper, we (i) show that the exact frame can be computed by a quadratic program (QP), (ii) reduce the optimization to an existing algorithm and (iii) provide theoretical and empirical justifications for the developed method.

[1]Leuphana University, Lüneburg, Germany [2]German Institute for Educational Research, Frankfurt am Main, Germany. Correspondence to: Sebastian Mair <mair@leuphana.de>.

Proceedings of the 34^{th} International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017.

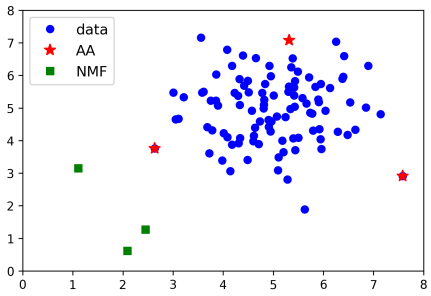

Figure 1. Comparison of factorizations computed with NMF (green squares) and AA (red stars). While the former may place the factors far from data, Archetypal Analysis uses data points on the boundary of the convex hull and yields an intuitive solution.

The remainder is structured as follows. Section 2 reviews Archetypal Analysis and Section 3 contains the main contribution, a new computational method for finding the frame and a frame-based matrix factorization. Section 4 reports on empirical results and Section 5 introduces an application as an autoencoder. Section 6 concludes.

2. Archetypal Analysis

Let $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}_{i=1}^n$ be a data set consisting of n data points in d dimensions and $\mathbf{X} \in \mathbb{R}^{n \times d}$ be the data matrix. The goal of Archetypal Analysis (AA) (Cutler & Breiman, 1994) is to find a factorization of the data

$$\mathbf{X} = \mathbf{ABX} = \mathbf{AZ}, \tag{1}$$

where $\mathbf{A} \in \mathbb{R}^{n \times p}$, $\mathbf{Z} = \mathbf{BX} \in \mathbb{R}^{p \times d}$ are the factors and $p \in \mathbb{N}$ is the latent dimensionality. The matrix $\mathbf{A}$ contains the weights $\mathbf{a}_i \in \mathbb{R}^p$ $(i = 1, \dots, n)$ of the convex combinations of the p archetypes $\mathbf{z}_1, \dots, \mathbf{z}_p$ which are stacked in the matrix $\mathbf{Z}$, i.e.,

$$\forall i: \quad \mathbf{x}_i = \sum_{k=1}^{p} (\mathbf{a}_i)_k \cdot \mathbf{z}_k, \quad \sum_{k=1}^{p} (\mathbf{a}_i)_k = 1, \quad (\mathbf{a}_i)_k \geq 0.$$

The archetypes $\mathbf{z}_k$ $(k = 1, \dots, p)$ are themselves convex combinations of data points. The matrix $\mathbf{B}$ holds the weights $\mathbf{b}_k \in \mathbb{R}^n$ $(k = 1, \dots, p)$ of those convex combinations and selects the archetypes from the given data set $\mathbf{X}$. The representation is as follows:

$$\forall k: \quad \mathbf{z}_k = \sum_{i=1}^{n} (\mathbf{b}_k)_i \cdot \mathbf{x}_i, \quad \sum_{i=1}^{n} (\mathbf{b}_k)_i = 1, \quad (\mathbf{b}_k)_i \geq 0.$$

The matrices $\mathbf{A}$ and $\mathbf{B}$ are row-stochastic due to their constraints. The factorization is obtained by minimizing the residual sum of squares (RSS)

$$\min \ \mathrm{RSS}(p) = \|\mathbf{X} - \mathbf{AZ}\|_F^2 = \|\mathbf{X} - \mathbf{ABX}\|_F^2,$$

Algorithm 1 Archetypal Analysis (AA)

Input: data matrix $\mathbf{X}$, number of archetypes p
Output: factor matrices $\mathbf{A}, \mathbf{Z} = \mathbf{BX}$ $(\mathbf{ABX} \approx \mathbf{X})$
$\mathbf{Z}$ = initial guess of archetypes on $\mathbf{X}$
while not converged **do**
  for $i = 1, 2, \dots, n$ **do**
    $\mathbf{a}_i = \underset{\|\mathbf{a}_i\|_1 = 1, a_{ij} \geq 0}{\operatorname{argmin}} \|\mathbf{Z}^\top \mathbf{a}_i - \mathbf{x}_i\|_2^2$
  end for
  $\mathbf{Z} = (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{X}$
  for $k = 1, 2, \dots, p$ **do**
    $\mathbf{b}_k = \underset{\|\mathbf{b}_k\|_1 = 1, b_{kj} \geq 0}{\operatorname{argmin}} \|\mathbf{X}^\top \mathbf{b}_k - \mathbf{z}_k\|_2^2$
  end for
  $\mathbf{Z} = \mathbf{BX}$
end while

where $\|\cdot\|_F$ denotes the Frobenius norm. The optimization problem is non-convex in $\mathbf{A}$ and $\mathbf{B}$ but convex in $\mathbf{A}$ for a fixed $\mathbf{B}$ and vice versa. Usually, it is solved by iteratively solving for $\mathbf{A}$ and $\mathbf{B}$ as outlined in Algorithm 1. Cutler and Breiman (1994) prove that the archetypes $\mathbf{z}_1, \dots, \mathbf{z}_p$ lie on the boundary $\partial \operatorname{conv}(\mathcal{X})$ of the convex hull of the data $\mathcal{X}$ for $1 < p < n$. From a geometrical point of view, AA yields an approximation of the convex hull of size p. Every point inside this polygon can be reconstructed in a lossless way and every point outside will be projected onto it. The optimization of the RSS will minimize the reconstruction error.

3. Frame-based Factorizations

3.1. Preliminaries

We consider a discrete data set $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}_{i=1}^n$ consisting of n data points in d dimensions, i.e., $\mathbf{x}_i \in \mathbb{R}^d$ for $i = 1, 2, \dots, n$. The convex hull $\operatorname{conv}(\mathcal{X})$ is the intersection of all convex sets containing $\mathcal{X}$. Furthermore, $\operatorname{conv}(\mathcal{X})$ is the set of all convex combinations of points in $\mathcal{X}$. The intersection of $\mathcal{X}$ with the boundary of $\mathcal{X}$, i.e., $\mathcal{X} \cap \partial \operatorname{conv}(\mathcal{X})$, is called the *frame* $\mathcal{F}$. Hence, the frame consists of the extreme points of $\mathcal{X}$. Those points cannot be represented as convex combinations of other points rather than themselves. Note that the frame $\mathcal{F}$ and the data set $\mathcal{X}$ yield the same convex hull, i.e., $\operatorname{conv}(\mathcal{F}) = \operatorname{conv}(\mathcal{X})$. By $q = |\mathcal{F}|$ we refer to the size of the frame and we call the portion of points in $\mathcal{X}$ belonging to the frame $\mathcal{F}$, the *frame density*. In the remainder we assume $n > d$.

There are some straightforward consequences that will become handy in the remainder. When the inner product between two points is maximized, one of the points is an extreme point and thus belongs to the frame of the data.

Algorithm 2 Frame-AA

Input: data $\mathcal{X}$, number of archetypes $p \in \mathbb{N}$
Output: factor matrices $\mathbf{A}, \mathbf{Z} = \mathbf{BH}$ ($\mathbf{ABH} \approx \mathbf{X}$)
$\mathcal{F} = \text{Frame}(\mathcal{X})$
$\mathbf{H} =$ frame matrix
$\mathbf{A}, \mathbf{Z} = \text{ArchetypalAnalysis}(\mathbf{H}, p)$
$\mathbf{A} = \mathbf{0} \in \mathbb{R}^{n \times p}$
for $i = 1, 2, \ldots, n$ **do**
  $\mathbf{a}_i = \underset{\|\mathbf{a}_i\|_1 = 1, a_{ij} \geq 0}{\text{argmin}} \|\mathbf{Z}^\top \mathbf{a}_i - \mathbf{x}_i\|_2^2$
end for

Lemma 1. *Let $\mathcal{X}$ be a finite set of discrete points, then*

$$\forall \mathbf{x} \in \mathcal{X}: \quad \underset{\mathbf{x}' \in \mathcal{X}}{\text{argmax}} \langle \mathbf{x}, \mathbf{x}' \rangle \in \mathcal{F}.$$

Proof. Linearity and convexity of the inner product imply that its maximum is realized by an extreme point of the domain. Since the domain is $\mathcal{X}$, the maximum belongs to its frame $\mathcal{F}$. □

In addition, every point of the domain lies in the span of some points on the frame.

Proposition 1. *Every point $\mathbf{x}$ of $\mathcal{X}$ can be written as a convex combination of at most $d+1$ points from the frame $\mathcal{F}$ of $\mathcal{X}$.*

Proof. See Brondsted (2012). □

3.2. Motivation

The idea of the proposal is as follows. Due to the fact that the archetypes lie on the boundary of the convex hull, it is possible to restrict the search of the archetypes to the frame. However, we intend to go one step further. Since the frame $\mathcal{F}$ and the data $\mathcal{X}$ yield the same convex hull, we restrict the factorization in Equation (1) to the matrix $\mathbf{H} \in \mathbb{R}^{q \times d}$ of stacked points of the frame, i.e.,

$$\mathbf{H} = \mathbf{ABH} = \mathbf{AZ}.$$

The idea is depicted in Figure 2. Although Archetypal Analysis is only computed on the frame it often yields almost identical archetypes as AA computed on the whole data set as illustrated in the right subplot of Figure 2. The reduction of points yields an accelerated computation.

Assuming a low frame density, i.e., $q \ll n$, a sufficient approximation of the frame, and therefore the convex hull, will also cover most of the points in the interior of its induced polygon. On the other hand, if the frame density is high, the problem will reduce to standard AA in the limit $q \to n$ and the speed up will vanish. Based on the nature of the problem, we claim that AA makes no sense in scenarios of high frame densities. This is because almost all points will be projected unless p is chosen very high which is usually not the case.

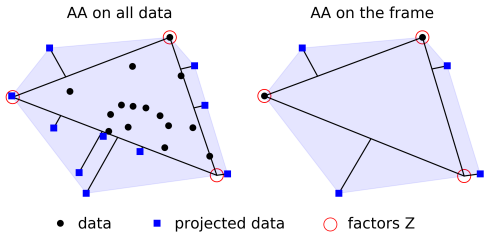

Figure 2. Approximation of Archetypal Analysis.

After computing the factorization by AA, we have to recompute the weight matrix $\mathbf{A} \in \mathbb{R}^{q \times p}$ for all data points using the optimized archetypes within $\mathbf{Z}$. This procedure is called *Frame-AA* and presented in Algorithm 2. Note that the idea does not only apply for standard Archetypal Analysis as presented in Cutler and Breiman (1994) but also all variants thereof (Mørup & Hansen, 2010; Chen et al., 2014; Bauckhage et al., 2015).

Until now, we assumed that the frame $\mathcal{F}$ or equivalently the frame matrix $\mathbf{H} \in \mathbb{R}^{q \times d}$, as required in the first line of the algorithm, is already given. In the following section, we present a novel algorithm for efficiently computing the frame of a discrete data set.

3.3. Representing a Single Point

Let $\mathcal{F}$ be the frame of $\mathcal{X}$ and $\mathbf{X} \in \mathbb{R}^{d \times n}$ be the design matrix of $\mathcal{X}$. The idea is as follows. For every $\mathbf{x} \in \mathcal{X}$, we aim to solve the linear system $\mathbf{Xs} = \mathbf{x}$ subject to the constraints that the solution $\mathbf{s}$ is non-negative, sums up to one, and uses only points from the frame, i.e.,

$$\begin{aligned} &\mathbf{Xs} = \mathbf{x} \\ \text{s.t.} \quad &\mathbf{s}_i \geq 0 \ \wedge \ \mathbf{1}^\top \mathbf{s} = 1 \ \wedge \ \mathbf{s}_i \neq 0 \Rightarrow \mathbf{x}_i \in \mathcal{F}. \end{aligned} \tag{2}$$

Theorem 1 reduces this to a least-squares approach.

Theorem 1. *Let $\mathcal{F}$ be the frame of $\mathcal{X}$, $\mathbf{X} \in \mathbb{R}^{d \times n}$ be the design matrix of $\mathcal{X}$, $\mathbf{x} \in \mathcal{X}$ and $\bar{\mathbf{X}} = \left[\mathbf{X}^\top, \mathbf{1}\right]^\top \in \mathbb{R}^{(d+1) \times n}$ as well as $\bar{\mathbf{x}} = (\mathbf{x}^\top, 1)^\top \in \mathbb{R}^{d+1}$ be the augmented versions of $\mathbf{X}$ and $\mathbf{x}$. Then, the following problems have the same solutions.*

(i) $\mathbf{Xs} = \mathbf{x}$ s.t. $\mathbf{s}_i \geq 0 \wedge \mathbf{1}^\top \mathbf{s} = 1 \wedge \mathbf{s}_i \neq 0 \Rightarrow \mathbf{x}_i \in \mathcal{F}$.

(ii) $\bar{\mathbf{X}}\mathbf{s} = \bar{\mathbf{x}}$ s.t. $\mathbf{s}_i \geq 0 \ \wedge \ \mathbf{s}_i \neq 0 \Rightarrow \mathbf{x}_i \in \mathcal{F}$.

(iii) $\text{argmin}_{\mathbf{s} \geq 0} \frac{1}{2} \|\bar{\mathbf{X}}\mathbf{s} - \bar{\mathbf{x}}\|_2^2$ s.t. $\mathbf{s}_i \neq 0 \Rightarrow \mathbf{x}_i \in \mathcal{F}$.

Proof. (i) is equivalent to (ii) as it integrates the constraint $\mathbf{1}^\top \mathbf{s} = 1$ into the system of linear equations. Proposition 1

Algorithm 3 Lawson and Hanson's active set algorithm

Input: design matrix $\bar{\mathbf{X}}$ and right hand side $\bar{\mathbf{x}}$
Output: solution $\mathbf{s}$ with $\mathbf{s}_i \geq 0 \ \ \forall i$
$\mathcal{P} = \emptyset$
$\mathcal{Z} = \{1, 2, \ldots, n\}$
$\mathbf{s} = \mathbf{0}$
$\mathbf{w} = -\nabla f(\mathbf{s}) = \bar{\mathbf{X}}^\top(\bar{\mathbf{x}} - \bar{\mathbf{X}}\mathbf{s})$
while $\mathcal{Z} \neq \emptyset$ and $\exists\, \mathbf{w}[\mathcal{Z}] \geq \varepsilon$ **do**
  $k = \text{argmax}_j\{\, \mathbf{w}_j \mid j \in \mathcal{Z} \,\}$
  $\mathcal{Z} = \mathcal{Z} \setminus \{k\}$
  $\mathcal{P} = \mathcal{P} \cup \{k\}$
  $\bar{\mathbf{X}}_\mathcal{P} = \bar{\mathbf{X}}[:, \mathcal{P}] \in \mathbb{R}^{(d+1)\times|\mathcal{P}|}$
  $\mathbf{z} = \text{argmin}_\mathbf{z} \|\bar{\mathbf{X}}_\mathcal{P}\mathbf{z} - \bar{\mathbf{x}}\|_2^2$
  $\mathbf{z}[\mathcal{Z}] = 0$
  $\mathcal{J} = \{\, j \in \mathcal{P} \mid \mathbf{z}_j \leq \varepsilon \,\}$
  while $|\mathcal{J}| > 0$ **do**
    $\alpha = \min\{\, \frac{s_j}{s_j - z_j} \mid j \in \mathcal{J} \,\}$
    $\mathbf{s} = \mathbf{s} + \alpha \cdot (\mathbf{z} - \mathbf{s})$
    for $i = 1, 2, \ldots, n$ **do**
      if $i \in \mathcal{P}$ and $|s_i| \leq \varepsilon$ **then**
        $\mathcal{P} = \mathcal{P} \setminus \{i\}$
        $\mathcal{Z} = \mathcal{Z} \cup \{i\}$
        $\bar{\mathbf{X}}_\mathcal{P}[:, i] = \mathbf{0}$
      end if
    end for
    $\mathbf{z} = \text{argmin}_\mathbf{z} \|\bar{\mathbf{X}}_\mathcal{P}\mathbf{z} - \bar{\mathbf{x}}\|_2^2$
    $\mathbf{z}[\mathcal{Z}] = 0$
  end while
  $\mathbf{x} = \mathbf{z}$
  $\mathbf{w} = -\nabla f(\mathbf{s}) = \bar{\mathbf{X}}^\top(\bar{\mathbf{x}} - \bar{\mathbf{X}}\mathbf{s})$
end while

assures that (i) has a solution. Hence, the minimum of $\frac{1}{2}\|\bar{\mathbf{X}}\mathbf{s} - \bar{\mathbf{x}}\|_2^2$ is always zero and a solution of (iii) is also a solution to (ii) and (i). □

If the condition that the positive positions of $\mathbf{s}$ refer to points on the frame is dropped, then problem

$$\mathbf{s} = \text{argmin}_{\mathbf{s}\geq 0}\, f(\mathbf{s}) = \frac{1}{2}\|\bar{\mathbf{X}}\mathbf{s} - \bar{\mathbf{x}}\|_2^2 \tag{3}$$

is equivalent to the non-negative least squares (NNLS) problem which is a special case of a QP. The active-set method from Lawson and Hanson (1995) is proven to yield a least-squares estimate of Equation (3). As the minimum is zero, it also gives a solution to the linear problem up to the condition that the points contributing to the solution belong to the frame. The method of Lawson and Hanson (1995) called NNLS is outlined in Algorithm 3.

However, until now it remains unclear whether the positive elements of the solution refer to points on the frame. The following theorem shows that this is actually the case.

Theorem 2. *Algorithm 3 from Lawson and Hanson (1995) solves the problem in Equation (2).*

Proof. Let Algorithm 3 solve the problem in Equation (3). Denote by $\mathbf{w} = -\nabla f(\mathbf{s}) = \bar{\mathbf{X}}^\top(\bar{\mathbf{x}} - \bar{\mathbf{X}}\mathbf{s})$ the negative gradient with respect to $\mathbf{s}$. The points $\{\, \bar{\mathbf{x}}_i \mid i \in \mathcal{P} \,\}$ involved in the solution are selected via

$$\begin{aligned} k &= \text{argmax}_j\{\, \mathbf{w}_j \mid j \in \mathcal{Z} \,\} \\ &= \text{argmax}_j\{\, \left[\bar{\mathbf{X}}^\top(\bar{\mathbf{x}} - \bar{\mathbf{X}}\mathbf{s})\right]_j \mid j \in \mathcal{Z} \,\} \\ &= \text{argmax}_j\{\, \langle \bar{\mathbf{x}}_j, \bar{\mathbf{x}} \rangle - \sum_{i=1}^{n} \mathbf{s}_i \cdot \langle \bar{\mathbf{x}}_i, \bar{\mathbf{x}}_j \rangle \mid j \in \mathcal{Z} \,\}. \end{aligned}$$

Lemma 1 assures that those points belong to the frame. Since there is no other way of selecting points, all of them belong to the frame. Hence, the condition $\mathbf{s}_i \neq 0 \Rightarrow \mathbf{x}_i \in \mathcal{F}$ is satisfied. The claim follows with Theorem 1. □

The following proposition from Lawson and Hanson (1995) characterizes the solution of this method.

Proposition 2 (Kuhn-Tucker conditions)**.** *A vector $\mathbf{s} \in \mathbb{R}^n$ is a solution for $f(\mathbf{s}) = \frac{1}{2}\|\bar{\mathbf{X}}\mathbf{s} - \bar{\mathbf{x}}\|_2^2$ if and only if there exists a vector $\mathbf{w} \in \mathbb{R}^n$ and a partitioning of the integers from 1 to $d+1$ into subsets $\mathcal{Z}$ and $\mathcal{P}$ such that with $\mathbf{w} = -\nabla f(\mathbf{s}) = \bar{\mathbf{X}}^\top(\mathbf{x} - \bar{\mathbf{X}}\mathbf{s})$ it holds that*

$$\begin{aligned} &\mathbf{s}_i = 0 \quad \textit{for } i \in \mathcal{Z}, && \mathbf{s}_i > 0 \quad \textit{for } i \in \mathcal{P}, \\ &\mathbf{w}_i \leq 0 \ \textit{for } i \in \mathcal{Z}, && \mathbf{w}_i = 0 \ \textit{for } i \in \mathcal{P}. \end{aligned}$$

Proposition 1 states that no more than $d+1$ extreme points are needed to define a point. In addition, Algorithm 3 chooses iteratively one extreme point after another. As it is guaranteed to lower the error in each iteration (Lawson & Hanson, 1995) it terminates with at most $d+1$ non-negative positions and yields a sparse convex combination.

3.4. Computing the Frame

Until now, we represented a single point as a sparse convex combination of points on the frame. By doing this for every point $\mathbf{x}_i$ $(i = 1, 2, \ldots, n)$ in the data set, we obtain the frame $\mathcal{F}$ of $\mathcal{X}$. Algorithm 4 summarizes this procedure, called *NNLS-Frame*, and Corollary 1 proves this claim.

Corollary 1. *Algorithm 4 yields the frame $\mathcal{F}$ of $\mathcal{X}$.*

Proof. Theorem 2 ensures that the positive positions of every solution $\mathbf{s}_k$ $(k = 1, 2, \ldots, n)$ refers to points on the frame. Taking the union of those positions recovers the frame indices $\mathcal{E}$ since every extreme point defines itself and therefore its index is part of the union. □

Note that Algorithm 4 already realizes a matrix factorization for the special case $p = q$. This is also stated in the following corollary.

Algorithm 4 NNLS-Frame

Input: design matrix $\bar{\mathbf{X}}$
Output: indices of ext. points $\mathcal{E}$ and weight matrix $\mathbf{W}$
$\mathcal{E} = \emptyset$
$\mathbf{W} = \mathbf{0} \in \mathbb{R}^{n \times n}$
for $k = 1, 2, \ldots, n$ **do**
  $\mathbf{s}_k = \text{NNLS}(\bar{\mathbf{X}}, \bar{\mathbf{x}}_k)$
  $\mathcal{P}_k = \{\, i \in \{1, 2, \ldots, n\} \mid (\mathbf{s}_k)_i > 0 \,\}$
  $\mathcal{E} = \mathcal{E} \cup \mathcal{P}_k$
  $\mathbf{W}[k, :] = \mathbf{s}_k$
end for

Corollary 2. *Let $\mathcal{E}$ and $\mathbf{W}$ and be the result of Algorithm 4. Then it holds that*

(i) $\mathbf{X} = \mathbf{X}\mathbf{W} = \mathbf{X}[:, \mathcal{E}]\mathbf{W}[\mathcal{E}, :]$ *is a factorization of the design matrix $\mathbf{X}$ with $\mathbf{W}[\mathcal{E}, :] \in \mathbb{R}^{p \times n}$ being a non-negative stochastic matrix and $\mathbf{X}[:, \mathcal{E}] \in \mathbb{R}^{d \times p}$ being the submatrix of $\mathbf{X}$ containing only the frame of $\mathcal{X}$.*

(ii) the factorization above is lossless, i.e.,

$$\|\mathbf{X} - \mathbf{X}[:, \mathcal{E}]\mathbf{W}[\mathcal{E}, :]\|_F^2 = 0.$$

3.5. Complexity Analysis

There are two main computations within the NNLS method in Algorithm 3. First, the negative gradient is being computed, which has a complexity of $\mathcal{O}(n(d+1))$. Second, an unconstrained least-squares problem $\mathbf{z} = \text{argmin}_{\mathbf{z}} \|\bar{\mathbf{X}}_{\mathcal{P}}\mathbf{z} - \bar{\mathbf{x}}\|_2^2$ is being solved. The latter can be rewritten as $\mathbf{z}[\mathcal{P}] = (\bar{\mathbf{X}}_{\mathcal{P}}^\top \bar{\mathbf{X}}_{\mathcal{P}})^{-1} \bar{\mathbf{X}}_{\mathcal{P}}^\top \bar{\mathbf{x}}$ and has a complexity of $\mathcal{O}((d+1) \cdot |\mathcal{P}|^2)$, where the average size of $\mathcal{P}$ is $\frac{1}{2}(d+1)$. Since no more than $(d+1)$ points are sufficient for the solution (compare Theorem 2) the while loop is being executed at most $(d+1)$ times. Therefore, the complexity of the NNLS method is $\mathcal{O}(\text{NNLS}) = \mathcal{O}((d+1)[\frac{1}{4}(d+1)^3 + n(d+1)])$ on average. The complexity of NNLS-Frame presented in Algorithm 4 is $\mathcal{O}(n \cdot \text{NNLS}) = \mathcal{O}(\frac{n}{4}(d+1)^4 + n^2(d+1)^2)$. Actually, the multiplier is less than n since points which are already known to be extreme can be omitted. In contrast, the fastest LP-based approach (Dulá & López, 2012) so far scales in $\mathcal{O}(n \cdot \text{LP}_{d+1,|\mathcal{F}|} + dn|\mathcal{F}|)$, where $\text{LP}_{r,t}$ is the runtime of a LP with r equality constraints and t non-negative variables.

3.6. Computing the Frame Efficiently

One way to speed up the frame computation is a divide and conquer approach. The underlying principle is stated in the following lemma.

Lemma 2. *Let $\mathcal{A}$ and $\mathcal{B}$ be non-empty discrete sets, then*

$$\text{conv}(\mathcal{A} \cup \mathcal{B}) = \text{conv}\left(\text{conv}(\mathcal{A}) \cup \text{conv}(\mathcal{B})\right).$$

Algorithm 5 Divide and Conquer strategy of NNLS-Frame

Input: data $\mathcal{X}$, number of splits $K \in \mathbb{N}$
Output: indices of extreme points $\mathcal{E}$
$\mathcal{X} = \bigcup_{k=1}^{K} \mathcal{X}^{(k)}$ with $\mathcal{X}^{(i)} \cap \mathcal{X}^{(j)} = \emptyset$ for $i \neq j$
$\mathcal{E} = \emptyset$
for $k = 1, 2, \ldots, K$ **do**
  $\mathcal{E}_k = \text{NNLS-Frame}(\mathcal{X}^{(k)})$
  $\mathcal{E} = \mathcal{E} \cup \mathcal{E}_k$
end for
$\mathcal{E} = \text{NNLS-Frame}(\mathcal{X}_{\mathcal{E}})$

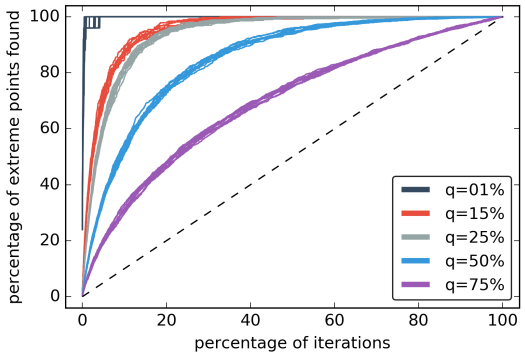

Figure 3. Percentage of discovered extreme points versus percentage of iterations on synthetic data with $n = 2500$ points in $d = 5$ dimensions.

The idea is as follows. Let $\mathcal{X}^{(1)} \cup \ldots \cup \mathcal{X}^{(K)}$ be a random partition of $\mathcal{X}$. The size n_k of every subset $\mathcal{X}^{(k)}$ should be significantly smaller than the size of $\mathcal{X}$, i.e., $n_k \ll n$. The assumption of having a pairwise disjunction is not necessary but reasonable. Instead of the whole data set $\mathcal{X}$, Algorithm 4 is now executed on every subset $\mathcal{X}^{(k)}$ for $k = 1, 2, \ldots, K$. Finally, the frame of $\mathcal{X}$ is obtained by merging the frames of every subset and run the proposed approach again on it. The procedure is summarized in Algorithm 5. The `for`-loops of Algorithms 2, 4 and 5 can be trivially parallelized.

4. Experiments

4.1. Computing the Frame

For the experiments in this section we want to control the frame density. Hence, use the same synthetic data[1] as in Dulá and López (2012) which was generated according to a procedure described in López (2005). The data consists of $n = 2500, 5000, 7500, 10000$ data points in $d = 5, 10, 15, 20$ dimensions with a frame density of $1, 15, 25, 50, 75$ percent respectively.

[1] http://www.people.vcu.edu/~jdula/FramesAlgorithms/

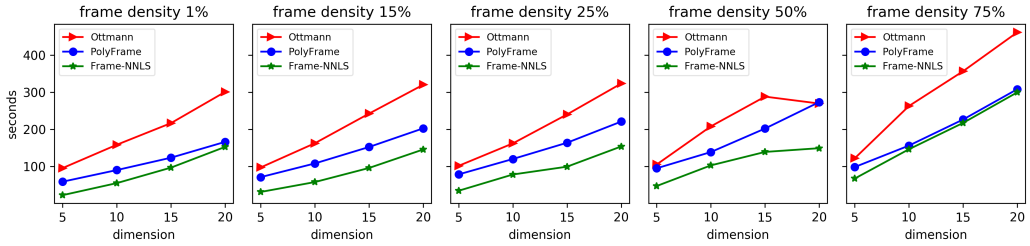

Figure 4. Timing results on synthetic data with $n = 10000$ points.

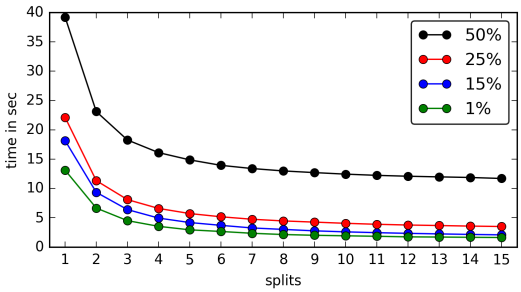

Figure 5. Timing results for the divide and conquer approach on synthetic data with $n = 10000$ points in $d = 5$ dimensions.

The first experiment is about the discovery of extreme points. In contrast to LP-based approaches (Dulá & Helgason, 1996; Ottmann et al., 2001), which discover up to one extreme point per iteration, NNLS-Frame finds up to $d + 1$ extreme points. This is a result of Theorem 2 and illustrated in Figure 3. The graph shows the percentage of discovered extreme points against the percentage of iterations conducted on a synthetic data set with $n = 2500$ points in 5 dimensions with various frame densities. The lower the frame density the faster the frame is being discovered. Even for a fairly high frame density of 75%, the discovery is faster than approximately linear as one could expect from an LP-based approach. This is depicted as a dashed line. Note that if an approximation of the frame is sufficient, the computation could be stopped.

In our second experiment we show that finding the frame is faster than using LP-based approaches. The two baselines are taken from Ottmann et al. (2001) and Dulá and Helgason (1996). We make use of the implementation by Dulá and López (2012) and implement our approach in the same programming language for compatibility. We test on data sets of sizes $n = 2500, 5000, 7500, 10000$ and of dimensionality $d = 5, 10, 15, 20$. Figure 4 shows timing results for $n = 10000$. The figure shows that our approach is always faster than the baselines, irrespectively of the configuration. We obtain the same result on the remaining data set sizes $n = 2500, 5000, 7500$ which are not shown.

The third experiment is about the divide and conquer approach introduced in Section 3.6. As shown in Figure 5, the runtime is halved for every configuration using only three partitions. The take-home message is, that even a small number of partitions can lead to a substantial speed up. Note that those partitions can be processed in parallel.

Table 1. Data sets and their frame size and density.

data set		n	d	q	frame density
Banking2	(Dulá & López, 2012)	12456	8	715	5.74%
Banking1	(Dulá & López, 2012)	4971	7	345	6.94%
USAFSurvey	(Epifanio et al., 2013)	2420	6	368	15.21%
yeast	(Horton & Nakai, 1996)	1484	8	242	16.31%
Banking3	(Dulá & López, 2012)	19939	11	4960	24.88%
SpanishSurvey	(Vinué, 2017)	600	5	150	25.00%
swiss-heads	(Cutler & Breiman, 1994)	200	6	115	57.50%
skel2	(Heinz et al., 2003)	507	10	431	85.01%
ozone	(Cutler & Breiman, 1994)	330	10	308	93.33%

4.2. Matrix Factorization

We now compare the frame-based Archetypal Analysis, Frame-AA (F-AA), to several baselines including standard Archetypal Analysis (Cutler & Breiman, 1994) (AA), ConvexHull-NMF (Thurau et al., 2011) (CH-NMF), Convex-NMF (Ding et al., 2010) (C-NMF) and standard NMF (Lee & Seung, 1999). The first two methods are implemented in Python. For CH-NMF and C-NMF we use *pymf* [2], and for NMF we use *scikit-learn* [3]. Table 1 depicts the data sets used for this experiment. The frame sizes and densities are computed with our proposed NNLS-Frame method.

Table 2 reports on the results for the choice of $p = 6$ in terms of reconstruction error measured with the Frobenius norm. We obtain similar results for $p = 8, 10, 12$. The number of iterations executed per algorithm is fixed to 100 in order to obtain fair results. We use random initializations for all algorithms and report on averages over 36 repetitions. As seen in Table 2, F-AA yields similar errors as AA. The lowest errors are achieved with NMF. The baseline CH-NMF, which approximates the

[2] http://pypi.python.org/pypi/PyMF
[3] http://scikit-learn.org

Table 2. Average Frobenius norm reported on 36 repetitions with random initializations for $p = 6$. "-" indicates that the method could not be executed due to negative data.

data set	F-AA	AA	CH-NMF	C-NMF	NMF
Banking2	90.08	72.35	-	-	-
Banking1	67.20	58.97	-	-	-
USAFSurvey	904.22	902.07	1239.67	2192.30	688.35
yeast	5.43	5.02	9.18	7.04	3.52
Banking3	134.30	131.81	-	-	-
SpanishSurvey	94.84	93.51	117.91	254.92	32.14
swiss-heads	75.05	74.67	87.06	116.07	47.16
skel2	64.84	64.87	77.78	101.73	51.75
ozone	1532.12	1669.70	-	-	-

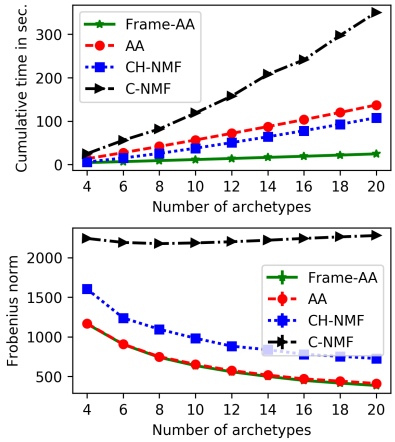

Figure 6. Cumulative time for several evaluations of p as well as reconstruction error for USAFSurvey data.

frame instead of computing it exactly, yields higher error of approximately 20% while C-NMF is even worse. In summary, the error of F-AA is similar to standard AA and much better than CH-NMF as well as C-NMF.

Usually, it is a-priori not known how to choose the latent dimensionality p. A standard approach is to choose p with respect to the so-called elbow criterion which requires several runs for different values of p to be executed. In such a scenario our approximative approach is particularly beneficial. Frame-AA requires the computation of the frame $\mathcal{F}$ before archetypes can be located. However, once the frame is complete, it can be used for finding any number of archetypes. Hence, it is interesting to see the cumulative time taken by the methods when evaluating several configurations, say $p = 4, 6, 8, \ldots, 16$.

This is depicted in Figure 6 for the USAFSurvey data set. We report again on averages as well as standard errors on 36 repetitions executed on random initializations. Frame-AA turns out fastest at the first evaluation of $p = 4$ despite the computation of the frame. Since the frame is static for a data set, it can be reused for all remaining computations and Frame-AA clearly outperforms its peers. Note that the divide and conquer strategy proposed in Algorithm 5 leads to an additional speed-up that is not shown here. The reconstruction error is shown in the bottom part of Figure 6. Once again, Frame-AA yields a very similar error as standard AA, which is computed on the whole data set, while C-NMF and CH-NMF baselines perform much worse.

5. Autoencoders

We consider a prototypical application of Frame-AA as an autoencoder similar to the approach of Bauckhage et al. (2015). However, instead of focusing on the reconstruction, we are interested in the quality of the learned embedding given by the weights of the convex combinations.

For this exemplary task we make use of the mnist data set. It consists of handwritten images of the digits zero to nine which are of size 28×28 rendering the problem 784-dimensional. We split every image into 4×4 sub-images yielding 49 patches per image. Those patches are stacked-up to form a new design matrix. For lack of space we focus on a subtask and aim to separate digit 1 from digit 7. The task is thus phrased as a standard binary classification task. We sample 500 images per class. The final design matrix is of size 49000×16 and its frame density is approximately 14%. The design matrix is then factorized as described in Section 3.2. After the factorization, every patch is described by p weights contained within the matrix $\mathbf{A}$. The weights of the convex combinations are then vectorized yielding an embedding of size $m = 49p$ per image.

As baseline we employ a deep learning autoencoder that uses a convolutional layer (LeCun et al., 1989) to learn intermediate representations of images. To perform a fair comparison, the convolutional layer learns p filters of the same size as Frame-AA, that is 4×4 and stride 4×4. This layer has a sigmoid activation function and is followed by an upsampling layer that recovers the original image size. The last layer is another convolutional layer with one filter of size 4×4, stride one, and sigmoid activation. We optimize the network using squared loss as in the matrix factorization. We compute an embedding analogously to Frame-AA by representing an image by its output of the first convolutional layer, yielding the same embedding size of $m = 49p$.

For every obtained embedding, an SVM is trained on $p = 2, 4, \ldots, 16$ archetypes/filters meaning that every image is now represented by a vector of size $m = 98, 196, \ldots, 784$. Note that the last one is identical to the original image size. We deploy an RBF kernel and the parameters C and γ are optimized on a $2^{-7}, \ldots, 2^{7}$ grid respectively. For

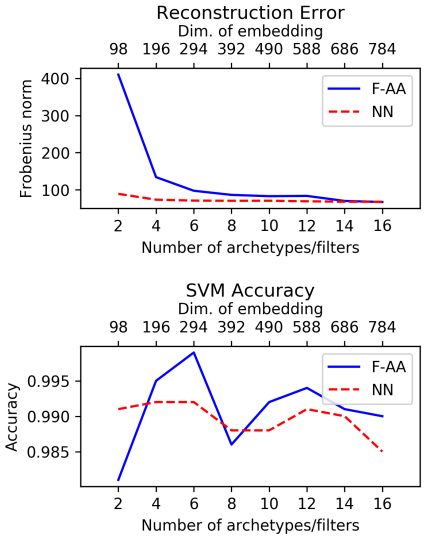

Figure 7. Predictive performance in terms of accuracy and reconstruction error in terms of Frobenius norm for various embedding sizes.

the hyperparameter search we use a hold out set consisting of another 500 images per class that are disjoint from the training set.

To evaluate the predictive performance we sample another 500 test images per class and obtain their embeddings using the trained approaches described above. The results are depicted in Figure 7. The top part of the figure depicts the reconstruction errors in terms of the Frobenius norm. The curve of Frame-AA has a typical elbow-structure as one would expect, while the error of the net is almost static. Although the net achieves smaller reconstruction errors, Figure 7 (bottom) shows that the SVM effectively leverages the representation learned by F-AA. The archetype-based embedding leads to more accurate classification models.

The learnt archetypes/filters are shown in Figure 8 for the case of $p = 8$ yielding an embedding of size $m = 392$. The archetypes in the left part of the figure resembles edge and border detectors while the filters on the right appear random. Moreover, every patch is represented as a convex combination of the shown archetypes. That is, the classification of a patch can be explained by the corresponding weights to render the results interpretable.

6. Conclusion

We proposed a novel method for computing the frame of a data set. The frame is a subset of the data set containing only the extreme points, i.e., the vertices of the convex hull of the data. While standard approaches like Quickhull were infeasible for scenarios with more than three dimensions,

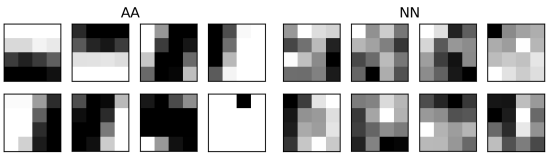

Figure 8. Archetypes (left) and filters (right) for $p = 8$.

we computed the frame by leveraging the well known active-set method for non-negative least squares problems called NNLS. We provided a theoretical underpinning for our approach and conducted a series of experiments to compare the computation of the frame with two LP-based approaches to show our competitiveness.

Moreover, we proposed an approximation of Archetypal Analysis called Frame-AA by restricting the optimization of the archetypes to the frame. We showed that if a small number of data points approximated the frame sufficiently, the resulting approximation of AA on the whole data set turned out appropriate. This approximation does not only work for Archetype Analysis but also for all variants of it like for instance Mørup and Hansen (2010), Chen et al. (2014), and Bauckhage et al. (2015).

Empirically, we showed that the error of Frame-AA is only slightly higher than standard Archetypal Analysis but was often much faster once the frame has been computed. We leveraged this observation and proposed an efficient model selection strategy to find the optimal number of archetypes. We proposed to compute the frame only once and then re-used it for all subsequent computations for other numbers of archetypes. We observed an enormous acceleration in scenarios with low frame densities.

Low frame densities were attained when only small portions of the data were extreme. On the other hand, high frame densities diminish this speed-up and the computational time converges to that of a standard Archetypal Analysis. However, we argued that these scenarios are not suited for Archetypal Analyses in the first case.

Furthermore, we compared Frame-AA to several state-of-the-art baselines. Frame-AA either outperformed its competitors or performed on-par. On the example of an autoencoder, Frame-AA led to nicely interpretable classifications in contrast to convolutional neural networks.

Acknowledgements

This work has been funded in parts by the German Federal Ministry of Education and Science BMBF under grant QQM/01LSA1503C.

References

Arora, Raman, Gupta, Maya, Kapila, Amol, and Fazel, Maryam. Clustering by left-stochastic matrix factorization. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 761–768, 2011.

Arora, Raman, Gupta, Maya R, Kapila, Amol, and Fazel, Maryam. Similarity-based clustering by left-stochastic matrix factorization. *Journal of Machine Learning Research*, 14(1):1715–1746, 2013.

Barber, C Bradford, Dobkin, David P, and Huhdanpaa, Hannu. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software (TOMS)*, 22(4): 469–483, 1996.

Bauckhage, C, Kersting, HF, Thurau, C, et al. Archetypal analysis as an autoencoder. In *Workshop New Challenges in Neural Computation 2015*, pp. 8. Citeseer, 2015.

Brondsted, Arne. *An introduction to convex polytopes*, volume 90. Springer Science & Business Media, 2012.

Chen, Yuansi, Mairal, Julien, and Harchaoui, Zaid. Fast and robust archetypal analysis for representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1478–1485, 2014.

Cutler, Adele and Breiman, Leo. Archetypal analysis. *Technometrics*, 36(4):338–347, 1994.

Ding, Chris HQ, Li, Tao, and Jordan, Michael I. Convex and semi-nonnegative matrix factorizations. *IEEE transactions on pattern analysis and machine intelligence*, 32(1):45–55, 2010.

Dulá, José H and Helgason, Richard V. A new procedure for identifying the frame of the convex hull of a finite collection of points in multidimensional space. *European Journal of Operational Research*, 92(2):352–367, 1996.

Dulá, José H and López, Francisco J. Competing output-sensitive frame algorithms. *Computational Geometry*, 45(4):186–197, 2012.

Epifanio, Irene, Vinué, G, and Alemany, Sandra. Archetypal analysis: contributions for estimating boundary cases in multivariate accommodation problem. *Computers & Industrial Engineering*, 64(3):757–765, 2013.

Heinz, Grete, Peterson, Louis J, Johnson, Roger W, and Kerk, Carter J. Exploring relationships in body dimensions. *Journal of Statistics Education*, 11(2), 2003.

Horton, Paul and Nakai, Kenta. A probabilistic classification system for predicting the cellular localization sites of proteins. In *Ismb*, volume 4, pp. 109–115, 1996.

Lawson, Charles L and Hanson, Richard J. *Solving least squares problems*, volume 15. SIAM, 1995.

LeCun, Yann, Boser, Bernhard, Denker, John S, Henderson, Donnie, Howard, Richard E, Hubbard, Wayne, and Jackel, Lawrence D. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

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

Lopez, Francisco-Javier. Generating random points (or vectors) controlling the percentage of them that are extreme in their convex (or positive) hull. *Journal of Mathematical Modelling and Algorithms*, 4(2):219–234, 2005.

Mørup, Morten and Hansen, Lars Kai. Archetypal analysis for machine learning. In *Machine Learning for Signal Processing (MLSP), 2010 IEEE International Workshop on*, pp. 172–177. IEEE, 2010.

Ottmann, Thomas, Schuierer, Sven, and Soundaralakshmi, Subbiah. Enumerating extreme points in higher dimensions. *Nordic Journal of Computing*, 8(2):179–192, 2001.

Paatero, Pentti and Tapper, Unto. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2):111–126, 1994.

Thurau, Christian, Kersting, Kristian, Wahabzada, Mirwaes, and Bauckhage, Christian. Convex non-negative matrix factorization for massive datasets. *Knowledge and information systems*, 29(2):457–478, 2011.

Vinué, Guillermo. Anthropometry: An R package for analysis of anthropometric data. *Journal of Statistical Software*, 77(6):1–40, 2017.