IADIS International Journal on Computer Science and Information Systems
Vol. 9, No. 2, pp. 67-81
ISSN: 1646-3692

USER ASSISTED EXPLORATION AND SAMPLING OF THE SOLUTION SET OF NON-NEGATIVE MATRIX FACTORIZATIONS

Joachim Staib, Marcel Spehr and Stefan Gumhold, *Chair of Computer Graphics, TU Dresden, Germany*

ABSTRACT

The non-negative matrix factorization provides a valuable tool for the analysis of positive data by representing it as an additive linear superposition of a small number of non-negative basis elements. This property allows the base elements to be interpreted in the same domain as the input data. The problem though lies in the ambiguity of equally valid solutions from which only one is obtained. Its selection depends on the initialization of the applied factorization algorithm or further constraints.
We propose a new approach which is based on sampling the set of valid factorizations, given one initial solution. First we derive a parameterization of the set of valid solutions by means of a strong membership oracle. This function returns true if a parameter tuple represents a valid solution and false otherwise. Furthermore, we present an algorithm that explores and samples parts of the non-convex solution set. To assist the otherwise automatic process and to alleviate the drawbacks of sampling a non-convex space, we provide a graphical user interface that puts the user in the loop. From an initial set of samples the user is allowed to select elements that serve as the starting point for subsequent samplings. With this browser-like tool a steering of the sampling of the NMF can be performed without further knowledge on the underlying algorithm and without the need to express possibly hard to formulate constraints. An evaluation of the sampling procedure reveals promising results for a factorization of data in up to 4 basis elements.

KEYWORDS

Data Mining, Interactive Sampling, Non-Negative Matrix Factorization

1. INTRODUCTION

Non-negative matrix factorizations (NMF) have a wide range of applications in the domain of data mining and understanding, for example in text mining, analysis of emission spectra, micro array analysis or music transcription. Given a set of non-negative data like documents, measured amplitude spectra or images, the NMF aims to find a small set of elements that serve

as a linear basis to approximate the complete data. In contrast to other widely used techniques like PCA or SVD this basis is constrained to be non-negative and thus rendering it meaningful in the domain of the original data. For example the basis of a set of images can be understood as images as well.

For factorizations in general, discrete data points are arranged column-wise in a matrix. Given n observations with size m each, the resulting data matrix is $\overline{D} \in \mathbb{R}_+^{m \times n}$. NMF factorizes this matrix into two matrices U and V with possibly lower rank k such that $\overline{D} = U \cdot V + E$ where U is called the *basis factor* while V is the *weighting factor* plus a residual matrix E. This implies an interpretation of the observations in D as an additive linear superposition of the basis elements encoded in the columns of U. For specific types of data, such as absorption spectra, this assumption is physically sound. For the remainder of this paper we assume the residual to equal the zero matrix, thus a data matrix D can be fully described by $U \cdot V$.

Besides probabilistic approaches, NMF algorithms usually formulate the problem as an optimization task that minimizes the residual with respect to some distance or divergence measure, for example Euclidean distance (Lee, 1999). In general the optimization problem is highly non-convex, and many pairs of U and V matrices exist that approximate the data equally well (in fact, under special circumstances the problem can be convex, see Esser, 2011 for more details). Existing solvers impose additional constraints in the optimization to obtain a unique solution, e.g. sparseNMF (Hoyer, 2004) favors solutions where each observation is approximated by a small number of the basis elements in U. Whereas the detNMF (Schachtner, 2009) algorithm seeks basis vectors that are as close to the data as possible. However, especially in the field of data mining, constraints like the aforementioned might be impossible to formulate, because the scientist only has a fuzzy idea of them or he might not be able to formulate them mathematically.

We propose a complementary approach which is based on a sampling of the set of possible factorizations. Because of its non-convex nature, this is a challenging task. Without further constraints it is considered impossible to achieve a guaranteed coverage of the complete set. Our solution alleviates this problem in two directions: via a region growing process that occupies as much space as possible and by putting the user in the loop. After a sampling of a part of the solution space, the user is presented with representative solutions to steer subsequent sampling processes into new parts of the solution space. This makes it possible to get desired solutions without a complete sampling of the (unknown) space. Instead of using one factorization method and hoping for the best, it is possible to enumerate many factorizations and let the user subsequently steer to a solution that meets his criteria. Section 6 explains this user interface in greater detail.

Our main contributions are:

- Based on an initial factorization, we define a parameterization of the solution space. To test a parameter tuple for the membership of this parameter space we define an oracle function that returns true if a parameter tuple leads to a valid factorization and false otherwise,
- Given such an oracle we present a novel sampling technique for non-convex sets that generates a uniform sampling of a part of the solution space via region growing and
- We present a user interface to guide the sampling process of a part of this non-convex space into desired regions. The tool follows the metaphor of selecting solutions and locking parts of one solution and thus allows even a novice in the field of NMF to find a desired solution without the need to expose the underlying factorization and sampling algorithms.

The remainder of this paper is organized as follows. In section 2 related work is reviewed. Sections 3 to 5 comprise the major part of this work. In Section 3, a parameterization of the set of factorizations for one dataset is derived. The sampling technique is explained in section 4 and evaluated in section 5. Section 6 provides an overview of the visual component. A conclusion is given in section 7.

2. RELATED WORK

Although to the best of our knowledge, no directly comparable work exists, much work has been done in the field of NMF itself and the analysis of its ambiguity. Many efficient algorithms have been proposed beginning with the algorithms of Lee and Seung in (Lee, 1999). Donoho and Stodden define criteria under which a factorization delivers correct results (Donoho, 2003). Plis et al. show that a unique factorization cannot be guaranteed under the presence of correlated noise (Plis, 2010). Laurberg and Hansen (Laurberg, 2008) analyze conditions for unique factorizations and show that a factorization is not unique if an additive offset is applied, that means if none of the weight coefficients in one row of the weighting matrix V is zero. In our work we exploit this finding to generate non-unique test data sets.

The proposed solution shares properties with work that is settled in the field of interactive optimization. While the underlying process differs significantly, a strong similarity exists from the user's perspective; the interactive search for a desired solution, given initial data and constraints. In the work of Klau et al. this problem is referred to as "Human guided search" (Klau, 2010). They present an interactive tool where the user plans routing and layout tasks for vehicle routing that are then further optimized to comply with previously defined constraints. The work of Malinchik et al. (2004) is settled in the field of Interactive Evolutionary Computation that is especially suited for problems where the optimization tasks are unclear. A study is described in which the user is presented different solutions of a point clustering result to be rated. The underlying algorithm uses this information to apply evolutionary strategies like combination and mutation in order to find the best solution. In the work of Bhuiyan et al. (2012) a tool is presented that aims for the identification of frequent patterns in data by allowing the user to make a binary decision between presented samples of data. The complete data set is considered unknown and only samples can be obtained on demand. A sampling strategy is applied that adjusts heuristics based on the made decisions. While the concept of sampling unknown spaces is related to our work we do not adjust heuristics in this sense. Another noteworthy reference is the work of Chimani et al. (2005) where they show a case study in large-scale human-in-the-loop optimization for selected problems. They conclude that an appropriate visualization that disrupts the users input only minimally after reoptimization is one crucial property of a good visual optimization.

In general, the sampling of sets where only a membership oracle is given, is the common basis of a variety of problems. Among others, it is the foundation of many Monte-Carlo based volume approximation algorithms. Widely used methods include rejection-sampling approaches as in (Press, 1990) that produce good results in low dimensions but are prone to fail for high dimensional spaces. Random-walk methods have been a subject of study over the last decades and led to a variety of approaches, suited for convex spaces, such as different Markov-Chain-Monte-Carlo methods like hit-and-run (Kiatsupaibul, 2011) or simulated annealing approaches (Lovász, 2003). A comprehensive survey of volume approximation

algorithms based on oracles and sampling can be found in (Simonovits, 2003). Evolutionary algorithms, such as Gaussian adaptation where a multidimensional Gaussian is adapted to an unknown space are used, for example, in network optimization tasks (Kjellström, 1969), but are of limited use for non-convex spaces either.

In the case of non-convex sets, not much work is available. An approach based on a combination of global and local sampling techniques as described in (Zamora-Sillero, 2011) is used to estimate a parameter set for systems biology. The process is split in two phases: a global exploration where ellipsoids are fitted into a non-convex space and a local uniform sampling in the inner of these primitives. Although it shares ideas with the method proposed in this work it is rather hard to implement and not suited for higher dimensions as it suffers from the decay of volume for high-dimensional ellipsoids.

As will be seen, the visual tool incorporates a set of standard visualization techniques, such as parallel coordinates which has gained a greater attention through the work of Alfred Inselberg (2010).

3. PARAMETERIZATION OF NMF-FACTORIZATIONS

In order to sample the solution set, a parameterization of possible solutions is defined. Let U be the column-wise basis and V be the matrix of linear coefficients with rank k of a data matrix D. In general other matrices U' and V' exist that reconstruct the same data matrix. These factors can be derived from rank k matrices U and V through an invertible transformation matrix $\mathrm{T} \in \mathbb{R}^{\mathrm{k}\times\mathrm{k}}$ as

$$\mathrm{D} = \mathrm{UV} = \mathrm{UTT}^{-1}V = (UT)(T^{-1}V) = U'V'. \quad (1)$$

This means, given a basis matrix U, its corresponding weighting matrix V and a transformation matrix T, a new factorization with a basis matrix UT and a weighting matrix $T^{-1}V$ can be obtained.

Clearly, not all invertible matrices T lead to non-negative factors U' and V'. Therefore we define the notion of a valid transformation.

Def. 1: An invertible matrix $\mathrm{T} \in \mathbb{R}^{\mathrm{k}\times\mathrm{k}}$ is called *valid with respect to a non-negative factorization UV*, iff $\mathrm{UT} \geq 0$ and $\mathrm{T}^{-1}\mathrm{V} \geq 0$. The set of all valid transformations is called the *transformation set*.

These transformation matrices and the set of basis factors are in a one-to-one relation through $U' = UT$. Among others, this includes all non-negative invertible transformations with non-negative inverse. It can be shown that this is exactly the class of pure scaled permutation matrices (Minc, 1974). We call these matrix components the trivial transformations as they do not reveal anything compared to the initial factorization UV; while scale matrices only apply a constant factor to each of the base vectors, permutation matrices rearrange their order. In fact, permutations and scaling of otherwise identical solutions hinder the process of visual inspection of the solution set and should not be part of the final transformation set. The remaining transformation set is defined as follows:

Def. 2: The set $\mathrm{T_r} \subseteq \mathrm{T}$ that does not scale or permute the columns in U is called the *relevant transformation set*.

As will be shown in the following subsection, the degree of freedom of a relevant transformation matrix for a factorization with rank k reduces from $k \cdot k$ to $(k-1) \cdot k$. Storing these degrees of freedom in a tupleleads to the notion of the valid parameter set:

Def. 3: The set of $(k-1) \cdot k$-dimensional parameter tuples that lead to relevant transformations $\mathrm{T_r} \subseteq \mathrm{T}$ is called *the transformation parameter set*.

This set in turn stands in a one-to-one relation with the relevant transformation set.
Elements from this parameter set *are to be sampled* in the remainder of this work. They generate the elements from the *relevant transformation set* which in turn generate a subset of the solution set that does not scale or permute columns of U. In conclusion the relationship between the transformation parameter set, the relevant transformation set and the transformed base matrices can be expressed as:

$$\varphi \in \mathbb{R}^{(k-1)\cdot k} \leftrightarrow T_r \subseteq T \leftrightarrow \{UT \mid T \in T_r\} \tag{2}$$

The membership of the *transformation parameter set* is expressed through a function $\Omega(\varphi) = \mathit{true}$ or *false* which is called an *oracle* and returns true if a parameter tuple φ leads to a relevant transformation matrix.
It is composed of

(1) a function $\tau(\varphi)$ that maps a parameter tuple $\varphi \in \mathbb{R}^{(k-1)\cdot k}$ to a $k \times k$ transformation matrix while ensuring a constant scale,
(2) the permutation oracle $\Omega_{Perm}(T)$ that returns *true* if the resulting factors $\mathrm{U'} = \mathrm{UT}$ and $\mathrm{V'} = \mathrm{T^{-1}V}$ do not contain permutations of U and V, and
(3) the NMF-oracle $\Omega_{nmf}(T)$.

Their composition is defined as:

$$\Omega(\varphi) \coloneqq \Omega_{Perm}(\tau(\varphi)) \wedge \Omega_{NMF}(\tau(\varphi)) \tag{3}$$

Algorithmically, a parameter tuple is first converted to a transformation matrix and then checked for validity with respect to the permutation oracle and the NMF-oracle. While the NMF-Oracle is naturally given by checking every entry of UT and $T^{-1}V$ for non-negative entries, the function $\tau(\varphi)$ and the permutation oracle have a more complex nature.

3.1 Independence of Scaling

In this step, a $k \times k$ matrix is constructed from a $(k-1) \cdot k$ dimensional parameter tuple. Note that this matrix does not have to be a valid transformation. This property is checked in a later step, using the NMF-Oracle Ω_{NMF} as can be seen in equation 3 above.

In order to have a constant scale on the column vectors in the basis matrix U, they are constrained to have unit-length. This must also be true in the transformed basis factors UT and thus forbids a scale in T, which is the desired property of this step.

From a given parameter tuple $\varphi \in \mathbb{R}^{(k-1)\cdot k}$, a transformation matrix is constructed, that preserves unit length in the transformed factor $U' = UT$. This means the equation $\|UT\|^2 = (UT)^T(UT) = Z$, where Z is a $k \times k$ matrix with ones on the main diagonal, must hold. To

achieve this property, the transformation is generated from two factors: a pre-transformation T_0, which is purely derived from the initial factorization, such that $(UT_0)^T(UT_0) = I$ and the main transformation T_1, defined by the parameter tuple φ, which preserves the unit length. The final transformation is then defined as the product $T = T_0T_1$. To obtain the pre-transformation T_0, the covariance matrix of the initial basis matrix is decomposed using an eigenvalue decomposition. Let this decomposition be given as $U^TU = R\,\Sigma\,R^T$. Then T_0 is defined as $T_0 = R\Sigma^{-0.5}$.
Proof:

$$\begin{aligned}(UT_0)^T(UT_0) &= T_0^T(U^TU)T_0\\ &= T_0^T(R\Sigma R^T)T_0\\ &= (R\Sigma^{-0.5})^T(R\Sigma R^T)(R\Sigma^{-0.5})\\ &= \Sigma^{-0.5}(R^TR)\,\Sigma(R^TR)\Sigma^{-0.5}\\ &= \Sigma^{-0.5}\Sigma\Sigma^{-0.5}\\ &= I\end{aligned}$$

In geometric terms, T_0 is the back transformation from an ellipsoid spanned by the columns of U to a unit hypersphere. To preserve the unit length of UT_0 after multiplication with T_1, the columns of T_1 must also have unit length. This reduces the degrees of freedom in T_1 from $k \cdot k$ to $(k-1) \cdot k$ because one entry per column is dependent on all other column entries. For the construction of a k-column vector with unit length from an arbitrary $(k-1)$-column vector, an inverse stereographic projection is used that, among other advantages, does not suffer from the sign ambiguity of the naïve way to calculate a unit-length k-vector from $(k-1)$ elements. Its inverse operation, the stereographic projection, is a well-known mapping technique that projects points from a sphere onto a plane. It is bijective and, except for one point, which is called the projection point, defined everywhere and vice versa. As projection point, the negated vector of ones is chosen as this can never be a column of a valid transformation matrix. Let $\phi(\vec{t})\colon \mathbb{R}^{k-1} \to \mathbb{R}^k$ be a function that performs this inverse stereographic projection. To generate a transformation matrix from the parameter tuple $\varphi \in R^{(k-1)\cdot k}$, $k-1$ subsequent elements of φ are combined to a vector for the inverse projection to form a column of T_1. Overall, the function τ to generate a transformation matrix from a parameter tuple has the following form:

$$\tau(\varphi) := T_0 \cdot \left(\phi(\varphi_{1\dots k-1}) \quad \phi(\varphi_{k\dots 2k-2}) \quad \dots \quad \phi\left(\varphi_{(k-1)(k-1)+1\dots(k-1)k}\right)\right) \tag{4}$$

3.2 Independence of Permutations

In order to restrict the permutation freedom, only those matrices T are considered valid that, under all permutations of columns in UT, minimize the difference to the untransformed matrix. In other words, when permuting UT with a permutation matrix P the minimal difference between U and UTP is found for P being the identity matrix: $I = argmin_P\|U - UTP2$.

For this constraint to be applicable we assume that the basis vectors in U differ enough, such that a permutation of two or more vectors yields a greater distance to the transformed matrix UT. The minimization problem can be efficiently computed by expressing it as a linear sum assignment problem (LSAP). Widely-used algorithms exist for this combinatorial problem, for example the Hungarian method. Given a gain matrix G that assigns some gain of k items (one per column) to k machines (one per row) a permutation is sought such that the

sum of the gain is maximized (Burkard, 2009). Here, the task is solved by checking if the permutation matrix that maximizes the trace of $U^T UT$ (which would correspond to the gain matrix in the LSAP context) is the identity matrix.

The proof can be given by expanding and simplifying the following minimization:

$$\begin{aligned} \mathrm{I} &= \operatorname{argmin}_{\mathrm{P}} \|\mathrm{U} - \mathrm{UTP}\|^2 \quad = \quad \dots \\ &= \operatorname{argmin}_{\mathrm{P}} (\operatorname{tr}(\mathrm{UU}^{\mathrm{T}}) - 2\operatorname{tr}(\mathrm{U}^{\mathrm{T}}\mathrm{UTP}) + \operatorname{tr}(\mathrm{UTT}^{\mathrm{T}}\mathrm{U}^{\mathrm{T}})) \\ &\equiv \operatorname{argmin}_{\mathrm{P}} (-\operatorname{tr}(\mathrm{U}^{\mathrm{T}}\mathrm{UTP})) \quad \equiv \quad \operatorname{argmax}_P \operatorname{tr}(\mathrm{U}^{\mathrm{T}}\mathrm{UTP}) \end{aligned}$$

In conclusion, the permutation oracle is given as:

$$\Omega(T)_{Perm} \coloneqq (\operatorname{argmax}_P \operatorname{tr}(U^T UTP) == I) \tag{5}$$

This concludes the definition of the oracle. In the next section our strategy is described to explore and samples parts of the transformation parameter set given through the oracle.

4. UNIFORM SAMPLING OF THE PARAMETER SET

Given the oracle $\Omega(\varphi)$, a sampling is performed that generates uniformly distributed parameter tuples $\{\varphi \in \mathbb{R}^{(k-1)\cdot k} | \Omega(\varphi) = true\}$. While the set is convex for $k = 2$ this does not hold for higher dimensions. The main cause for this non-convexity is the inversion of the generated matrix T to be applied on V which creates highly non-linear dependencies between parts of the φ-tuple.

No method is known that delivers a guaranteed sampling in such sets. We circumvent the effects of this problem by giving up the demand for a complete sampling and instead sample a region around a user defined point. As will be seen later, the user is able to select one sample in an interactive manner to guide further samplings into a direction he prefers.

The choice of the sampler depends on a variety of conditions.

- C1. It must be able to ensure a uniform sampling. This is especially imported for clustering and when the parameter set is seen as a space that has a defined volume. We claim that this volume is a direct measurement of the degree of the ambiguity in the NMF of a data set.
- C2. It must provide an efficient way to subsequently sample local parts of the parameter set
- C3. and also to sample subspaces as needed for the visual browser which is described in greater detail in section 6.

These requirements render the problem not well suited for widely-used algorithms like random walk, which does ensure a uniform sampling only for convex, well-formed sets and has to be restarted for every sampling point. Besides the low mixing rate of simpler instances of random walk algorithms, they cannot exploit the existence of prior samplings to accelerate the process.

The main idea of our proposed algorithm is to split the sampling process into two phases: (1) an exploration phase as shown in the following section and outlined in figure 1 and (2) a sampling phase as explained in section 4.2.

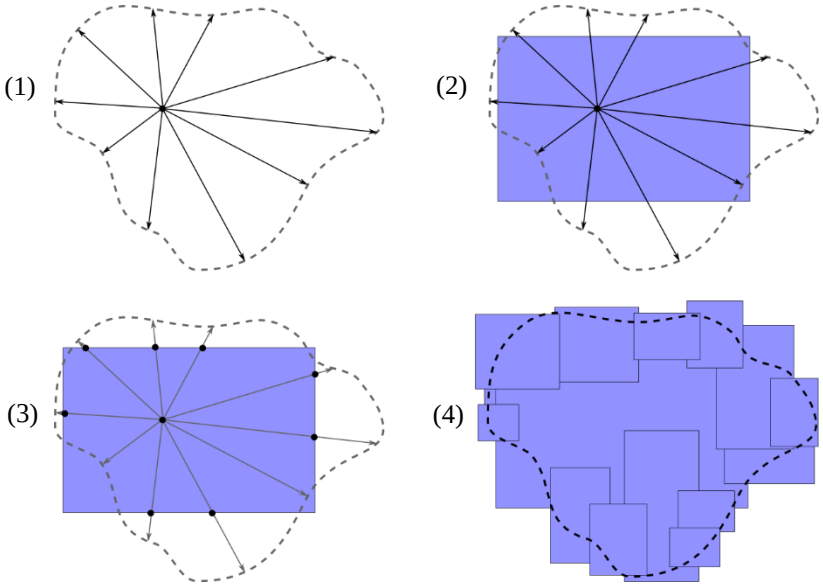

Figure 1. Outline of the exploration process. (1) From a starting point, rays are constructed to find border points of the viable space. (2) They serve to fit one of several cuboids (3) new possible center points are generated. (4) After a certain amount of iterations a part of the space is approximated by a set of overlapping cuboids

4.1 Exploration Phase

The algorithm creates a list of axis-aligned hyper cuboids, defined by a minimum and maximum point, to approximate parts of the parameter space. An integral part for their construction is a list that contains possible starting points for cuboids and initially contains one element.

One iteration involves the following steps: First, a point is taken from the list of center points. It serves as the common point for a set of rays that are used to find the borders of the reachable space. Then a cuboid is fitted into this space. In the next step, a number of points are created on its surface and, if valid, stored in the center list. This process is repeated until no more points are available or a fixed number of iterations is reached.

To generate the rays, a set of uniformly distributed directions is used. They are drawn from a normal distribution, see (Marsaglia, 1972) for further details. The number of directions depends on the dimensionality. For each ray, a continued bisection process is used to approximate the border of the parameter set which serves as a base for the minimum and maximum point of a new cuboid. This is done by partitioning each dimension into two sets, depending on whether the value at this dimension is smaller or larger than the corresponding value of the center point. The minimum values of the new cuboid are then found by averaging the values in these partitions and weighted with a constant that is proportional to $\frac{1}{\sqrt{n}}$ to prevent an overestimation for increasing dimensionalities.

The fitting of the cuboid to the sampled border points has ideally two properties. On the one hand, cuboids grown around center points that are far away from the border, should be

underestimated in order to not include far away border points. Otherwise the cuboid can enclose a huge invalid region especially in high dimensions. A simple example is the exploration of a sphere-shaped space. If the initial point is identical to the center of the sphere, the distances to these points equal its radius and so does the average. Without weighting the minimum and maximum point, the cuboid encloses the sphere completely. With increasing dimensionality the volume fraction of sphere and cuboid decreases exponentially which hinders an efficient sampling in the second step. The introduced weighting factor ensures dimension dependent underestimation of the cuboid to avoid this problem.

The second property concerns cuboids near the border. Here, the cuboids should ideally be overestimated to enclose bigger parts of the local border. Such an overestimation prevents the region near the border from being filled by a large number of small cuboids, but approximates it roughly by very few cuboids.

Averaging the distances from the found border points to generate minimum and maximum points with additional weighting meets these properties. If a center point (i.e. the center point of the rays to generate for one cuboid) has a high distance from the border, the mean distance of all border points that are reachable by a straight line becomes large. Due to the subsequent weighting this leads to a cuboid which surrounds a large part of this space, but does not contain the more distant border points. On the other hand, these distance values have a strong variance for center points near the border; small values are produced in the direction of the nearest border and large values in the direction of the farthest border. Here, the averaging leads to an overestimation and therefore creates a cuboid as desired, which approximates the border of the space roughly.

To keep the list of cuboids small, all previously generated cuboids and unprocessed starting points that are completely contained in the new cuboid are removed.

4.2 Uniform Sampling Phase

The list of cuboids is used to generate an arbitrarynumber of sample points in the second step. This process is not trivial, as overlapping areas may only be considered once. In our algorithm, a rejection sampling approach similar to the one in (Zamora-Sillero, 2011) is used.

In the first step, each cuboid is assigned a unique order number. For the generation of one sample point a cuboid is then randomly chosen as follows: an interval is defined between 0 and the cumulative volume of the cuboids. It is partitioned by the volumes of the ordered cuboids. Then a random number in this range is drawn and the interval it resides is determined, which then delivers the order number of the cuboid to generate a sample in. In the next step an inner random point is created. If it is also contained in cuboids that have a smaller order number, the point is rejected. Since the cuboids exist partly outside of the valid set, the sample is finally checked for validity using $\Omega(\varphi)$.

The whole process meets the desired conditions C1 – C3. On the one hand, the algorithm can be trivially adapted to support a local sampling by limiting the points to be sampled to not exceed a defined distance from a center point. On the other hand, a sampling of sub spaces can be achieved by fixation of arbitrary components of the parameter tuples during the exploration phase.

5. EVALUATION OF THE SAMPLING ALGORITHM

For the evaluation of the exploration and sampling process, test data is generated from natural images taken from the image data base in (Oliva, 2001) as shown in figure 2. They form the base for an artificial data set of 100 images that are derived from mixing the base images.

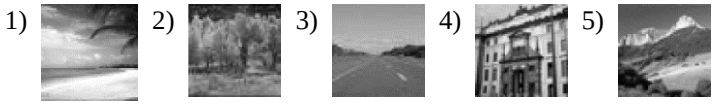

Figure 2. Base images used to generate the test data sets

For a data set $D^{(k)}$ the first k base images are used to create a ground truth base matrix U_*^k. The elements of V_*^k are initialized as uniformly distributed random numbers between 0 and 1, plus a constant additive offset to ensure a non-unique factorization, see (Rickard, 2008) for more details. The data set D^k is calculated by multiplying U_*^k and V_*^k. For each of the data sets an initial factorization U^k and V^k is calculated using the unconstrained minimization of the Euclidean distance between D^k and $U^k V^k$ as proposed in (Lee, 1999). In a next step the ground truth solution is mapped to a parameter tuple φ_*. Besides performance measurements, one goal of the evaluation is to find this ground truth tuple in the explored region of the parameter set through the proposed algorithm. Note, that this means that the ground can be found without any user interaction. The calculations were performed on a system with a Intel Xeon CPUs running at 2.67GHz. While the exploration and sampling algorithms are not optimized for parallel processing, the underlying linear algebra libraries use multiple threads for matrix calculus.

The results are shown in table 1. The column "k" denotes the number of base images in the creation of one test dataset and "n" is the dimensionality of the parameter set to be sampled given as $n = (k-1) \cdot k$. For each k, 1000 sampled parameter tuples were created in a first test and 5000 in a second test. The column "Offset" contains the additive offset applied during mixing. Every experiment was repeated 20 times to account for different results due to used random initializations. The column "Inside" contains the fraction of trials where the ground truth tuple φ_* is inside one of the created cuboids. The distance of the closest sampled tuple, which is considered to be the best, relative to the farthest sampled tuple is averaged over all trials in the column "Dist. Ratio". The column "t" holds the averaged time in seconds needed for the complete process. For the trials the maximum number of exploration iterations was set to 1000.

Table 1. Evaluation of the exploration and sampling. Left: 1000 samples, right: 5000 samples

k	n	Offset	Inside (%)	Dist. Ratio (%)	t (s)
2	2	1	100.0	1.5761	1.3
		2	100.0	0.9938	1.6
3	6	1	60.0	13.4885	8.2
		2	100.0	18.0813	7.7
4	12	1	50.0	17.9622	15.0
		2	55.0	20.8724	30.2
5	20	1	0.0	51.3582	27.4
		2	0.0	35.3822	34.1

k	n	Offset	Inside (%)	Dist. Ratio (%)	t (s)
2	2	1	100.0	0.5945	1.2
		2	100.0	0.4467	1.9
3	6	1	50.0	8.8938	7.9
		2	100.0	13.8358	6.8
4	12	1	50.0	20.0827	16.9
		2	50.0	16.5900	30.4
5	20	1	0.0	49.2524	27.1
		2	0.0	34.9876	35.6

The measured values generally indicate good results for the analysis of data sets that are factored into two to four base images. In the case of $k = 2$ the ground truth is contained in the explored part of the parameter set in all trials. For three base images, more than half of the 20 runs per configuration lead to an explored set that contains the ground truth. The average of the ratio between the closest and farthest sampled tuple lies between 10 and 20 percent which allows for (visual) identification of properties of the ground truth, depending on the data. For trials with $k > 4$, the exploration phase is mostly unable to find the area that contains the ground truth, before the maximum number of iterations was reached. However, as mentioned before, this only means that the proposed solution is not able to find it without human steering, i.e. a user that selects new center points for subsequent exploration and sampling processes.

Finally it is noteworthy that the average time needed to compute 5000 samples is only slightly higher than the time needed for the generation of 1000 samples. After the calculation of the cuboid structure is done, which comprises the majority of the overall time, the actual sampling task is comparatively fast.

6. VISUAL ASSISTANCE OF THE SAMPLING PROCESS

As mentioned in section 4 and supported through the presented evaluation, the sampling algorithm is generally not able to detect the ground truth in one run of the process for $k > 3$. This is caused by the non-convex nature of the space to be sampled. To tackle this problem, we propose a tool for visual analysis that allows subsequent runs steered by the user through variation of the seed parameter tuple and additional filters that reduce the dimensionality of the task. The main idea is to generate a sampling with a selectable number of points and cluster the samples into 10 representative solutions that are then visually presented to the user in the domain of the original data (here images) for further refinement. For clustering, we use an online version of k-means, with a modification similar to the algorithm presented in (Zhong, 2005) that prevents empty clusters.

The proposed tool utilizes the metaphor of a browser. The work flow is as follows: In the first step the user selects a data set, for example images. He then provides the rank k and triggers the initial factorization leading to the two factors U and V. Subsequently a first exploration and sampling is performed and the results are clustered. The user is then presented a view as shown in figure 3. It consists of three main parts:

1. The *navigation view* serves as a tree-like history of performed exploration, sampling and clustering runs. After the initial factorization is done, it contains one entry – the sampling of the set without user selected restrictions – which is the base for all subsequent samplings that, depending on the new starting solution to explore, sample and cluster, are visualized as child entries.
2. The *basis element* view shows the results of the sampling and clustering for all entries in the navigation view. By selecting one of the entries in the navigation view, one block in the results view is highlighted. One block consists of 10 cluster representatives, in the case of figure 3 images, where each representative is composed of three images that are arranged vertically.
3. The parameter set view shows different visualizations of the sampled parameter set. Among others, parallel coordinate plots and scatter plot matrices can be chosen. Visualizations are shown for each run independently.

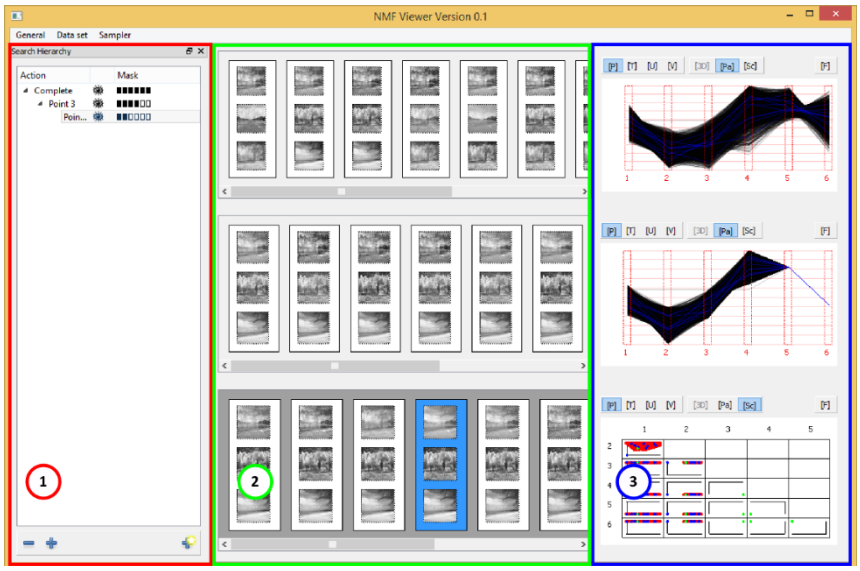

Figure 3. The browser application. (1) navigation view: from a first sampling and subsequent clustering, the 3rd solution was selected to conduct a new sampling and another sampling branches from a point of this solution. (2) visualization of representative solutions. (3) different visualizations of the sampled parameter set

To further refine the search for the desired factorization, the user has the possibility to select one of the solutions, i.e. one vertically arranged block of basis elements, optionally lock basis images that meet his criteria for subsequent runs, thus reducing the process to a lower dimensional problem. This is especially helpful if one of the elements already complies with the sought result. By selecting this element in the configuration view for subsequent explorations and samplings, the corresponding $(k-1)$ parameter values are bound to be constant and this element will stay the same in all branched processes. From the selected solution, a new exploration and sampling phase is started and presented to the user in a new row. This process is repeated until the desired solution is found.

In the case of the example in figure 3, the result of the initial unconstrained factorization shown in figure 4 (center) does not reveal the true basis as shown in figure 4 (left). After performing a first exploration, sampling and clustering, a part of the true basis elements becomes visible. The result corresponds to the first of the three blocks in figure 3 and is partially shown in figure 4 (right). As can be seen, the third basis element becomes visible. It is kept constant and a new run is initiated. The results, as shown in the second block of figure 3, show cluster representatives with the second basis image clearly visible. In a third run, only the first basis image is not bound to be constant and the results of this run are shown in the third block of figure 3. Although the first basis image was not clearly separated from the other elements, its structure is obvious.

As mentioned, the user is assisted by a set of visualizations of the sampled parameter set that help to get an intuition of the extents of the already explored set. Especially a parallel coordinate plot helps the user to get an impression of the range of found solutions in each dimension, the proximity to other solutions and to the border of the valid set. A scatter plot is

incorporated to provide visual means for the user to estimate the position of a solution in parameter space, the density of the samples and its general shape.

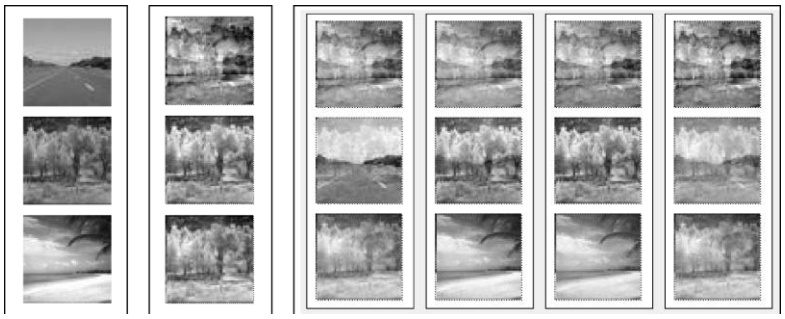

Figure 4. Exemplary run. Left: the true basis images that were used to create the test data. Center: the results after performing the unconstraint NMF. Right: 4 of the 10 representative solutions of the first exploration, sampling and clustering run.

A user study has shown that users were able to find the ground truth solution in nearly all cases for $k = 3$. Since the user base was quite small, this result is yet not representative and must be validated in future studies.

7. DISCUSSION AND CONCLUSION

The ambiguity of NMFs poses a hurdle on its usefulness for analyzing unknown data. The naïve application of this factorization technique to unknown data is prone to fail in delivering satisfying results because a local optimum will likely be found that might not correspond to the scientist's expectations. By sampling a whole set of local optima it is possible to inspect multiple solutions. Here our proposed algorithm can help greatly to increase the usefulness of NMF. The results are presented in a visual manner that does not require any further knowledge on the algorithm itself or the formulation of mathematical and algorithmic optimization constraints. The solution space is expressed by a set of transformation matrices that in turn are defined by a parameter space. Membership of the latter space is expressed in the form of an oracle that provides a thin interface to sampling approaches.

The region growing approach approximates parts of the set by simple convex bodies that are easy to sample. However, because it is applied to an unknown, non-convex set, there is no guarantee that the explored part is free of holes or contains the desired solution. Evaluations of the algorithm have shown that this is indeed a problem for higher dimensions, beginning from $k > 4$ and rendering this technique more suited for a small number of base elements. Through a simple user interface the process is steered in a desired direction that diminishes the need for a complete sampling. Again caused by the unknown shape of the solution space, it can never be completely avoided that the user of the browser gets lost during the exploration. First user studies have shown that the ground truth solution of the generated test data for small k's is found in almost all cases.

The browser component itself is kept simplistic by only showing results in the same domain as the test data, i.e. as images, which is one of the key advantages of NMF. Simple and easily understandable actions like "holding parts of a correct solution" or "narrowing the

analysis to the range of interesting solutions" are possible. This increases the chance for users without further knowledge in factorization techniques to get satisfying results from the NMF. The visual representation of the base factors are straight forward by presenting them column-wise. This visualization is well suited for a small set of base factors. As with the sampling this design becomes cluttered for factorization tasks with many basis elements. By providing simple and established visualization helpers like parallel coordinates and scatter plots a further understanding of the search space is supported. More dedicated visualization techniques could help to improve this assistance.

The mentioned algorithms provide a rich base for extensions. For example, further support in finding certain basis factors could be implemented. This ranges from simply adding more filter criteria, such as the selection of images with special features like high contrast, to semi-automatic searches steered by search criteria that can be defined interactively by the user. Other extensions include the support of other forms of data besides images, for example absorption spectra. The exploration phase of the proposed process has potential for improvements as well. The choice of averaging distances to fit the cuboids is a first step. We believe that more advanced and optimized fitting strategies can significantly speed up convergence of the exploration phase. A parallelization of the algorithm can further increase the speed.

The browser provides a simple interface to steer the sampling and is easy to use. Although first user studies were conducted, more elaborate experiments have to be carried out which would be out of the scope of this paper.

ACKNOWLEDGEMENT

This work was partially funded by European Social Fund (ESF) Project No. 100108862.

REFERENCES

Bhuiyan et al, 2012, Interactive pattern mining on hidden data: a sampling-based solution. *Proceedings of the 21st ACM international conference on Information and knowledge management*, Sheraton, USA, pp. 95 – 104.

Burkard R. et al, 2009. *Assignment Problems: Revised Reprint*. Graz University of Technology, Graz, Austria.

Chimani M. Et al, 2005. A Case Study in Large-Scale Interactive Optimization. *Artificial Intelligence and Applications*, pp. 24 – 29.

Donoho D. and Stodden V., 2003. When does non-negative matrix factorization give a correct decomposition into parts? *Advances in Neural Information Processing Systems*, Vol. 16.

Esser E. et al., 2011. A convex model for non-negative matrix factorization and dimensionality reduction on physical space. *IEEE Transactions on Image Processing*, Vol. 21, pp. 3239 – 3252.

Hoyer P.O. and Dayan P., 2004. Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research*, Vol. 5, pp. 1457 – 1469.

Inselberg, A., 2010. Parallel coordinates: Visual multidimensional geometry and its applications. *SIGSOFT Software Engineering Notes*, Vol. 35, No. 3, pp. 39-39.

Kiatsupaibul S. et al, 2011. An analysis of a variation of hit-and-run for uniform sampling from general regions. *ACM Transactions on Modeling and Computer Simulation*, Vol. 21, No. 16, pp. 1 – 11.

Kjellström, G., 1969. Network Optimization by Random Variation of Component Values. *Ericsson Technics*, Volume 25, Ch. 3, pp. 133 – 151.

Klau G.W. et al, 2010. Human-guided search. *Journal of Heuristics*, Vol. 16, No. 3, pp. 289 – 310.

Laurberg H. et al., 2008. Theorems on Positive Data: On the Uniqueness of NMF. *Computational Intelligence and Neuroscience*, Vol. 2008, pp. 1 – 9.

Lee D.D. and Seung H.S., 1999. Learning the parts of objects by nonnegative matrix factorization. *Nature*, Vol. 401, pp. 788 – 791.

Lovász, L. and Vempala, 2003, Simulated annealing in convex bodies and an $O^*(n^4)$ volume algorithm. *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science, FOCS '03.*Washington, DC, USA, pp. 650 – 659.

Malinchik S. And Bonabeau E., 2004. Exploratory Data Analysis with Interactive Evolution. *Lecture Notes in Computer Science*, Vol. 3103, pp. 1151 – 1161.

Marsaglia, G., 1972. Choosing a point from the surface of a sphere. *The Annals of Mathematical Statistics*, Vol. 43, No. 2, pp. 645 – 646.

Minc, H., 1974, *Nonnegative matrices*. Technion-Israel Institute of Technology, Dept. of Mathematics, Israel.

Oliva, A. and Torralba, A., 2001. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, Vol. 42, No. 3, pp. 145 – 175.

Plis S.M. et al, 2010, Correlated Noise: How it breaks NMF and what to do about it. *Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, Kittilä, Finland, pp. 1-9

Press W.H. and Farrar G.R., 1990. Recursive stratified sampling for multidimensional Monte Carlo integration. *In Journal on Computers in Physics*, Vol. 4, No. 2, pp. 190 – 195.

Rickard, S. and Cichocki, A., 2008, When is non-negative matrix decomposition unique? *Proceedings of the 42nd Annual Conference on Information Sciences and Systems*, Princeton, USA, pp. 1091 – 1092.

Schachtner R. Et al, 2009. Minimum determinant constraint for non-negative matrix factorization. *Lecture Notes in Computer Science*, Vol. 5441, pp. 106 – 113.

Simonovits, M., 2003, How to compute the volume in high dimension? *Mathematical Programming*, Vol. 97, No. 1-2, pp. 337 – 374.

Zamora-Sillero E. et al, 2011. Efficient characterization of high-dimensional parameter spaces for systems biology. *Bmc Systems Biology*, Vol. 5, No. 1, pp. 1 – 22.

Zhong, S., 2005, Efficient online spherical k-means clustering. *Proceedings of the IEEE International Joint Conference on Neural Networks*, New York, USA, pp. 3180 – 3185.