The Discrete Basis Problem

Pauli Miettinen[1], Taneli Mielikäinen[1], Aristides Gionis[1], Gautam Das[2,⋆], and Heikki Mannila[1]

[1] HIIT Basic Research Unit, Department of Computer Science, University of Helsinki, P.O. Box 68, FIN-00014, Finland
{pamietti,tmielika,gionis,mannila}@cs.Helsinki.FI
[2] Computer Science and Engineering Department, University of Texas at Arlington, Arlington TX 76019, USA
gdas@cse.uta.edu

Abstract. Matrix decomposition methods represent a data matrix as a product of two smaller matrices: one containing basis vectors that represent meaningful concepts in the data, and another describing how the observed data can be expressed as combinations of the basis vectors. Decomposition methods have been studied extensively, but many methods return real-valued matrices. If the original data is binary, the interpretation of the basis vectors is hard. We describe a matrix decomposition formulation, the Discrete Basis Problem. The problem seeks for a Boolean decomposition of a binary matrix, thus allowing the user to easily interpret the basis vectors. We show that the problem is computationally difficult and give a simple greedy algorithm for solving it. We present experimental results for the algorithm. The method gives intuitively appealing basis vectors. On the other hand, the continuous decomposition methods often give better reconstruction accuracies. We discuss the reasons for this behavior.

1 Introduction

Given an $n \times m$ matrix $\boldsymbol{C}$ and an integer $k < m$, classical matrix decomposition methods aim at finding an $n \times k$ matrix $\boldsymbol{S}$ and a $k \times m$ matrix $\boldsymbol{B}$ such that $\boldsymbol{C}$ can be approximately represented as the product of $\boldsymbol{S}$ and $\boldsymbol{B}$. The decomposition method represents the data by using k components: the matrix $\boldsymbol{B}$ tells how the components are related to the original attributes (columns), and the matrix $\boldsymbol{S}$ indicates how strongly each component is related to each row.

Singular value decomposition (SVD) [1] and nonnegative matrix factorization (NMF) [2] are typical examples of decomposition methods; the difference between the two is that NMF assumes that $\boldsymbol{C}$ is nonnegative and requires that $\boldsymbol{S}$ and $\boldsymbol{B}$ are nonnegative. Other matrix decomposition methods include latent Dirichlet allocation (LDA) [3] and multinomial PCA [4]; see Section 3 for additional discussion of related work.

⋆ Part of the work was done while the author visited HIIT Basic Research Unit, Department of Computer Science, University of Helsinki, Finland.

These (and other) matrix decomposition methods allow the matrices $\boldsymbol{S}$ and $\boldsymbol{B}$ to contain arbitrary real numbers. However, if the input matrix $\boldsymbol{C}$ is binary, it is natural to require that $\boldsymbol{S}$ and $\boldsymbol{B}$ are also binary. In this paper we consider the matrix decomposition problem created by this requirement. In this case the combination operation of matrices $\boldsymbol{S}$ and $\boldsymbol{B}$ is the Boolean matrix product (i.e., the matrix product in the semiring of Boolean $\wedge$ and $\vee$).

The intuition behind considering Boolean operations is as follows. Consider a course enrollment dataset in a CS department. Such a dataset indicates which students enroll to which courses. Naturally, courses are divided into specialization areas. A student X interested in the `Systems` specialization needs to take, among others, courses on {`Operating Systems, Programming languages`}, and a student Y interested in the `Software` specialization needs to take courses on {`Compilers, Programming languages`}. On the other hand, a student Z interested in combining both of the above two specializations should take all courses {`Compilers, Operating systems, Programming languages`} (among others). The point is that student Z should (obviously) take `Programming languages` only once. Thus the set union operation is more appropriate for describing the actual data from the basis vectors (specialization areas).

Following the intuition in the previous example, we formulate the problem of finding a decomposition into binary matrices that give the best approximation to the input matrix. We call this problem the *Discrete Basis Problem* (DBP). We show that DBP is NP-hard and it cannot be approximated unless P = NP.

We give a simple greedy algorithm for solving the DBP and assess its empirical performance. We show that the algorithm produces intuitively appealing basis vectors. On the other hand, the continuous decomposition methods often give better reconstruction accuracies and are also stronger in providing predictive features for classification. We discuss the reasons for this behavior.

The rest of the paper is organized as follows. In Section 2 we formally define the Discrete Basis Problem and in Section 3 we review related work. In Section 4 we compare continuous and discrete matrix decomposition approaches. In Section 5 we discuss issues related to the computational complexity of DBP. In Section 6 we present our greedy algorithm, and in Section 7 we report experimental results. Finally, Section 8 is a short conclusion.

2 Problem definition

Consider an $n \times m$ binary matrix $\boldsymbol{C}$. The rows of the matrix represent observations and the columns the attributes of the dataset. For instance, in a document corpus dataset, rows are documents and columns are words, and $\boldsymbol{C}_{ij} = 1$ denotes that the i'th document contains the j'th word.

A *basis vector*, intuitively, represents a set of correlated attributes. In the document corpus example, a basis vector corresponds to a set of words that constitute a *topic*. The DBP formulation aims at discovering the topics that are present in the dataset, and also discovering how each observation (document) in the dataset can be expressed by a combination of those topics.

Let $\boldsymbol{S}$ and $\boldsymbol{B}$ be binary matrices of dimensions $n \times k$ and $k \times m$, respectively. Let $\boldsymbol{P} = \boldsymbol{S} \circ \boldsymbol{B}$ denote the ($n \times m$ matrix) Boolean product of $\boldsymbol{S}$ and $\boldsymbol{B}$, i.e., the matrix product with addition defined by $1 + 1 = 1$. The i'th row of $\boldsymbol{P}$ is the logical OR of the rows of $\boldsymbol{B}$ for which the corresponding entry in the i'th row of $\boldsymbol{S}$ is 1. Intuitively, $\boldsymbol{S}$ is the *usage matrix*, i.e., it contains information about which topics appear in each observation, and $\boldsymbol{B}$ is the *basis vector matrix*, i.e., it contains information about which attributes appear in each topic.

The DBP seeks k binary basis vectors such that the binary data vectors can be represented by using disjunctions of the basis vectors. The key aspect of the formulation is that both decomposition matrices, $\boldsymbol{S}$ and $\boldsymbol{B}$, are required to be binary, and are thus more easily interpretable than arbitrary real matrices. Formally, the DBP is defined as follows.

Problem 1 (The Discrete Basis Problem). Given a binary $n \times m$ matrix $\boldsymbol{C}$ and a positive integer $k < \min\{m, n\}$, find an $n \times k$ binary matrix $\boldsymbol{S}$ and a $k \times m$ binary matrix $\boldsymbol{B}$ that minimize

$$|\boldsymbol{C} - \boldsymbol{S} \circ \boldsymbol{B}| = \sum_{i=1}^{n} \sum_{j=1}^{m} |c_{ij} - (\boldsymbol{S} \circ \boldsymbol{B})_{ij}| \,. \tag{1}$$

3 Related work

Probably the best-known method to decompose a matrix is the Singular Value Decomposition (SVD) [1]. The SVD decomposes a matrix $\boldsymbol{A}$ into the form $\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$, where $\boldsymbol{U}$ and $\boldsymbol{V}$ are orthonormal matrices and $\boldsymbol{\Sigma}$ is a diagonal matrix with positive entries—the singular values of $\boldsymbol{A}$. SVD gives the *optimal* rank-k approximation of the matrix $\boldsymbol{A}$ (simply by setting all but the k largest singular values to 0). Optimality of SVD means that the approximation produced by SVD is the best with respect to the squared reconstruction error and using normal matrix multiplication for real matrices. SVD has been widely used in data mining for matrix simplification and topic identification.

One problem with SVD is that the factor matrices can also contain negative values that are difficult to interpret (see also Section 4). To overcome this problem, and also to avoid the restriction to orthogonal matrices, Lee and Seung proposed a method known as non-negative matrix factorization (NMF) [2]. While NMF does not minimize the global squared reconstruction error, the existing algorithms for NMF converge to the local optima [5].

In addition to SVD and NMF, many other matrix decomposition methods have been proposed, most of which are based on probabilistic models. Such methods include multinomial PCA [4], probabilistic Latent Semantic Indexing [6], Latent Dirichlet Allocation [3], aspect Bernoulli models [7], and topic models [8]. The last two models are closest to DBP as they are designed for binary data.

Also hierarchical descriptions of binary data have been studied: the Proximus framework constructs a hierarchical clustering of rows of a given binary matrix [9] and hierarchical tiles are probabilistic models hierarchically decomposing a binary matrix into almost monochromatic 0/1 submatrices [10].

Tiling transaction databases (i.e., binary matrices) is another line of related research [11]. A tiling covers a given binary matrix with a small number of submatrices full of 1's. The main difference to DBP is that no 0's can be covered in a feasible tiling. Methods have been developed for finding also large approximate tiles, for example fault-tolerant patterns [12] and conjunctive clusters [13], but obtaining an accurate description of the whole dataset with a small number of approximate tiles has not been studied previously explicitly.

Boolean factorization, i.e., factoring Boolean functions [14], is an important part of logic synthesis. Rectangular coverings of Boolean matrices are one method used to obtain good factorizations. However, the weight functions used and the acceptance of noise are different to those of our work.

Finally, in co-clustering (or bi-clustering) the goal is to cluster simultaneously both dimensions of a matrix [15]. A co-cluster is thus a tuple (R, C), R giving the indices of rows and C giving the indices of columns. Decomposing a Boolean matrix into two matrices can be seen as a co-clustering of binary data where the clusters can overlap. Different methods for co-clustering have been proposed, see, for example, work of Banerjee, Dhillon and others [15].

4 Continuous and discrete decompositions

In this section we discuss the properties of continuous and discrete approaches to matrix decomposition, and in particular the properties of SVD as compared to those of DBP.

In SVD the resulting matrices, $\boldsymbol{U}$ and $\boldsymbol{V}$, have real-valued and even negative entries, so they do not necessarily have an intuitive interpretation. As an example, consider the case of matrix $\boldsymbol{C}$ and its rank-2 SVD decomposition:

$$\boldsymbol{C} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix}, \quad \boldsymbol{U} = \begin{pmatrix} 0.50 & 0.71 \\ 0.71 & 0 \\ 0.50 & -0.71 \end{pmatrix}, \quad \boldsymbol{\Sigma} = \begin{pmatrix} 2.41 & 0 \\ 0 & 1 \end{pmatrix}, \quad \boldsymbol{V} = \begin{pmatrix} 0.50 & 0.71 \\ 0.71 & 0 \\ 0.50 & -0.71 \end{pmatrix}.$$

The basis vectors in $\boldsymbol{V}$ are not the easiest to interpret. Matrix $\boldsymbol{C}$ has rank 3, and the approximation to $\boldsymbol{C}$ produced by SVD with rank-2 decomposition is

$$\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T = \begin{pmatrix} 1.10 & 0.85 & 0.10 \\ 0.85 & 1.21 & 0.85 \\ 0.10 & 0.85 & 1.10 \end{pmatrix}.$$

By the optimality of SVD, this is the best that can be achieved by looking at real matrices and squared error. On the other hand, DBP produces the representation

$$\boldsymbol{C} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{pmatrix} \circ \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix},$$

which has no error and is easy to understand.

As noted above, SVD produces optimal rank-k representations of matrices with respect to the Frobenius norm (sum of squares of elements). It is also relatively fast to compute, requiring time $O(nm \min\{n, m\})$ [1].

Optimality for arbitrary matrices is not the whole story, however. For binary matrices, one can study two types of ranks. The *real rank* $r_R(\boldsymbol{C})$ of a binary matrix $\boldsymbol{C}$ is simply the smallest value of k such that $\boldsymbol{C} = \boldsymbol{S}\boldsymbol{B}$ with an $n \times k$ matrix $\boldsymbol{S}$, a $k \times m$ matrix $\boldsymbol{B}$, and using normal matrix multiplication. The *Boolean rank* $r_B(\boldsymbol{C})$ of $\boldsymbol{C}$ is the smallest k such that $\boldsymbol{C} = \boldsymbol{S} \circ \boldsymbol{B}$, where $\boldsymbol{S}$ is an $n \times k$ matrix, $\boldsymbol{B}$ is a $k \times m$ matrix, and the matrix multiplication is Boolean. It can be shown that there are matrices $\boldsymbol{C}$ for which $r_R(\boldsymbol{C}) < r_B(\boldsymbol{C})$ and vice versa [16]. The complement of the identity matrix of size $n \times n$ is an example where $r_B(\boldsymbol{C}) = O(\log n)$, but $r_R(\boldsymbol{C}) = n$ [16]. This shows that while SVD can use the space of reals, DBP can take advantage of the properties of Boolean operations to achieve much smaller rank than SVD. Empirical experiments on generated data support this conclusion. Thus it is not a priori obvious that SVD will produce more concise representations than the Boolean methods.

The concepts of real and Boolean rank discuss the exact representation of the matrix $\boldsymbol{C}$, and we are more interested in the approximate representation. One could define the ε-ranks $r_R^\varepsilon(\boldsymbol{C})$ and $r_B^\varepsilon(\boldsymbol{C})$ as the smallest k such that there is a representation of $\boldsymbol{C}$ as $\boldsymbol{S}\boldsymbol{B}$ or $\boldsymbol{S} \circ \boldsymbol{B}$ with $\boldsymbol{B}$ being a $k \times m$ matrix and $|\boldsymbol{C} - \boldsymbol{S}\boldsymbol{B}| < \varepsilon$. Even less seems to be known about such concepts than about exact real and Boolean ranks. One goal of our paper is to investigate empirically and theoretically whether the Boolean decompositions are feasible alternatives of the continuous methods.

5 Computational Complexity of DBP

The DBP is an optimization problem: find the matrix decomposition into k basis vectors that minimizes the representation error according to the definition of Problem 1. To put the problem in the perspective of complexity theory, we formulate the decision version of the problem. This is defined as in Problem 1, but additionally we are given a target cost t and the task is to decide whether there is a decomposition of the input binary matrix $\boldsymbol{C}$ into binary matrices $\boldsymbol{S}$ and $\boldsymbol{B}$ that yields an error at most equal to t.

The problem is NP-hard as the Set Basis Problem (SBP) [17, problem SP7] is a special case of the decision version of the DBP.

Problem 2 (The Set Basis Problem). Given a collection $\mathcal{C}$ of subsets of a finite universe U and a positive integer k, decide whether or not there is a collection $\mathcal{B} \subseteq 2^U$ of at most k sets ($|\mathcal{B}| \leq k$) such that for every set $C \in \mathcal{C}$ there is a subcollection $\mathcal{B}_C \subseteq \mathcal{B}$ with $\bigcup_{B \in \mathcal{B}_C} B = C$.

More specifically, for any instance of SBP there is an equivalent instance of DBP with $t = 0$, even when only the matrix $\boldsymbol{B}$ is requested. The NP-hardness can also be shown by observing that the Biclique Covering Problem is a special case of DBP with $t = 0$ where both $\boldsymbol{S}$ and $\boldsymbol{B}$ are needed. It is immediate that DBP is in NP. Thus we have:

Theorem 1. *The decision version of* DBP *is* NP*-complete.*

The reduction from SBP to DBP with $t = 0$ implies also the following simple inapproximability result:

Theorem 2. *DBP cannot be approximated within any factor in polynomial time, unless* P = NP*.*

The problem of solving the whole decomposition of the matrix for given basis vectors, i.e., finding the matrix $\boldsymbol{S}$ for given $\boldsymbol{B}$ and $\boldsymbol{C}$, can be solved by a straightforward algorithm in time $O(2^k mn)$ where k is the number of basis vectors (i.e., the number of rows in $\boldsymbol{B}$): Each of the n rows in $\boldsymbol{C}$ can be decomposed independently and there are only 2^k different ways to choose a subset of basis vectors. Thus the problem of finding the optimal decomposition after the basis vector matrix is known, is in the class of *fixed-parameter tractable* problems (see [18]).

6 The Algorithm

In this section we give a greedy algorithm for DBP. The basic idea of the algorithm is to exploit the correlations between the columns. First, the associations between each two columns are computed. Second, the associations are used to form candidate basis vectors. Third, a small set of candidate basis vectors are selected in a greedy way to form the basis.

In the rest of the section we denote a row vector of a matrix $\boldsymbol{M}$ by $\boldsymbol{m}_i$, a column vector by $\boldsymbol{m}_{\cdot i}$ and a matrix entry by m_{ij}. The confidence of an association between the i-th and j-th column is defined as in association rule mining [19], i.e., $c(i \Rightarrow j) = \langle \boldsymbol{c}_{\cdot i}, \boldsymbol{c}_{\cdot j} \rangle / \langle \boldsymbol{c}_{\cdot i}, \boldsymbol{c}_{\cdot i} \rangle$, where $\langle \cdot, \cdot \rangle$ is the vector inner product operation. An association between columns i and j is τ-strong if $c(i \Rightarrow j) \geq \tau$.

We construct an *association matrix* $\boldsymbol{A}$ where row $\boldsymbol{a}_i$ consists of 1's in columns j such that $c(i \Rightarrow j) \geq \tau$. Each row of $\boldsymbol{A}$ is considered as a candidate for being a basis vector. The threshold τ controls the level of confidence required to include an attribute to the basis vector candidate, and it is assumed that τ is a parameter of the algorithm.

The DBP objective function, described by (1), penalizes equally for both types of errors: for 0 becoming 1 in the approximation, and for 1 becoming 0. We have found that in practice the results of DBP can be improved if we distinguish between these two types of error. Thus we introduce weights w^+ and w^- that are used to reward for covering 1's and penalize for covering 0's, respectively. Clearly, without loss of generality, we can assume that $w^- = 1$.

The basis vectors are selected from the matrix $\boldsymbol{A}$ and the columns of the usage matrix $\boldsymbol{S}$ are fixed in a greedy manner as follows. Initially $\boldsymbol{B} = 0^{k \times m}$ and $\boldsymbol{S} = 0^{n \times k}$. The basis $\boldsymbol{B}$ is updated in the iteration l by setting the row $\boldsymbol{b}_l$ to be the row $\boldsymbol{a}_i$ in $\boldsymbol{A}$ and the column $\boldsymbol{s}_{\cdot l}$ to be the binary vector maximizing

$$\begin{aligned} \mathit{cover}(\boldsymbol{B}, \boldsymbol{S}, \boldsymbol{C}, w^+, w^-) = {} & w^+ |\{(i,j) : c_{ij} = 1, (\boldsymbol{S} \circ \boldsymbol{B})_{ij} = 1\}| \\ & - w^- |\{(i,j) : c_{ij} = 0, (\boldsymbol{S} \circ \boldsymbol{B})_{ij} = 1\}|, \end{aligned}$$

Algorithm 1 An algorithm for the DBP using association rules

Input: Matrix $\boldsymbol{C} \in \{0,1\}^{n \times m}$ for data, positive integer $k < \min\{n, m\}$, threshold value $\tau \in]0, 1]$, and real-valued weights w^+ and w^-.
Output: Matrices $\boldsymbol{B} \in \{0,1\}^{k \times m}$ and $\boldsymbol{S} \in \{0,1\}^{n \times k}$.

```
1: function ASSOCIATION(C, k, τ, w+, w−)
2:     for i = 1, ..., n do          ▷ Construct the association matrix A row by row.
3:         a_i ← {j : c(i ⇒ j) ≥ τ}
4:     B ← 0^{k×m}
5:     for l = 1, ..., k do                    ▷ Select the k basis vectors from A.
6:         b_l ← a_i and s_{·l} ← {0,1}^n maximizing cover(B, S, C, w+, w−)
7:     return B and S
```

which can be considered as the "profit" of describing $\boldsymbol{C}$ using the basis $\boldsymbol{B}$ and the decomposition $\boldsymbol{S}$.

The association matrix can be constructed in time $O(nm^2)$ and a single discrete basis vector can be obtained in time $O(nm^2)$. Thus, Algorithm 1 has time complexity $O(knm^2)$. The run-time can be improved in practice by using upper bounds and approximations for the confidences.

The algorithm has two parameters that control the quality of results: the threshold τ, and weight w^+ (again assuming that $w^- = 1$). The straightforward way to set the parameters is to try several different possibilities and take the best. Alternatively the weight w^+ can be used to express different valuations for covering 1's and 0's.

Unfortunately there exist cases, where the algorithm is able to find only suboptimal solution. For example, if all 1's in some basis vector occur in some other basis vectors, then the algorithm is unable to find that basis vector.

7 Experimental results

We have performed tests using Algorithm 1 on generated and real-world datasets. The goals of the experiments are (*i*) to verify whether DBP produces intuitive basis vectors, (*ii*) to check whether DBP can reconstruct basis vectors used to generate artificial data, and (*iii*) to compare the reconstruction accuracy of DBP against SVD and NMF both for real and generated data.

7.1 Data and error measures

Generated data. We generated three sets of data to test the effects of (*i*) noise, (*ii*) overlap between basis vectors, and (*iii*) input size. First, a set of basis vectors was generated; then random subsets of these basis vectors were used to generate the data rows; finally, random uniform noise was added. Details on the parameters used to generate the three sets of data are shown in Table 1.
Real data. The real data consists of the following datasets: NSF Abstracts, 20 Newsgroups, Digits, and Courses. Details of the datasets are given in Table 2.

Table 1. Details on generated datasets. Each row represents a set of generated datasets. #bv: number of basis vectors; #bv/row: average number of basis vectors used to generate each data row; 1s/bv: number of 1's per basis vector; noise: number of entries flipped in the data as a percentage of the total number of 1's.

dataset	rows	columns	#bv	#bv/row	1s/bv	noise
set 1	1000	500	12	4	50	5–40%
set 2	1000	500	12	4, 8	25–200	0%
set 3	1K–16K	1K–16K	10–160	5–80	0.5K–80K	0%

NSF Abstracts[3] contain document–word information on a collection of project abstracts submitted for funding by NSF. 20 Newsgroups[4] is a collection of approximately 20000 newsgroup documents across 20 different newsgroups [20]. Digits[5] is a collection of 1000 binary images of handwritten digits [21]. Courses is a student–course dataset of courses completed by the CS students of the University of Helsinki. A random sample of NSF Abstracts and 20 Newsgroups was used for comparisons between different algorithms due to memory constraints of SVD and NMF implementations.

Table 2. Attributes of the real-world datasets.

dataset	rows	columns	1s in data	avg. 1s/row	avg. 1s/column
NSF Abstracts	12 841	4 894	564 462	43.96	115.34
20 Newsgroups	10 000	5 163	455 526	45.55	88.23
Digits	2 000	240	291 654	145.83	1 215.23
Courses	2 405	615	52 739	21.92	85.75

Error measures. We use two measures to quantify the error of the approximation: sum-of-absolute-values distance d_1 and Frobenius distance d_2, defined as

$$d_1(\boldsymbol{A}, \boldsymbol{B}) = \sum_i \sum_j |a_{ij} - b_{ij}| \quad \text{and} \quad d_2(\boldsymbol{A}, \boldsymbol{B}) = \sqrt{\sum_i \sum_j (a_{ij} - b_{ij})^2}.$$

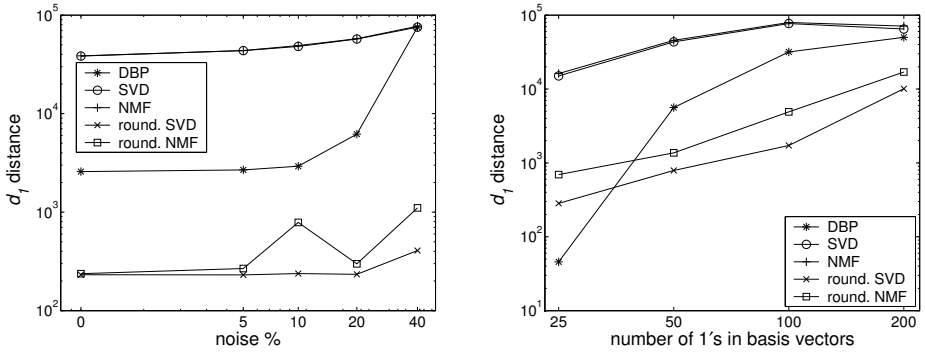
7.2 Results

Reconstructing the basis vectors from generated data. We studied the effects of noise and overlap between basis vectors to the reconstruction error. The main measure was the d_1 distance. Algorithm 1 was compared against SVD and NMF. For SVD and NMF we also used the knowledge that the matrix is supposed to be binary, and rounded the reconstructed matrix *before* computing the error with

[3] http://kdd.ics.uci.edu/databases/nsfabs/nsfawards.html
[4] http://people.csail.mit.edu/jrennie/20Newsgroups/
[5] http://www.ics.uci.edu/∼mlearn/databases/optdigits/

respect to the original matrix. Values smaller than 0.5 are rounded to 0 and values greater than 0.5 are rounded to 1. We call these methods *rounded* SVD and *rounded* NMF.

The effects of noise are illustrated in Figure 1(a). Lines for plain SVD and NMF coincide at the top of the figure, partly because of the logarithmic scale. In general, plain SVD and NMF are the worst, rounded SVD and NMF are the best, and DBP is in between. Additionally, all methods seem to be rather immune to small amounts of noise. With one input set, rounded NMF converged far from global optimum, thus causing a peak in the graph.



(a) (b)

Fig. 1. Reconstruction errors using d_1 as a function of (a) noise (dataset 1), and (b) 1's in basis vector (dataset 2). Points in plots represent the mean error over five random data matrices with the same attributes. Logarithmic scale on both axes of both plots.

Figure 1(b) illustrates the effects of basis vector's overlap. The expected overlap of two random vectors is uniquely defined by the number of 1's in basis vectors, i.e., the values in x-axis of Figure 1(b). If basis vectors have high overlap, it becomes harder to distinguish the basis vectors using association confidences. Thus higher overlap degrades the quality of DBP results, as one can clearly see from Figure 1(b). On the other hand, rounded SVD and NMF seem to have more problems on reconstructing data with high overlap in basis vectors. In this experiment, rounded SVD and NMF are again the best, plain SVD and NMF the worst, and DBP is in between. The only exception is the first point, in which the DBP is the best. The explanation is that with small overlap, Algorithm 1 is very effective.

Reconstruction errors for real data. Reconstruction errors for the real datasets are given in Tables 3 (d_1 distance) and 4 (d_2 distance).

We used two additional methods in these experiments, namely *0–1* SVD and *0–1* NMF. The idea is to make binary the factor matrices of SVD and NMF and

multiply them using Boolean algebra. However, it is far from obvious how to binarize the factor matrices. We used again a threshold approach: values below the threshold are rounded to 0 and values above the threshold are rounded to 1. To be fair, we used a brute-force search to select the optimal thresholds for both factor matrices (different threshold for each matrix).

Table 3 shows that in d_1 DBP is comparable to other methods, including rounded SVD. For example, in 20 Newsgroups and NSF Abstracts with $k = 5$ DBP gives the smallest error. While DBP cannot beat SVD or NMF in d_2 (Table 4), it is not too far away from them in most of the cases. The 0–1 SVD and 0–1 NMF have the largest reconstruction error.

Table 3. Reconstruction error of real-world datasets using d_1 distance. Values are scaled and truncated to three decimals.

			algorithm						
dataset	k	scale	SVD	NMF	r. SVD	r. NMF	0–1 SVD	0–1 NMF	DBP
NSF Abstr.	5	10^6	1.124	1.089	0.563	0.563	5.171	3.328	0.559
NSF Abstr.	10		1.152	1.091	0.561	0.561	6.065	5.890	0.554
NSF Abstr.	20		1.197	1.099	0.555	0.556	9.605	10.228	0.545
20 Newsgr.	5	10^6	0.900	0.875	0.450	0.450	4.185	2.612	0.449
20 Newsgr.	10		0.928	0.881	0.446	0.447	4.950	4.447	0.446
20 Newsgr.	20		0.969	0.889	0.440	0.441	7.293	8.096	0.441
Digits	5	10^5	1.308	1.382	0.763	0.855	1.678	1.113	2.133
Digits	10		1.070	1.206	0.471	0.610	1.817	0.967	2.125
Digits	20		0.878	1.028	0.254	0.444	1.678	0.815	2.119
Courses	5	10^4	6.467	6.204	3.215	3.350	8.202	6.418	3.783
Courses	10		6.164	5.779	2.770	2.951	14.051	9.840	3.515
Courses	20		5.949	5.186	2.219	2.495	20.490	17.021	3.160

Empirical time complexity. Set 3 was used to verify the empirical time complexity of the algorithm. The results obtained agreed with theoretical complexity results perfectly, i.e., the running time of Algorithm 1 increased linearly with the number of rows in data and with the size of the basis, and quadratically with the number of columns in data.

Quality of basis vectors for real data. We used the NSF Abstracts dataset to examine the quality of the DBP basis vectors. We used $\tau = 0.3$ and $w^+ = 6$ as the set of parameters that gave the most intuitive results. Examples of basis vectors and representative words are as follows.

`<fund, NSF, year>`,
`<cell, gene, molecular, protein>`,
`<gopher, internet, network, world, wide, web>`,
`<behavior, effect, estim, impact, measure, model, overestimate, predict, test>`, and

Table 4. Reconstruction error of real-world datasets using d_2 distance. Values are rounded to nearest the integer.

		algorithm						
dataset	k	SVD	NMF	r. SVD	r. NMF	0–1 SVD	0–1 NMF	DBP
NSF Abstr.	5	727	728	751	750	2 274	1 825	748
NSF Abstr.	10	719	721	749	749	2 463	2 427	745
NSF Abstr.	20	709	713	745	746	3 099	3 198	738
20 Newsgr.	5	649	650	671	672	2 046	1 616	671
20 Newsgr.	10	643	644	668	669	2 225	2 109	668
20 Newsgr.	20	634	637	664	665	2 701	2 845	665
Digits	5	239	248	276	293	410	334	462
Digits	10	201	221	217	247	426	311	461
Digits	20	168	196	159	211	410	286	460
Courses	5	165	167	179	183	286	253	195
Courses	10	154	158	166	172	375	314	188
Courses	20	141	147	149	158	453	413	178

`<course, education, enrol, faculty, institute, school, student, undergraduate>`.

8 Discussion and conclusions

We have described the Discrete Basis Problem, investigated its computational complexity, given a simple algorithm for it, and have shown empirical results on the behavior of the algorithm. The results indicate that the algorithm discovers intuitively useful basis vectors. In generated data, the method can reconstruct the basis vectors that were used to generate the data; this holds even with high amounts of noise.

On the other hand, in many cases, SVD has lower reconstruction error than DBP. There are several possible reasons for this. The first possibility is that SVD is in some sense inherently more powerful than DBP. This is of course vaguely expressed. While we know that SVD is optimal with respect to the Frobenius norm, we also know that the Boolean rank of a matrix can be much smaller than its real rank. SVD in some ways has more power than DBP, as SVD works on the continuous values; on the other hand, DBP can take advantage of the Boolean semiring on which it operates. This suggests that the relative performance of DBP against SVD should improve as the overlap between basis vectors increases.

The second alternative reason for the good performance of SVD is that the DBP algorithm is suboptimal. This suboptimality certainly degrades the results: for example, overlap between the basis vectors makes them harder to be discovered. However, for our generated data, in many cases, the DBP algorithm reconstructs the original basis vectors perfectly. Thus, at least for those data sets the algorithm is sufficiently good.

We have shown that Boolean approaches to matrix decomposition form a viable alternative for traditional methods. For further work, it would be of interest to understand the relationship between the approximate Boolean and real ranks of binary matrices better. Also, a more detailed comparison of DBP against the probabilistic approaches such as LDA and multinomial PCA would be useful.

References

1. Golub, G., Van Loan, C.: Matrix Computations. JHU Press (1996)
2. Lee, D., Seung, H.: Learning the parts of objects by Non-negative Matrix Factorization. Nature **401** (1999) 788–791
3. Blei, D., Ng, A., Jordan, M.: Latent Dirichlet Allocation. Journal of Machine Learning Research **3** (2003) 993–1022
4. Buntine, W.: Variational extensions to EM and multinomial PCA. In: ECML. (2002)
5. Lee, D., Seung, H.: Algorithms for Non-negative Matrix Factorization. Advances in Neural Information Processing Systems **13** (2001) 556–562
6. Hofmann, T.: Probabilistic Latent Semantic Indexing. In: ACM Conference on Research and Development in Information Retrieval. (1999) 50–57
7. Kabán, A., Bingham, E., Hirsimäki, T.: Learning to read between the lines: The aspect Bernoulli model. In: ICDM. (2004)
8. Seppänen, J., Bingham, E., Mannila, H.: A simple algorithm for topic identification in 0–1 data. In: PKDD. (2003)
9. Koyutürk, M., Grama, A., Ramakrsihnan, N.: Compression, clustering, and pattern discovery in very-high-dimensional discrete-attribute data sets. IEEE Transactions on Knowledge and Data Engineering (2005) 447–461
10. Gionis, A., Mannila, H., Seppänen, J.K.: Geometric and combinatorial tiles in 0-1 data. In: PKDD. (2004)
11. Geerts, F., Goethals, B., Mielikäinen, T.: Tiling databases. In: Discovery Science: 7th International Conference, Proceedings. (2004)
12. Besson, J., Pensa, R., Robardet, C., Boulicaut, J.F.: Constraint-based mining of fault-tolerant patterns from boolean data. In: KDID. (2006)
13. Mishra, N., Ron, D., Swaminathan, R.: On finding large conjunctive clusters. In: COLT. (2003)
14. Brayton, R.K., Hachtel, G.D., Sangiovanni-Vincentelli, A.L.: Multilevel logic synthesis. Proceedings of the IEEE **78**(2) (1990) 264–300
15. Banerjee, A., et al.: A generalized maximum entropy approach to Bregman coclustering and matrix approximations. In: KDD. (2004) 509–514
16. Monson, S.D., Pullman, N.J., Rees, R.: A survey of clique and biclique coverings and factorizations of $(0, 1)$-matrices. Bulletin of the ICA **14** (1995) 17–86
17. Garey, M.R., Johnson, D.S.: Computers and intractability: A guide to the theory of NP-Completeness. W. H. Freeman & Co., New York (1979)
18. Downey, R.G., Fellows, M.R.: Parameterized Complexity. Monographs in computer science. Springer-Verlag New York (1999)
19. Agrawal, R., Imielinski, T., Swami, A.: Mining association rules between sets of items in large databases. In: SIGMOD. (1993)
20. Lang, K.: Newsweeder: Learning to filter netnews. In: Proceedings of the Twelfth International Conference on Machine Learning. (1995) 331–339
21. Newman, D., Hettich, S., Blake, C., Merz, C.: UCI repository of machine learning databases (1998)