SPARSE MATRIX DECOMPOSITIONS FOR CLUSTERING

Thomas Blumensath

ISVR Signal Processing and Control Group
The University of Southampton
Southampton, SO17 1BJ

ABSTRACT

Clustering can be understood as a matrix decomposition problem, where a feature vector matrix is represented as a product of two matrices, a matrix of cluster centres and a matrix with sparse columns, where each column assigns individual features to one of the cluster centres. This matrix factorisation is the basis of classical clustering methods, such as those based on non-negative matrix factorisation but can also be derived for other methods, such as k-means clustering. In this paper we derive a new clustering method that combines some aspects of both, non-negative matrix factorisation and k-means clustering. We demonstrate empirically that the new approach outperforms other methods on a host of examples.

***Index Terms*—** Clustering, Low-Rank Matrix Approximation, Sparsity, Brain Imaging

1. INTRODUCTION

Clustering is the process of grouping a set of objects (represented typically by a set of feature vectors) into distinct classes. This is often done through the construction of a data model for each feature. In an unsupervised setting, where the model as well as the cluster assignments have to be learned, a fundamental problem is the fact that it is difficult to construct a model for a cluster without knowledge of cluster assignment whilst cluster assignment cannot proceed without the cluster models. Many approaches are therefore iterative. Typical example include Expectation Maximisation algorithms [1] and the k-means algorithm [2].

An alternative approach is based on matrix decompositions such as non-negative matrix factorisations (NMF) or semi-non-negative matrix factorisations (semi-NMF) [3]. Methods such as semi-NMF use two steps which are *not* iterated. First, feature vectors are modelled as sums of a small number of feature vectors, where additional constrained such as non-negativity are used to guide this decomposition. A second step then provides hard cluster assignment, typically assuming that the model features are cluster centres.

Our approach will start from a data model similar to that used in NMF, but instead of using the non-negativity constraint, we iteratively enforce hard cluster assignment. Let there be N objects that are to be clustered. Each object is represented through a feature vector $\mathbf{x}_i$. Let us stack the feature vectors (treated as column vectors) into a matrix $\mathbf{X}$. As a model for the clusters assume we have K cluster centre vectors $\mathbf{d}_k$, which we also stack into a matrix $\mathbf{D}$. Let there be errors $\mathbf{e}_i$ which are the columns of a matrix $\mathbf{E}$. With this notation, we can write the model as

$$\mathbf{X} = \mathbf{D}\mathbf{S} + \mathbf{E}, \tag{1}$$

where $\mathbf{S}$ is a coefficient matrix. If we want each cluster to be modelled as a perturbation of a single cluster centre, then the matrix $\mathbf{S}$ has to have columns that are 1-sparse, that is, each column contains a single non-zero element. In fact, the non-zero elements are restricted to be 1.

If we were to relax the constraint on the non-zero entries and replace it with a positivity constraint, then features are modelled as lying in the direction of the cluster centre. This provides us with an opportunity to cluster feature vectors that are only known up to a scale factor. This *directional* clustering occurs, for example, in text classification [4] and functional brain imaging [5].

2. CLUSTERING AS SPARSE MATRIX FACTORISATION

Whilst semi-NMF based clustering approaches calculate a matrix decomposition first and then apply a singe cluster assignment step, k-means iterates between cluster centre estimation and cluster assignment. Our approach that takes aspects from both of these methods. Inspired by the Iterative Hard Thresholding algorithm proposed for sparse approximation [6] we try to find a low-rank approximation to the feature vector matrix $\mathbf{X}$ that simultaneously enforce hard cluster assignment. This is done using the following iterative scheme, where clustering is performed through a hard thresholding step that only keeps the largest non-zero coefficient in each column of the coefficient matrix $\mathbf{S}$.

1. INPUT: data matrix $\mathbf{X}$, number of clusters K
2. initial decomposition of $\mathbf{X}$ into low rank factorisation (e.g. using an SVD or some initial cluster assignment)

This research was supported by EPSRC grant EP/K037102/1.

$\mathbf{X} = \mathbf{D}\mathbf{S} + \mathbf{E}$.

3. iterate until some convergence criterion is met
   (a) Calculate cluster assignment: $\bar{\mathbf{S}} = P(\mathbf{S})$
   (b) Check for empty clusters and randomly re-initialise (see below)
   (c) Update cluster centres: $\mathbf{D} = \mathbf{X}\bar{\mathbf{S}}^T(\bar{\mathbf{S}}\bar{\mathbf{S}}^T)^{-1}$
   (d) Update cluster weights: $\mathbf{S} = \mathbf{S} + \mathbf{D}^T(\mathbf{X} - \mathbf{D}\mathbf{S})$ or $\mathbf{S} = (\mathbf{D}^T\mathbf{D})^{-1}\mathbf{D}^T\mathbf{X}$

Here, the non-linear map $P(\cdot)$ is the hard thresholding step that sets all but the largest element in each column of $\mathbf{S}$ to zero.

2.1. Catching empty clusters

Step 3.b in the algorithm catches the problem that can arise when clusters are empty after thresholding. If there are empty clusters, then the matrix $\mathbf{S}\mathbf{S}^T$ will have a zero diagonal entry and is thus no longer invertible. In this case, the empty cluster is re-initialised by randomly assigning a feature from another cluster to the empty cluster.

2.2. Computational considerations

The following computational points are worth noting.

1. The matrix $\mathbf{D}^T\mathbf{D}$ is a K by K matrix so that for small K, the inverse in step (d) is easy to evaluate.
2. Because after cluster assignment, the matrix $\bar{\mathbf{S}}$ will have one-sparse columns so that the matrix $\bar{\mathbf{S}}\bar{\mathbf{S}}^T$ is diagonal and thus will be easy to invert.

It is also important to point out that there are scale ambiguities in the model. Left multiplication of $\bar{\mathbf{S}}$ by a diagonal matrix and simultaneous right division of $\mathbf{D}$ by the same matrix will not change the product $\mathbf{D}\bar{\mathbf{S}}$. It is thus customary to normalise the columns in $\mathbf{D}$ or rows in $\mathbf{S}$. For standard clustering, this normalisation is done automatically, as non-zero entries in $\bar{\mathbf{S}}$ are set to 1, however, for the directional clustering alternative, the scaling issue becomes more important. Two possibilities are to normalise columns in $\mathbf{D}$ or to normalise rows in $\mathbf{S}$ either before or after cluster assignment assignment.

For the directional clustering problem, another issue is that, in order for errors to be comparable between clusters, it is advisable to normalise the feature vectors $\mathbf{x}_i$ prior to clustering.

Finally, let us point out that it is also possible to estimate the number of clusters. This can be done, for example, by an iterative approach that tries to optimise a cost function that includes a model complexity term (e.g. the Akaike information criterion). This can be done efficiently using a line search [7].

3. THEORETICAL ANALYSIS

As with any iterative computational method, several theoretical questions arise. These include a characterisation of the minima of the associated cost function, an analysis of the fixed points of the algorithm, convergence properties and the distance of the solution from the global optimum.

We here report some initial results that go towards answering some of these points[1]. As there are several variations of the algorithm, we here restrict the analysis to the following scheme for directional clustering. We assume that we normalise the columns in $\mathbf{X}$ to unit length. Let us use the update recursion $\mathbf{S}_{n+1} = \mathbf{S}_n + \mathbf{D}_n^T(\mathbf{X} - \mathbf{D}_n\mathbf{S}_n)$, $\bar{\mathbf{S}}_{n+1} = P(\mathbf{S}_{n+1})$, and assume we normalise $\mathbf{D}_{n+1}$ and $\bar{\mathbf{S}}_{n+1}$ after each update of $\mathbf{D}_{n+1}$. For notational convenience, we write $\bar{\mathbf{S}}_{n+1/2} = P(\mathbf{S}_{n+1})$ and $\mathbf{D}_{n+1/2} = \mathbf{X}\bar{\mathbf{S}}_{n+1}^T(\bar{\mathbf{S}}_{n+1}\bar{\mathbf{S}}_{n+1}^T)^{-1}$ and let $\bar{\mathbf{S}}_{n+1}$ and $\mathbf{D}_{n+1}$ be the rescaled versions, such that $\mathbf{D}_{n+1}$ has unit norm columns and such that $\mathbf{D}_{n+1}\bar{\mathbf{S}}_{n+1} = \mathbf{D}_{n+1/2}\bar{\mathbf{S}}_{n+1/2}$.

3.1. Cost function minima

For a given $\bar{\mathbf{S}}$, let $\mathbf{X}_i$ be the sub-matrix of $\mathbf{X}$ containing only those features that are in cluster i and let $\mathbf{s}_i$ be the sub-vector of the i^{th} row of the matrix $\bar{\mathbf{S}}$, containing only non-zero entries. It can be shown that

Lemma 1. *The global minima of the constrained clustering cost function $\|\mathbf{X} - \mathbf{D}\mathbf{S}\|_F^2$ is achieved for $\mathbf{S}$ with row-subvectors $\mathbf{s}_i$ that are right singular vectors of $\mathbf{X}_i$, where the $\mathbf{X}_i$ are non-empty sub-matrices of $\mathbf{X}$, such that each column in $\mathbf{X}$ is in exactly one sub-matrix.*

3.2. Fixed points

Fixed points of the algorithm are those points that satisfy the recursion.

$$\mathbf{\Phi}\bar{\mathbf{S}} = P(\bar{\mathbf{S}} + ((\bar{\mathbf{S}}\bar{\mathbf{S}}^T)^{-1}\bar{\mathbf{S}}\mathbf{X}^T\,(\mathbf{X} - \mathbf{X}\bar{\mathbf{S}}^T(\bar{\mathbf{S}}\bar{\mathbf{S}}^T)^{-1}\bar{\mathbf{S}}))), \quad (2)$$

where $\mathbf{\Phi}$ is a diagonal matrix (a function of $\bar{\mathbf{S}}$)[2] that normalises the columns of the matrix $\mathbf{X}\bar{\mathbf{S}}^T(\bar{\mathbf{S}}\bar{\mathbf{S}}^T)^{-1}$.

With this fixed point condition and the notation above, it can be shown that

Lemma 2. *The stationary points of the algorithm provide a partition of the data set such that the non-zero elements in $\bar{\mathbf{S}}$ associated with cluster i are either orthogonal to the rows in $\mathbf{X}_i$ or are eigenvectors of the matrix $\mathbf{X}_i^T\mathbf{X}_i$.*

3.3. Towards a convergence analysis

To analyse the asymptotic behaviour of the algorithm, we can derive the following lemma.

[1]Proofs will be omitted here due to space constraints.
[2]Note that $\bar{\mathbf{S}}\bar{\mathbf{S}}^T$ is diagonal and so is $\mathbf{\Phi}$.

Lemma 3. *There exist an $\mathbf{X}^\star$ and an infinite subset of indices n_i such that for all ϵ, we can choose an $N_\epsilon < \infty$ such that*

$$\|\mathbf{X}^\star - \mathbf{D}_{n_i}\bar{\mathbf{S}}_{n_i}\| \le \epsilon, \tag{3}$$

hold for all $n_i > N_\epsilon$.

Lemma 4. *The matrix factorisation algorithm produces a sequence of estimates $\bar{\mathbf{S}}_n$ that satisfy:*

$$\|\bar{\mathbf{S}}_{n+1/2} - \bar{\mathbf{S}}_n\|^2 \to 0. \tag{4}$$

Lemma 5. *Assume that the matrices $(\bar{\mathbf{S}}_n\bar{\mathbf{S}}_n^T)^{-1}\bar{\mathbf{S}}_n$ are bounded. The matrix factorisation algorithm then produces a sequence of estimates $\mathbf{D}_n\bar{\mathbf{S}}_n$ that satisfy:*

$$\|\mathbf{D}_{n+1}\bar{\mathbf{S}}_{n+1} - \mathbf{D}_n\bar{\mathbf{S}}_n\|^2 \to 0. \tag{5}$$

Whilst these do not directly imply that the algorithm converges, due to the fact that clustering is based on selecting the largest element in each columns of $\mathbf{S}$, lemma 4 implies that the non-zero elements in $\bar{\mathbf{S}}_n$ either converge themselves to zero or, that the location of the non-zero entries in the matrix $\bar{\mathbf{S}}$ does not change after some iteration.

4. EMPIRICAL RESULTS

We here again concentrate on directional clustering examples. The performance of our new approach is evaluated using Normalised Mutual Information (NMI) [8]. We tried a range of other measures, all of which gave comparable results. Results are contrasted with those obtained by k-means clustering [2] and semi non-negative matrix factorisation (semi-NMF) [3]. We used two different versions of k-means, standard k-means clusters based on euclidean distance, whist spherical k-means makes cluster assignments based on the angle between features and cluster centres.

4.1. Comparison of Different Versions of our Approach using a Synthetic Data Set

Synthetic data sets were generated by randomly generating matrices $\mathbf{D}^\star \in \mathbb{R}^{1000}$ and binary $\mathbf{S}^\star \in \mathbb{R}^{10}$ from which the observations were constructed as $\mathbf{X} = \mathbf{D}^\star\mathbf{S}^\star + \mathbf{E}$, where $\mathbf{E}$ is an i.i.d. Gaussian noise term.

The standard deviation of $\mathbf{E}$ was varied from $0.01, 0.1, 1$ and 10. Two distinct regimes were compared, one, in which the average number of features in each cluster were identical and and extreme example in which one cluster had 91 features and all other clusters had a single feature.

The first experiment contrasted several variations of the approach, contrasting three different normalisation steps and two methods to update $\mathbf{S}$. Results, averaged over 1000 random problem instances, are shown in Tables 1 and 2, where, for each noise level, we have highlighted the best performing algorithm version in bold.

Table 1. Performance of variations of our method with equally sized clusters in terms of **NMI**. Columns: different ways to update $\mathbf{S}$; Rows: different normalisations.

	$\mathbf{S} + \mu\mathbf{D}^T(\mathbf{X} - \mathbf{DS})$	$(\mathbf{D}^T\mathbf{D})^{-1}\mathbf{D}^T\mathbf{X}$
std	0.01, 0.1, 1, 2	0.01, 0.1, 1, 2
D	0.95, 0.94, 0.84, 0.56	0.98, 0.97, 0.87, 0.58
S	0.95, 0.93, 0.84, **0.59**	**0.99, 0.98,** 0.87,0.58
NONE	0.95, 0.94, 0.84, 0.57	0.98, 0.97, 0.86, 0.58

Table 2. Performance of variations of our method with widely varying cluster sizes in terms of **NMI**. Columns: different ways to update $\mathbf{S}$; Rows: different normalisations.

	$\mathbf{S} + \mu\mathbf{D}^T(\mathbf{X} - \mathbf{DS})$	$(\mathbf{D}^T\mathbf{D})^{-1}\mathbf{D}^T\mathbf{X}$
std	0.01, 0.1, 1, 2	0.01, 0.1, 1, 2
D	0.44, 0.39, 0.33, 0.27	**0.88, 0.73, 0.41,** 0.26
S	0.49, 0.37, 0.33, 0.26	0.83, 0.67, 0.31, 0.25
NONE	0.62, 0.38, 0.33, **0.27**	0.57, 0.68, 0.34, 0.26

Apart from the condition with very high noise, an update of $\mathbf{S}$ based on the pseudo-inverse of $\mathbf{D}$ performed best. For clusters of similar size a pre-thresholding normalisation of the rows of $\mathbf{S}$ performed better, whilst for clusters of widely varying size, normalisation of columns of $\mathbf{D}$ worked better.

4.2. Comparison of Different Algorithms on Synthetic Data Sets

The next synthetic data sets were generated again by randomly generating matrices $\mathbf{D}^\star$, sparse $\mathbf{S}^\star$ and i.i.d. Gaussian noise $\mathbf{E}$. Four different algorithms (our method, semi-NMF and standard as well as spherical k-means) were contrasted in three different scenarios. In each case, $\mathbf{D} \in \mathbb{R}^{M\times K}$ was generated with i.i.d Gaussian zero-mean unit-variance entries.

1. **Scenario 1:** $\mathbf{S} \in \mathbb{R}^{K\times N}$ was generated with each column set to zero apart from one entry whose location was chosen at random and whose value was set to 1.
2. **Scenario 2:** $\mathbf{S}$ was generated deterministically so that each cluster had different numbers of observations $\mathbf{x}_i$. We here used an extreme example, where there were 3 clusters with only 1 observation, 2 clusters with 3 observations, and 1 cluster each with 6, 10, 14, 24 and 36 observations respectively.
3. **Scenario 3:** This was generated in the same way as dataset 2, with the exception that the cluster centres in $\mathbf{D}$ where each scaled by a zero-mean, unit-variance Gaussian. Thus each cluster did have a different level of noise compared to the size of the cluster centre.

Four different levels of noise were added with variance of $0, 1, 4$ and 9 (See figures (1) to (3) for average SNR values for each condition). Noise was added before normalisation of the observations and results are averaged over 1000 different realisations of each datasets and noise condition.

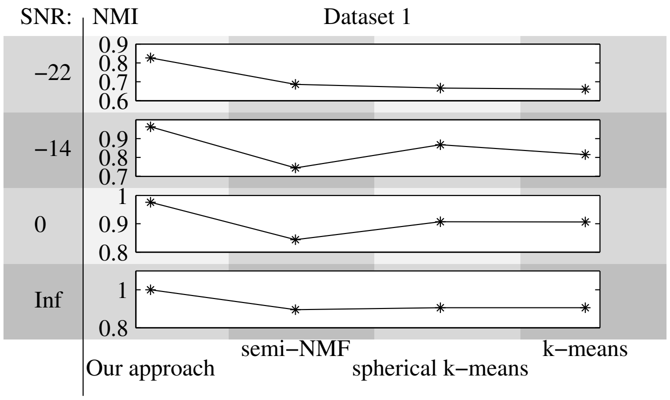

Fig. 1. Performance (in terms of NMI) of our algorithm, semi-NMF clustering and two k-means variants (spherical and standard) for scenario 1 for 4 noise levels.

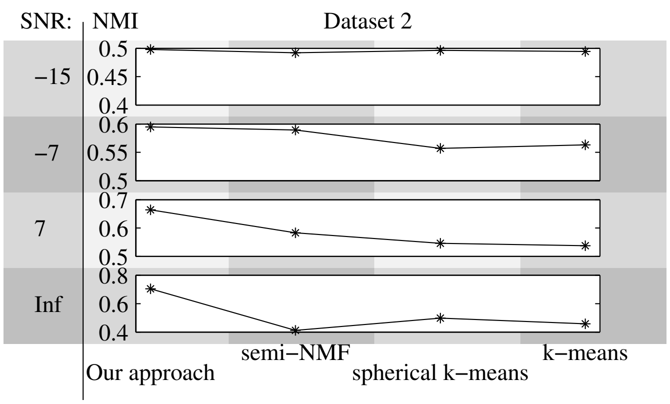

Fig. 2. Performance (in terms of NMI) of our algorithm, semi-NMF clustering and two k-means variants (spherical and standard) for scenario 2 for 4 noise levels.

The results for the three datasets are shown in Figures 1, 2 and 3. The figures are partitioned into four rows, one for each noise level, and four columns, one for each algorithm. The SNR value next to each row are empirical estimate for the level of noise added.

It is clear that for the experiments reported here, our approach outperforms all other reproaches over all datasets and noise conditions. Other key observations are

1. The semi-NMF algorithm sometimes performs better than k-means and sometimes it performs worse.
2. Spherical k-means performs better than non-spherical k-means run on normalised vectors.
3. There is a clear performance decrease when going from dataset 1 to dataset 2, though going from dataset 2 to dataset 3 only reduces performance slightly.
4. The difference between the standard k-means (Euclidean distance) and the spherical k-means is small.

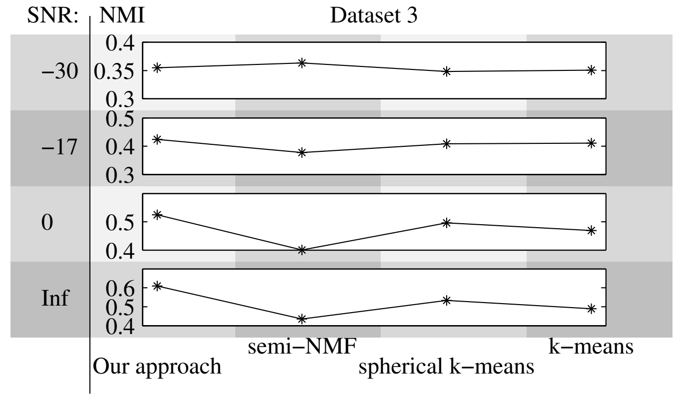

Fig. 3. Performance (in terms of NMI) of our algorithm, semi-NMF clustering and two k-means variants (spherical and standard) for scenario 3 for 4 noise levels.

4.3. Clustering of functional MRI data

We initially developed our new approach for the clustering of brain imaging data-sets. There is extensive interest in the neuroscience community in the development of algorithms to partition the human brain based on functional Magnetic Resonance Imaging (fMRI) data acquired during rest (see [5] for a recent review and additional references). We used fMRI data from 66 subjects, collected during the initial stages of phase 2 of the human connectome project (http://humanconnectome.org/). The data had 2mm isotropic spatial resolution and a temporal resolution of 1.4 seconds. The data was processed using a preliminary version of the Human Connectome Project's structural and functional minimal preprocessing pipelines, final versions to be published separately (Glasser et al. unpublished). Briefly, this involved brain extraction, registration of different MRI modalities, bias field correction, registration to a standard brain template and cortical surface modelling. Functional data were motion corrected, distortion corrected, mean normalised and resampled to the cortical surface. Standard surface smoothing and temporal filtering was applied and ICA based noise reduction used.

For each of the 66 subjects, the dataset consisted of a set of approximately 64000 functional MRI time series, each with approximately 1000 temporal samples each. We split the dataset into two, with 33 subjects each. For each of these splits, we combined the data across subjects by estimating the 1000 left singular vectors of the spatio-temporal data matrix (concatenated in the temporal direction over the 33 subjects). We thus produced two sets of feature vectors, where each vector had a length of 1000 and was associated with one of the vertex locations on the cortical grid representation.

Without ground truth, we use split half repeatability to evaluate performance using Dice similarity, which is the common measure used in the field. The results obtained for different numbers of clusters and different methods is shown in Figure 4. Before calculating dice similarity, we split all clusters we estimated into spatially contiguous regions and then discarded very small clusters (we here removed clusters that had less than 20 features, though the flavour of the results does not vary much if we use another threshold). Also shown are results for the region growing method proposed in [5].

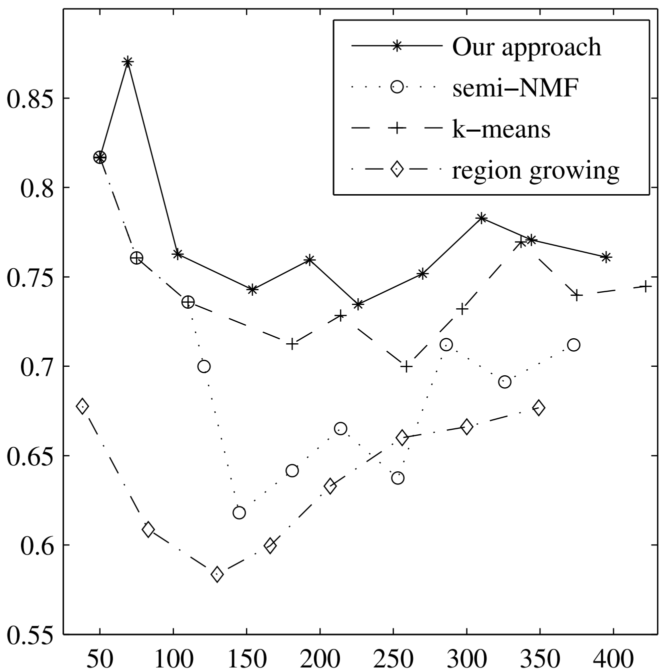

Fig. 4. Comparison of four different approaches for clustering of the cortical surface based on resting-sate fMRI data. Repeatability measured in terms of average Dice similarity between clusters plotted for different numbers of clusters.

Our new method out-performs the there methods. To interpret these results, it must be remembered that the region growing algorithm enforces clusters to be spatially connected. This is known to introduce additional biases into the estimated clusters, which in turn generally means that clusters are more repeatable. Our approach does not include such an additional spatial constraint and is thus not affected by the associated bias.

5. CONCLUSIONS

We have here proposed an alternative clustering algorithm that combines aspects of traditional k-means clustering and non-negative matrix factorisation approaches. Inspired by recent greedy methods for sparse signal decomposition, an iterative algorithm was developed that estimates a matrix factorisation, whilst iteratively enforcing hard cluster assignment. Several empirical studies highlight the performance advantages of the method. We were here particularly interested in applications in brain imaging and have therefore concentrated on directional clustering. For this setting, a characterisation of the fixed points of the algorithm and the global minima of the cost function in terms of singular vectors was derived. An initial analysis of several convergence properties was also presented. There still remain several open theoretical questions on convergence and performance that we are currently studying. Two extensions of the method, the inclusion of spatial constraints and the extension to cluster directly from compressed measurements are studied in a companion paper at this conference [9].

REFERENCES

[1] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[2] A. K. Jain, “Data clustering: 50 years beyond k-means,” *Pattern Recognition Letters*, vol. 31, pp. 651–666, 2010.

[3] C. Ding, L. T., and M. Jordan, “Convex and semi-nonnegative matrix factorizations,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 1, pp. 45–55, 2010.

[4] F. Shahnaz, M. W. Berry, V. P. Pauca, and R. J. Plemmons, “Document clustering using nonnegative matrix factorization,” *Information Processing and Management: an International Journal archive*, vol. 42, no. 2, pp. 373–386, 2006.

[5] T. Blumensath, S. Jbabdi, M. F. Glasser, D. C. V. Essen, K. Ugurbild, T. E. J. Behrens, and S. M. Smith, “Spatially constrained hierarchical parcellation of the brain with resting-state fmri,” *submitted to NeuroImage*, 2013.

[6] T. Blumensath and M. Davies, “Iterative hard thresholding for compressed sensing,” *Applied and Computational Harmonic Analysis*, vol. 27, no. 3, pp. 265–274, 2009.

[7] J. Kiefer, “Sequential minimax search for a maximum,” *Proc. Amer. Math. Soc.*, vol. 4, pp. 502–506, 1953.

[8] A. Strehl and J. Ghosh, “Cluster ensembles - a knowledge reuse framework for combining multiple partitions,” *Journal of Machine Learning Research 3*, pp. 583–617, 2002.

[9] A. Benichoux and T. Blumensath, “Towards functional parcellation of the human brain from compressive fMRI measurements: A spatially constrained, sparse low-rank matrix factorisation approach,” in *Proceedings of the 22nd European Signal Processing Conference (EUSIPCO 2014)*, 2014.