Bi-clustering via MDL-Based Matrix Factorization

Ignacio Ramírez[1] and Mariano Tepper[2]

[1] Universidad de la República
[2] Duke University

Abstract. Bi-clustering, or co-clustering, refers to the task of finding sub-matrices (indexed by a group of columns and a group of rows) within a matrix such that the elements of each sub-matrix are related in some way, for example, that they are similar under some metric. As in traditional clustering, a crucial parameter in bi-clustering methods is the number of groups that one expects to find in the data, something which is not always available or easy to guess. The present paper proposes a novel method for performing bi-clustering based on the concept of low-rank sparse non-negative matrix factorization (S-NMF), with the additional benefit that the optimum rank k is chosen automatically using a minimum description length (MDL) selection procedure, which favors models which can represent the data with fewer bits. This MDL procedure is tested in combination with three different S-NMF algorithms, two of which are novel, on a simulated example in order to assess the validity of the procedure.

1 Introduction

Given a set of data vectors arranged as columns (rows) of a matrix, traditional data clustering corresponds to the task of finding groups of similar columns (rows) within that matrix. Bi-clustering, or co-clustering, refers to the task of finding sub-matrices (indexed by a group of columns and a group of rows) within a matrix such that the elements of each sub-matrix are related in some way. This idea has been widely applied in the last ten years as a powerful tool to analyze data of many kinds, a notorious example being micro-array analysis, where correlations between groups of genes and expression patterns are sought.

Finding the clusters usually involves identifying subsets of row and column indexes so that the indexed sub-matrices exhibit some regularity, for example, that their elements have a constant value, or that their rows, or columns, are identical. As examples of other measures of regularity, the reader can refer to [1]. As a more general problem (that applies to traditional clustering as well), the very number of clusters present in the data may not be known, or assumed, *a priori*.

The present work presents a method to perform bi-clustering under the hypothesis that each sub-matrix is a rank-1 component of the assignment matrix. Assuming that such components are non-negative, something usually required

J. Ruiz-Shulcloper and G. Sanniti di Baja (Eds.): CIARP 2013, Part I, LNCS 8258, pp. 230–237, 2013.

for the resulting models to have physical interpretability, leads to the well known concept of *non-negative matrix factorization* (NMF) [2]. Further assuming that the sub-matrices have a small number of non-zero elements leads to a *sparse* NMF model (S-NMF) [3].

The technique presented here combines S-NMF with a model selection technique, MDL [4], to *automatically infer the proper number of groups from the data*, an idea that has already been proposed for model selection in bi-clustering, albeit using a different formulation, in [5]. To perform S-NMF, we resort to two variants of the well known *sparse dictionary learning* paradigm [6,7], which is based on penalized regression subproblems to impose sparsity on the colums/rows of $\mathbf{U}/\mathbf{V}$, and the sparse SVD technique employed in [8]. We demonstrate the performance of the three resulting methods on a simulated experiment from computer vision [8].

Details on the involved techniques, as well as the formalization of the problem, and the relationship to prior art, are given in Section 2. We then develop our main contribution on Section 3, and show the performance of the developed technique on Section 4, leavning the conclusions to Section 5.

2 Background and Prior Art

Consider a data matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$. We define $\mathcal{I} = \{1, 2, \ldots, m\}$ and $\mathcal{J} = \{1, d, \ldots, n\}$ as the sets of row and column indexes of $\mathbf{X}$ respectively. Given $I \subseteq \mathcal{I}$ and $J \subseteq \mathcal{J}$, we define a bi-cluster as the set $G = I \times J$, the Cartesian product of those two sets, and refer to the associated sub-matrix of $\mathbf{X}$ indexed by G, $\mathbf{X}_{[G]} = \{x_{ij}\}_{i \in I, j \in J}$. We define by $C = \{G_1, G_2, \ldots, G_c\}, G_k = I_k \times J_k$ the set of bi-clusters defined by some method. Note that $\{I_k : k = 1, \ldots, c\}$, $\{J_k : 1, \ldots c\}$ and C need not be partitions of, respectively, $\mathcal{I}$, $\mathcal{J}$ and $\mathcal{I} \times \mathcal{J}$; that is, bi-clusters may overlap.

Several techniques have been developed for performing bi-clustering. The differences among them being primarily the assumptions made about the data. We refer the reader to the recent surveys [1] for details on them. A number of them are based on the NMF idea [2], that is, that the data matrix $\mathbf{X}$ is the product of two other matrices $\mathbf{U} \in \mathbb{R}^{m \times p}$ and $\mathbf{V} \in \mathbb{R}^{p \times n}$ plus some perturbation $\mathbf{E}$, $\mathbf{X} = \mathbf{U}\mathbf{V} + \mathbf{E}$. In the context of NMF-based bi-clustering, one usually associates each k-th cluster to a corresponding pair of column $\mathbf{u}_k$ and row $\mathbf{v}^k$ vectors (observe the indexing notation), so that the NMF decomposition is expressed as

$$\mathbf{X} = \sum_{k=1}^{c} \mathbf{u}_k \mathbf{v}^k + \mathbf{E} \tag{1}$$

The use of the general NMF model (1) stretches well beyond bi-clustering, so that the above formulation does not help in identifying the clusters, but only rank-1 components (each $\mathbf{u}_k \mathbf{v}^k$). In order to obtain a useful bi-clustering, one needs to further assume that only a few elements in $\mathbf{u}_k$ and $\mathbf{v}^k$ are non-zero, so that the support of such vectors define the index subsets, $I_k = \text{supp}(\mathbf{u})_k$ and

$J_k = \text{supp}(\mathbf{v}^k)$. This leads to the idea of S-NMF, which admits several variants. One of them is the so-called *sparse modeling* one, where $\mathbf{V}$ is assumed column-sparse. The Sparse Principal Component Analysis (SPCA) technique developed in [9] belongs to this category. In the context of bi-clustering, the "all-sparse" technique developed in [3] has been applied for example in [10] with significant success. Finally, on the same line of work of this paper, [8] propose a multi-stage sparse SVD for bi-clustering with multi-model estimation as the target application.

In [3,10], sparsity is measured (and controlled) via a *sparseness* statistic which measures, indirectly, the number of non-zero entries in a data vector. The factorization problem is then expressed as an optimization problem constrained on the sparseness of the columns and rows of $\mathbf{U}$ and $\mathbf{V}$. The advantage of this approach is that sparsity is controlled in an indirect way that does not provide any types of guarantees regarding the recovery of correct sparse components in situations where those can be assumed to exist. Also, the degree of sparsity, as well as the number c of factors involved in the decomposition, are parameters which have a great impact on the effectiveness of the solution, and at the same time are challenging to tune.

3 Proposed Method

The main contribution of this work is an MDL-based method for selecting the optimum number of factors in an S-NMF decomposition which is guided by an objective criterion: to obtain a representation which yields the most compressed representation of the data. The method is similar in spirit to the also MDL-based [5], but uses exact codelengths for the data type at hand rather than an approximate expression for large sample sizes.

Our method is applied to three different numerical S-NMF techniques: the sparse SVD [11] used in [8], and two penalized regression-based dictionary learning methods, one using a ℓ_0 penalty, the other using a ℓ_1 regularizer. These are described next.

3.1 Sparse Matrix Factorization

Sparse SVD The standard SVD decomposition of a matrix $\mathbf{Y} \in \mathbb{R}^{m \times n}$ is given by $\mathbf{Y} = \mathbf{USV}^\intercal$, where $\mathbf{U}$ and $\mathbf{V}$ are orthonormal basis of $\mathbb{R}^m$ and $\mathbb{R}^n$ respectively, and $\mathbf{S}$ is a diagonal matrix of non-zero singular values. The sparse SVD [11] is a modification of the aforementioned decomposition where each column and row of $\mathbf{U}$ and $\mathbf{V}$ are forced to be sparse. This is a non-convex problem which is approximated by greedily extracting each i-th rank one component of $\mathbf{Y}$ as $\mathbf{u}_i\mathbf{v}_i^\intercal$ via the following formulation

$$(\mathbf{u}_i, \mathbf{v}_i) = \arg\min_{\alpha,\beta} \frac{1}{2} \left\|\mathbf{R}_i - \alpha\beta^\intercal\right\|_2 + \lambda_u \left\|\alpha\right\|_1 + \lambda_v \left\|\beta\right\|_1, \mathbf{R}_i = \mathbf{Y} - \sum_{l=1}^{i-1} \mathbf{u}_l\mathbf{v}_l^\intercal, \quad (2)$$

that is, the i-th rank one pair is obtained from the residual of $\mathbf{Y}$ after having extracted the previous $i-1$ components.

Penalized Regression. Contrary to [3], and inspired by recent developments in sparse signal recovery (see [12] for a comprehensive review), we obtain sparse factors $\mathbf{U}$ and $\mathbf{V}$ by means of ℓ_ρ regularization,

$$(\hat{\mathbf{U}}, \hat{\mathbf{V}}) = \arg \min_{(\mathbf{U},\mathbf{V})} \frac{1}{2} \|\mathbf{X} - \mathbf{U}\mathbf{V}\|_F^2 + \lambda_u \|\mathbf{U}\|_\rho + \lambda_v \|\mathbf{V}\|_\rho$$
$$\text{s.t.} \quad u_{ik} \geq 0,\ v_{kj} \geq 0 \ \forall\, i, j, k, \tag{3}$$

where $\|\cdot\|_F$ denotes Frobenius norm, and $\|\cdot\|_\rho$ denotes the ℓ_ρ norm of the vectorized matrix argument (that is, the sum of absolute values of the matrix). The problem (3) is non-convex in $(\mathbf{U}, \mathbf{V})$; but it is convex in $\mathbf{U}$ or $\mathbf{V}$ when the other is kept fixed when $\rho \geq 1$. As such, a common strategy to obtain a local minimum of (3) is to perform an alternate minimization between $\mathbf{U}$ and $\mathbf{V}$.

$$\mathbf{V}(t+1) = \arg\min_{\mathbf{V}} \frac{1}{2} \|\mathbf{X} - \mathbf{U}(t)\mathbf{V}\|_F^2 + \lambda_v \|\mathbf{V}\|_\rho \quad \text{s.t.} \quad v_{kj} \geq 0 \tag{4}$$

$$\mathbf{U}(t+1) = \arg\min_{\mathbf{U}} \frac{1}{2} \|\mathbf{X} - \mathbf{U}\mathbf{V}(t+1)\|_F^2 + \lambda_v \|\mathbf{U}\|_\rho \quad \text{s.t.} \quad u_{ik} \geq 0. \tag{5}$$

Note that the form of (5) is a transposed version of (4), so that both steps can be solved with exactly the same method. Note also that, in both cases, the problem is separable in the columns of $\mathbf{V}$ (rows of $\mathbf{U}$), which can greatly simplify the computations. For the case $\rho = 1$, we apply the Fast Iterative Soft Thresholding Algorithm (FISTA) [13] to each column of $\mathbf{V}$ (row of $\mathbf{U}$). For the case $\rho = 0$, we apply the Orthogonal Matching Pursuit (OMP) algorithm [14] to a constrained variant of (3) (and their corresponding alternate minimizations:

$$(\hat{\mathbf{U}}, \hat{\mathbf{V}}) = \arg \min_{(\mathbf{U},\mathbf{V})} \frac{1}{2} \|\mathbf{X} - \mathbf{U}\mathbf{V}\|_F^2$$
$$\text{s.t.} \quad \|\mathbf{U}\|_\rho \leq \lambda_u \qquad \|\mathbf{V}\|_\rho \leq \lambda_v, \quad u_{ik} \geq 0, \quad v_{kj} \geq 0\,, \quad \forall\, i, j, k, \tag{6}$$

3.2 Model Selection

As given above, all formulations (2) through (6) have three critical parameters to be dealt with: λ_u, λ_v and c, the number of factors (columns of $\mathbf{U}$, rows of $\mathbf{V}$). We select such values by posing the following model selection problem. (For simplicity of exposition, here we will consider $\lambda_u = \lambda_v = \lambda$, but in practice we optimize them independently). For a given pair of values (c, λ) we define a (local) solution to (3) as $(\mathbf{U}(c,\lambda), \mathbf{V}(c,\lambda), \mathbf{E}(c,\lambda))$. We then pose an hypothetical compression problem where the task is to describe $\mathbf{X}$ *losslessly* (that is, exactly) in terms of $(\mathbf{V}, \mathbf{U}, \mathbf{E})$. The model selection procedure then computes $(\mathbf{U}(c,\lambda), \mathbf{V}(c,\lambda), \mathbf{E}(c,\lambda))$ for different values of (c, λ) and keeps those that minimize the combined *codelenghts* (in bits) of those three components, $L(\mathbf{X}) = L(\mathbf{V}) + L(\mathbf{U}) + L(\mathbf{E})$..

From a modeling perspective, the bulk of the work at this point is to obtain expressions for the codelengths $L(\cdots)$ of each component. This is clearly a very

data-dependent task. Here we will focus on the simple case of $\mathbf{X}$ being the binary assignment matrix used in the example of Section 4.

A first observation is that, since $\mathbf{X}$ is binary, the only error values that can occur are also binary, so that $\mathbf{E}$ will also be binary. More specifically, by denoting by $[[\cdot]]$ the element-wise binarization function that sets $[[a]] = 0$ if $a < 0.5$ and $[[a]] = 1$ otherwise, we can write $\mathbf{E} = \mathbf{X} \oplus [[\mathbf{UV}]]$. Under the usual assumption that the residual $\mathbf{E}$ is decorrelated, we then have that $\mathbf{E}$ can be described as a (one dimensional) IID Bernoulli sequence of values, which in turn can be efficiently described using an *enumerative code* [15].

As for $\mathbf{U}$ and $\mathbf{V}$, they may not be binary, but they are sparse. Furthermore, since their product will be binarized (thresholded) to produce an approximation to $\mathbf{X}$, we only need to describe them up to a precision q which suffices to preserve their binarized product. Therefore, we represent $\mathbf{U}$ (and $\mathbf{V}$) in two steps: first, the locations of the non-zero entries are encoded using the same enumerative code used for $\mathbf{E}$, and then the values of the non-zero entries are encoded using a uniform distribution between the integers 0 and Q_u (Q_v), where Q_u (Q_v) is the largest integer so that $qQ_u \geq \max \mathbf{U}$ (same for Q_v). Denoting by $\|\cdot\|_0$ the pseudo-norm that counts the number of non-zero elements in the argument, we then have a total cost function for the model selection problem:

$$\begin{aligned} L(\mathbf{X}) = &\log_2 \binom{\|\mathbf{E}\|_0}{mn} + \log_2 mc + \log_2 cn \\ &\log_2 \binom{\|\mathbf{U}\|_0}{mc} + \log_2(\|\mathbf{U}\|_0 \, Q_u) + \log_2 \binom{\|\mathbf{V}\|_0}{cn} + \log_2(\|\mathbf{V}\|_0 \, Q_v) \end{aligned} \quad (7)$$

The total problem now involves the minimization of $L(\mathbf{X})$ in terms of three parameters: c, λ and q. The search for the best parameters is done in a nested fashion. Following a standard model selection approach, the order of the model (the number of clusters) c is the outermost loop, starting from $c = 0$, and increasing by one. For each fixed c, a solution is sought by optimizing for various values of λ and q. This part, in turn, is done by first obtaining a set of unquantized factors $\mathbf{V}$ and $\mathbf{U}$ for each candidate λ, and then evaluating (7) for different values of q. The innermost loop in q, and the evaluation of (7) are very fast operations. The real computational cost of the method comes from obtaining the unquantized pair $(\mathbf{U}, \mathbf{V})$ for each λ, which involves the alternating minimization algorithm (4)-(5), which are costly steps.

4 Experimental Results

Here we report on one of the simulated computer vision problems described in [8]. The task here is to find sets of aligned points which are mixed in a background of randomly scattered points. The ground truth consists of 11 line segments which are sampled (plus a small perturbation) 50 times, together with randomly sampled points, all within the 2D square $[-1, 1] \times [-1, 1]$. The total number of data points is 1100; those are shown in Fig. 1(b).

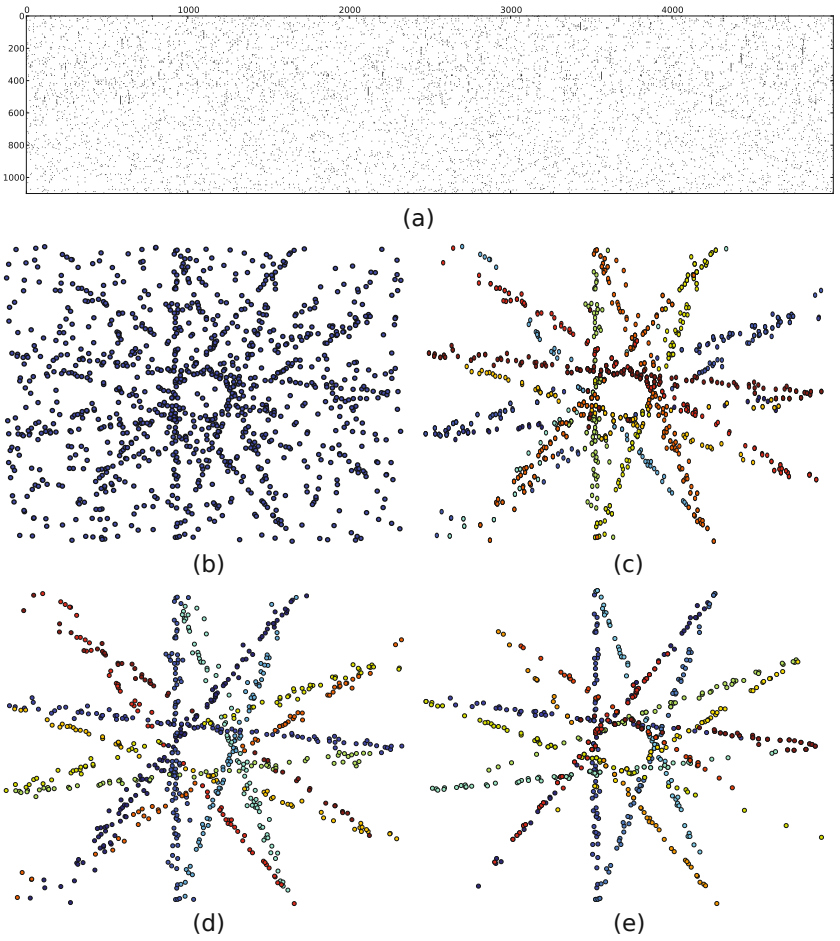

Fig. 1. Results for the line segment search problem. (a) Assignment data matrix, (b) data points, (c,d,e) final bi-clustering obtained using, respectively, ℓ_1, ℓ_0 and S-SVD: each color corresponds to one of the $c = 11$ blocks found by the algorithm. Each blocks groups together similar models (lines) which explain approximately the same points. Note that $c = 11$ is the correct number of total blocks as per the simulation. Here ℓ_0 (d) and S-SVD (e) give the best result, the latter yielding the correct number of clusters but including a few outliers, the former over-estimating the number of groups by $c = 13$ but doing a better job at rejecting outliers.

Each column j of the assignment matrix $\mathbf{X}$ (Figure 1(a)) is computed by selecting two points at random from the dataset, and setting x_{ij} to 1 or 0 if the i-th datapoint is respectively close (distance smaller than a given threshold τ) to the segment formed by those two points or not. This is performed $n = 5000$ times. A statistical tests is performed to remove all segments (columns of $\mathbf{X}$) for which the number of points assigned to them is too low (see [8] for details),

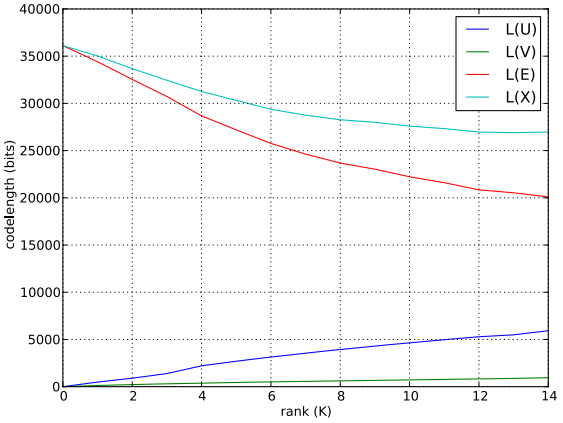

Fig. 2. Codelength vs rank for the sparse SVD case. Notice how the optimum achieves a balance between the bits saved by removing redundancy from $\mathbf{Y}$ (this is, $L(E)$) and the ones added by including $\mathbf{U}$ and $\mathbf{V}$ ($L(U)$ and $L(V)$) in the description.

and then the proposed algorithms are executed. Figures 1(c,d,e) are then drawn by plotting the points corresponding to the non-zero entries of each column of the matrix $\mathbf{U}$ obtained using, respectively, the ℓ_1 regression, ℓ_0 regression, and the sparse SVD methods, using the parameters which yield the shortest codelength in each case. Figure 2 shows the rank vs. codelength graphs obtained for the SVD case (the others are similar). For this example, both penalized regression methods yielded the correct number of groups in the original data (11), but had a tendency to add outliers, as can be seen in Figs. 2(c,d). In the case of S-SVD, the number of groups was over-estimated to 13 instead of 11, but the point assignments was better. As per the execution time, the different methods vary dramatically. The ℓ_0-based method took less than 1 second to perform the model selection for all possible ranges (all methods stop when increasing the range does not decrease the codelength), whereas the S-SVD method required about 30 seconds, and the ℓ_1 one required over an hour. The difference in speed here is due to ℓ_1-based optimization being much more expensive in general than ℓ_0 approximations. Clearly, which one is better will depend on the requirements of each case.

5 Conclusions

We have presented a novel method for bi-clustering that combines a non-negative matrix factorization with the powerful MDL model selection criterion for choosing the best model parameters. We have demonstrated the effectiveness of our approach for three different factorization methods. More results and a more in-depth analysis will be presented in a full article to be published elsewhere.

Acknowledgments. We acknowledge the support of Prof. Guillermo Sapiro from Duke University, Duke University, DHS/JHU and AFOSR.

References

1. Madeira, S., Oliveira, A.: Biclustering Algorithms for Biological Data Analysis: A Survey. IEEE Trans. CBB 1(1), 24–45 (2004)
2. Lee, D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization. Nature 401(6755), 7880–791 (1999)
3. Hoyer, P.: Non-negative matrix factorization with sparseness constraints. JMLR 5, 1457–1469 (2004)
4. Barron, A., Rissanen, J., Yu, B.: The minimum description length principle in coding and modeling. IEEE Trans. IT 44(6), 2743–2760 (1998)
5. Jornsten, R., Yu, B.: Simultaneous gene clustering and subset selection for sample classification via MDL. Bioinformatics 19(9), 1100–1109 (2003)
6. Olshausen, B.A., Field, D.J.: Sparse coding with an overcomplete basis set: A strategy employed by v1? Vision Research 37, 3311–3325 (1997)
7. Aharon, M., Elad, M., Bruckstein, A.: The K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representations. IEEE Trans. SP 54(11), 4311–4322 (2006)
8. A bi-clustering formulation of multiple model estimation (submitted, 2013)
9. Zou, H., Hastie, T., Tibshirani, R.: Sparse Principal Component Analysis. Computational and Graphical Statistics 15(2), 265–286 (2006)
10. Hochreiter, S., Bodenhofer, U., Heusel, M., Mayr, A., Mitterecker, A., Kasim, A., Adetayo, K., Tatsiana, S., Suzy, V., Lin, D., Talloen, W., Bijnens, L., Shkedy, Z.: FABIA: factor analysis for biclustering acquisition. Bioinformatics 26(12), 1520–1527 (2010)
11. Lee, M., Shen, H., Huang, J.Z., Marron, J.S.: Biclustering via sparse singular value decomposition. Biometrics 66(4), 1087–1095 (2010)
12. Bruckstein, A.M., Donoho, D.L., Elad, M.: From Sparse Solutions of Systems of Equations to Sparse Modeling of Signals and Images. SIAM Review 51(1), 34–81 (2009)
13. Beck, A., Teboulle, M.: A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. SIAM Journal on Imaging Sciences 2(1), 183–202 (2009)
14. Pati, Y.C., Rezaiifar, R., Krishnaprasad, P.S.: Orthogonal Matching Pursuit: Recursive function approximation with applications to wavelet decomposition. In: Proc. 27th Ann. Asilomar Conf. Signals, Systems, and Computers (1993)
15. Cover, T.M.: Enumerative source coding. IEEE Trans. Inform. Theory 19, 73–77 (1973)