Clustering and Metaclustering with Nonnegative Matrix Decompositions

Liviu Badea

AI Lab, National Institute for Research and Development in Informatics,
8-10 Averescu Blvd., Bucharest, Romania
badea@ici.ro

Abstract. Although very widely used in unsupervised data mining, most clustering methods are affected by the instability of the resulting clusters w.r.t. the initialization of the algorithm (as e.g. in k-means). Here we show that this problem can be elegantly and efficiently tackled by *meta-clustering* the clusters produced in several different runs of the algorithm, especially if "*soft*" clustering algorithms (such as Nonnegative Matrix Factorization) are used both at the object- and the meta-level. The essential difference w.r.t. other meta-clustering approaches consists in the fact that our algorithm detects frequently occurring *sub*-clusters (rather than *complete* clusters) in the various runs, which allows it to outperform existing algorithms. Additionally, we show how to perform *two-way meta*-clustering, i.e. take both object and sample dimensions of clusters simultaneously into account, a feature which is essential e.g. for *bi*clustering gene expression data, but has not been considered before.

1 Introduction and Motivation

Clustering is one of the most widely used unsupervised learning methods and the number of different clustering approaches is overwhelming. However, despite their wide variety, most clustering methods are affected by a common problem: the *instability* of the resulting clusters w.r.t. the initialization of the algorithm (as in the case of k-means) and w.r.t. slight differences in the input dataset as a result of resampling the initial data (e.g. in the case of hierarchical clustering). This is not surprising if we adopt a unifying view of clustering as a constrained optimization problem, since the fitness landscape of such a complex problem may involve many different local minima into which the algorithm may get caught when started off from different initial states.

Although such an instability seems hard to avoid, we may be interested in the clusters that keep reappearing in the majority of the runs of the algorithm. This is related to the problem of *combining multiple clustering systems*, which is the unsupervised analog of the classifier combination problem [8], a comparatively simpler problem that has attracted a lot of research in the past decade. Combining clustering results is more complicated than combining classifiers, as it involves solving an additional so-called *cluster correspondence* problem, which amounts to finding the best matches between clusters generated in different runs.

J. Gama et al. (Eds.): ECML 2005, LNAI 3720, pp. 10 – 22, 2005.

The cluster correspondence problem can also be cast as an unsupervised optimization problem, which can be solved by a (meta-) clustering algorithm. Choosing an appropriate meta-clustering algorithm for dealing with this problem crucially depends on the precise notion of cluster correspondence.

A very strict notion of *one-to-one correspondence* between the clusters of each pair of clustering runs may be too tough to be realized in most practical cases. For example, due to the above-mentioned *instability*, different runs of k-means clustering with different initializations may easily produce different sets of clusters, e.g. $run_1 = \{\{1,2,3\},\{4,5\}\}$, $run_2 = \{\{1,2\}, \{3,4,5\}\}$, ….

A more lenient notion of cluster correspondence would look for clusters that keep reappearing in all runs (while ignoring the rest), but only very few (if any) such clusters may exist for a large enough number of runs.

An even less restrictive notion could be envisioned by looking for clusters that are most similar (although not necessarily identical) across all runs. This is closest to performing something like single-linkage hierarchical clustering on the sets of clusters produced in the various clustering runs, with the additional constraint of allowing in each meta-cluster no more than a single cluster from each individual run. Unfortunately, this constraint will render the meta-clustering algorithm highly unstable. Thus, while trying to address the instability of (object-level) clustering using meta-level clustering, we end up with instability in the meta-clustering algorithm itself. Therefore, a "softer" notion of cluster correspondence is needed.

The main motivation for this work comes from genomics, more precisely from clustering *gene expression data* [2]. (Therefore, in the following we will frequently refer to clusters of genes rather than clusters of more abstract objects.) Most currently used clustering algorithms produce *non-overlapping* clusters, which represents a serious limitation in this domain, since a gene is typically involved in several biological processes. Here we adopt a biologically plausible simplifying assumption that the overlap of influences (biological processes) is *additive*

$$X_{sg} = \sum_c X(s, g \mid c) \tag{1}$$

where X_{sg} is the expression level of gene g in data sample s, while $X(s,g \mid c)$ is the expression level of g in s due to biological process c. We also assume that $X(s,g \mid c)$ is multiplicatively decomposable into the expression level A_{sc} of the biological process (cluster) c in sample s and the membership degree S_{cg} of gene g in c:

$$X(s, g \mid c) = A_{sc} \cdot S_{cg} \tag{2}$$

2 Nonnegative Matrix Factorization as a Soft Clustering Method

Combining (1) and (2) leads to the reformulation of our clustering problem as a *Nonnegative Matrix Factorization* (of the $n_s \times n_g$ matrix X as a product of an $n_s \times n_c$ matrix A and an $n_c \times n_g$ matrix S):

$$X_{sg} \approx \sum_c A_{sc} \cdot S_{cg} \tag{3}$$

with the additional nonnegativity constraints: $A_{sc} \geq 0,\ S_{cg} \geq 0$ (4)
(Expression levels and membership degrees cannot be negative.)

Such a problem can be cast as a constrained optimization problem:

$$\text{minimize } C(A,S) = \frac{1}{2} \| X - A \cdot S \|_F^2 = \frac{1}{2} \sum_{s,g} \left(X - A \cdot S\right)_{sg}^2 \quad (5)$$

subject to the nonnegativity constraints (4), and could be solved using Lee and Seung's *Nonnegative Matrix Factorization (NMF)* algorithm [4,5].

As explained above, such a factorization can be viewed as a "soft" clustering algorithm allowing for overlapping clusters, since we may have several significant S_{cg} entries on a given column g of S (so a gene g may "belong" to several clusters c).

Allowing for cluster overlap alleviates but does not completely eliminate the instability of clustering, since the optimization problem (5), (4) is non-convex.

In particular, the NMF algorithm produces different factorizations (biclusters) $(A^{(i)},S^{(i)})$ for different initializations and meta-clustering the resulting "soft" clusters $S^{(i)}$ could be used to obtain a more stable set of clusters.

However, using a "hard" *meta*-clustering algorithm would once again entail an unwanted instability. Therefore, we propose using Nonnegative Matrix Factorization as a "soft" meta-clustering approach.

This not only alleviates the instability of a "hard" meta-clustering algorithm, but also produces a "base" set of "*cluster prototypes*", out of which all clusters of all individual runs can be recomposed, despite the fact that they may not correspond to identically reoccurring clusters in all individual runs (see Figure 1).

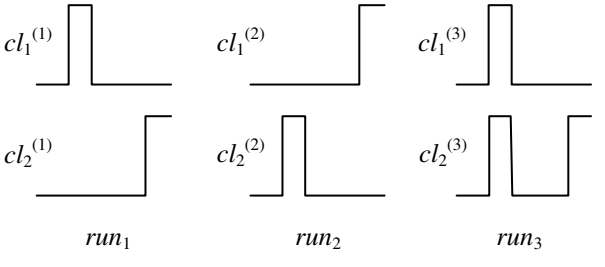

Fig. 1. Clusters obtained in different runs are typically combinations of a "base" set of "cluster prototypes" (rather than identical across all runs)

3 Metaclustering with NMF

We propose using NMF both for object-level clustering *and* meta-clustering. This unified approach solves in an elegant manner both the clustering and the cluster correspondence problem. More precisely, we first run NMF as object-level clustering r times:

$$X \approx A^{(i)} \cdot S^{(i)} \qquad i = 1,\ldots,\ r \quad (6)$$

where X is the data matrix to be factorized (samples × objects to be factorized), $A^{(i)}$ (samples × clusters) and $S^{(i)}$ (clusters × objects).

To allow the comparison of membership degrees S_{cg} for different clusters c, we scale the rows of $S^{(i)}$ to unit norm by taking advantage of the scaling invariance of the above factorization (6). More precisely:

Proposition. The NMF objective function (5) is invariant under the transformation $A \leftarrow A \cdot D$, $S \leftarrow D^{-1} \cdot S$, where $D = diag(d_1,\ldots,d_{nc})$ is a positive diagonal matrix.

Since a diagonal matrix D operates on the rows of S and on the columns of A, we can scale the rows of S to unit norm by using a diagonal scaling with $d_c = \sqrt{\sum_g S_{cg}^2}$.

Next, we build a global S-matrix of size $r \cdot n_c \times n_g$:

$$S^G = \begin{pmatrix} S^{(1)} \\ \vdots \\ S^{(r)} \end{pmatrix} \tag{7}$$

by collecting all clusters (i.e. rows of $S^{(i)}$) from all runs and then use NMF once more to meta-cluster these clusters (i.e. the *rows* of S^G):

$$S^G \approx \alpha \cdot \gamma \tag{8}$$

where α and γ are of sizes $r \cdot n_c \times n_c$ and $n_c \times n_g$ respectively. Note that whereas object level NMF clusters *columns* of X (e.g. genes in our genomics application), meta-clustering clusters *rows* of S^G.

Note that α encodes the *cluster – metacluster correspondence*. On the other hand, the rows of γ make up a base set of *cluster prototypes*, out of which all clusters of all individual runs can be recomposed:

$$S_c^{(i)} = \sum_m \alpha_{c+(i-1)n_c,m} \cdot \gamma_m \tag{9}$$

where $S_c^{(i)}$ is the row c of $S^{(i)}$, while γ_m is the row m of γ.

Using the notation $\alpha_{cm}^{(i)}$ for $\alpha_{c+(i-1)n_c,m}$, we can rewrite (9) as

$$S_c^{(i)} = \sum_m \alpha_{cm}^{(i)} \cdot \gamma_m \tag{9'}$$

Ideally (in case of a perfect one-to-one correspondence of clusters across runs), we would expect the rows of α to contain a single significant entry $\alpha_{c,m(i,c)}^{(i)}$, so that each cluster $S_c^{(i)}$ corresponds to a single cluster prototype $\gamma_{m(i,c)}$ (where $m(i,c)$ is a function of i and c): $S_c^{(i)} = \alpha_{c,m(i,c)}^{(i)} \cdot \gamma_{m(i,c)}$ (10)

Additionally, each meta-cluster m should contain no more than a single cluster from each individual run, i.e. there should be no significant entries $\alpha_{c'm}^{(i)}$ and $\alpha_{c''m}^{(i)}$ with $c' \neq c''$. Although it could be easily solved by a hard meta-clustering algorithm, such an ideal cluster correspondence is only very seldom encountered in practice, mainly due to the *instability* of most clustering algorithms.

Thus, instead of such a perfect correspondence (10), we settle for a weaker one (9) in which the rows of α can contain several significant entries, so that all clusters (rows of S^G) are recovered as *combinations* of cluster prototypes (rows of γ).

The nonnegativity constraints of NMF meta-clustering are essential both for allowing the interpretation of γ as cluster prototypes as well as for obtaining sparse factorizations (α,γ). (Experimentally, the rows of α tend to contain typically one or only very few significant entries.)

In order to make the prototype clusters (rows of γ) directly comparable to the clusters (rows) from S^G, we use the diagonal scaling

$$\alpha \leftarrow \alpha \cdot D^{-1}, \quad \gamma \leftarrow D \cdot \gamma \quad \text{with} \quad D = diag\left(\frac{1}{r}\sum\nolimits_j \alpha_{jm}\right).$$

The cluster prototypes matrix γ produced by meta-clustering (8) is subsequently used as seed for a final NMF run aiming at producing the final factorization. More precisely, the seed for the final NMF run is (A_0,γ), where A_0 is the *nonnegative least squares* solution to $X \approx A_0 \cdot \gamma$.

We thus obtain a final factorization (3), which can be interpreted as a stable clustering of X allowing for overlapping clusters. The algorithm is summarized below.

Clustering with Metaclustering (X) → (A, S)

for $i = 1,\ldots,r$

run NMF($X,A^{(0i)},S^{(0i)}$) with random initial matrices $A^{(0i)},S^{(0i)}$ to produce a factorization with n_c clusters: $X \approx A^{(i)} \cdot S^{(i)}$

scale the rows of $S^{(i)}$ to unit norm:

$A^{(i)} \leftarrow A^{(i)} \cdot D,\ S^{(i)} \leftarrow D^{-1} \cdot S^{(i)}$ with $D = diag\left(\sqrt{\sum_g S_{cg}^2}\right)$

end

Construct $S^G = \begin{pmatrix} S^{(1)} \\ \vdots \\ S^{(r)} \end{pmatrix}$ and use NMF($S^G,\alpha^{(0)},\gamma^{(0)}$) (with random $\alpha^{(0)},\gamma^{(0)}$) to produce a factorization ("meta-clustering") with internal dimensionality n_c: $S^G \approx \alpha \cdot \gamma$

scale the columns of α: $\alpha \leftarrow \alpha \cdot D^{-1}, \quad \gamma \leftarrow D \cdot \gamma$ with $D = diag\left(\sum_j \alpha_{jm} / r\right)$

Let A_0 be the *nonnegative least squares* solution to $X \approx A_0 \cdot \gamma$

Run NMF(X,A_0,γ) to produce the final factorization $X \approx A \cdot S$

NMF(X, A_0, S_0) → (A,S)

$A \leftarrow A_0,\ S \leftarrow S_0$

loop $S_{cg} \leftarrow S_{cg} \dfrac{(A^T \cdot X)_{cg}}{(A^T \cdot A \cdot S)_{cg}}$

$$A_{sc} \leftarrow A_{sc} \frac{(X \cdot S^T)_{sc}}{(A \cdot S \cdot S^T)_{sc}}$$

until convergence.

4 Sparser Decompositions

Although NMF tends to produce sparse factorizations that are quite immune to moderate levels of noise [4], even sparser decompositions may be desired to cope with higher noise levels. An ad-hoc approach to obtaining such sparser factorizations would fix to zero all the elements below a given threshold of an NMF factorization and then apply several re-optimization rounds until a fixpoint is attained. Hoyer's *Nonnegative Sparse Coding (NNSC)* algorithm [3] is a more elegant approach that factorizes $X \approx A \cdot S$ by optimizing an objective function that combines the *fit* of the factorization to the original data with a *size term* penalizing the non-zero entries of S:

$$\text{minimize} \qquad C(A,S) = \frac{1}{2}\|X - AS\|_F^2 + \lambda \sum_{c,g} S_{cg} \tag{11}$$

subject to the nonnegativity constraints $A_{sc} \geq 0,\ S_{cg} \geq 0$.
(NMF is recovered by setting the size parameter λ to zero, while a non-zero λ would lead to sparser factorizations.) Unfortunately however, the scaling invariance of the fitness term $\frac{1}{2}\|X - AS\|_F^2$ makes the size term ineffective, since the latter can be forced as small as needed by using a diagonal scaling D with small enough entries. Additional constraints are therefore needed to render the size term operational. Since a diagonal matrix D operates on the rows of S and on the columns of A, we could impose unit norms either for the rows of S, or for the columns of A.

Unfortunately, the objective function (11) used in [3] produces decompositions that depend on the scale of the original matrix X (i.e. the decompositions of X and ηX are essentially different), regardless of the normalization scheme employed. For example, if we constrain the rows of S to unit norm, then we cannot have decompositions of the form $X \approx A \cdot S$ and $\eta X \approx \eta A \cdot S$, since at least one of these is in general non-optimal due to the dimensional inhomogeneity of the objective function w.r.t. A and X: $C_{\eta X}(\eta A, S) = \eta^2 \frac{1}{2}\|X - AS\|_F^2 + \lambda \sum_{c,g} S_{cg}$. On the other hand, if we constrain the columns of A to unit norm, the decompositions $X \approx A \cdot S$ and $\eta X \approx A \cdot \eta S$ cannot be both optimal, again due to the dimensional inhomogeneity of C, now w.r.t. S and X: $C_{\eta X}(A, \eta S) = \eta^2 \frac{1}{2}\|X - AS\|_F^2 + \eta\lambda \sum_{c,g} S_{cg}$.

Therefore, as long as the size term depends only on S, we are forced to constrain the columns of A to unit norm, while employing an objective function that is *dimensionally homogeneous* in S and X. One such dimensionally homogeneous objective function is:

$$C(A,S) = \frac{1}{2}\|X - AS\|_F^2 + \lambda\|S\|_F^2 \tag{12}$$

which will be minimized subject to the nonnegativity constraints and the constraints on the norm of the columns of A: $\|A_c\| = 1$ (i.e. $\sum_s A_{sc}^2 = 1$).

It can be easily verified that this produces *scale independent decompositions*, i.e. if $X \approx A \cdot S$ is an optimal decomposition of X, then $\eta X \approx A \cdot \eta S$ is an optimal decomposition of ηX.

The constrained optimization problem could be solved with a gradient-based method. However, in the case of NMF, faster so-called "multiplicative update rules" exist [5,3], which we have modified for the NNSC problem as follows.

Modified NNSC algorithm

Start with random initial matrices A and S

loop $S_{cg} \leftarrow S_{cg} \dfrac{(A^T \cdot X)_{cg}}{(A^T \cdot A \cdot S + \lambda S)_{cg}}$

$A \leftarrow A + \mu(X - A \cdot S) \cdot S^T$

normalize the columns of A to unit norm: $A \leftarrow A \cdot D^{-1}$, $D = diag\left(\sqrt{\sum_s A_{sc}^2}\right)$

until convergence.

Note that we factorize X rather than X^{T} since the sparsity constraint should affect the clusters of genes (i.e. S) rather than the clusters of samples A. (This is unlike NMF, for which the factorizations of X and X^{T} are symmetrical.)

4.1 Sparser Factorizations and Noise

To demonstrate that sparser factorizations are better at coping with noise than simple NMF, we generated a synthetic dataset with highly overlapping clusters and very large additive noise (the standard deviation of the noise was 0.75 of the standard deviation of the original data).

We ran our modified NNSC algorithm with increasingly larger λ (ranging from 0 to 0.75) and observed that the gene clusters were recovered almost perfectly despite the very large noise, especially for small values of λ.

Figure 2 shows the original data without and with noise respectively (upper row), as well as the reconstructed data ($A \cdot S$) for λ=0.05 and λ=0.75 (lower row). Note that the reconstructed data is closer to the original noise-free data than to the noisy original. The following Table shows the relative error computed w.r.t. the noisy data X_{noisy} (i.e. $\varepsilon_{noisy} = \|X_{noisy} - AS\|_F / \|X_{noisy}\|_F$) as well as the relative error w.r.t. the original data X_{orig} (before adding noise, i.e. $\varepsilon_{orig} = \|X_{orig} - AS\|_F / \|X_{orig}\|_F$) for several values of λ ranging from 0 to 0.75.

λ	0	0.02	0.05	0.1	0.2	0.4	0.5	0.75
ε_{noisy}	0.2001	0.2017	0.2053	0.2161	0.2295	0.2452	0.2983	0.3228
ε_{orig}	0.1403	0.1375	**0.1364**	0.1419	0.1544	0.1719	0.2323	0.2604

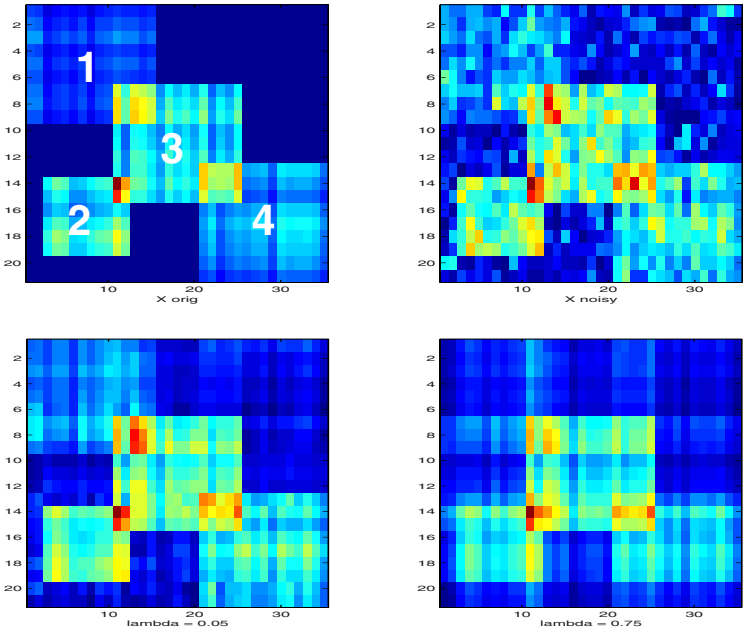

Fig. 2. A synthetic dataset

Note that ε_{orig} is always lower than ε_{noisy}. Also, whereas ε_{noisy} increases as expected with λ, ε_{orig} attains a minimum at λ=0.05 showing that small values of λ tend to improve not only the clusters, but also the error w.r.t. the *original* data.

5 Two-Way Meta-clustering

The meta-clustering approach based on (8) takes only the gene clusters (rows of $S^{(i)}$) into account. Although this works very well in many cases, it will fail whenever two clusters correspond to very similar sets of genes, while differing along the sample dimension. For example, clusters 1 and 2 from Figure 2 are quite similar along the gene dimension, so a meta-clustering method looking just at genes would be incapable of discriminating between the two clusters, unless it also looks at the "sample clusters" $A^{(i)}$. In the following, we show that a slight generalization of NMF, namely *Positive Tensor Factorization* (*PTF*) [6] can be successfully used to perform *two-way* meta-clustering, which takes both the gene and the sample dimensions into account. (As far as we know, this elegant view of metaclustering as a PTF problem has not been considered before.)

Naively, one would be tempted to try clustering the biclusters[1] $A_c^{(i)} \cdot S_c^{(i)}$ instead of the gene clusters $S_c^{(i)}$, but this is practically infeasible in most real-life datasets because it involves factorizing a matrix of size $r \cdot n_c \times n_s \cdot n_g$. On closer inspection,

[1] $A_c^{(i)}$ is the column c of $A^{(i)}$, while $S_c^{(i)}$ is the row c of $S^{(i)}$.

however, it turns out that it is not necessary to construct this full-blown matrix – actually we are searching for a *Positive Tensor Factorization* of this matrix [2]

$$A^{(i)}_{sc} \cdot S^{(i)}_{cg} \approx \sum_{k=1}^{n_c} \alpha^{(i)}_{ck} \cdot \beta_{sk} \cdot \gamma_{kg} \tag{14}$$

The indices in (14) have the following domains: s – samples, g – genes, c – clusters, k – metaclusters. To simplify the notation, we could merge the indices i and c into a single index (ic), so that the factorization becomes:

$$A_{s(ic)} \cdot S_{(ic)g} \approx \sum_{k=1}^{n_c} \alpha_{(ic)k} \cdot \beta_{sk} \cdot \gamma_{kg} \tag{14'}$$

Note that β and γ are the "unified" versions of $A^{(i)}$ and $S^{(i)}$ respectively, while α encodes the *cluster-metacluster correspondence*.

The factorization (14') can be computed using the following multiplicative update rules (the proofs are straightforward generalizations of those for NMF and can also be found e.g. in [6]):

$$\begin{aligned}
\alpha &\leftarrow \alpha * \frac{(A^T \cdot \beta) * (S \cdot \gamma^T)}{\alpha \cdot [(\beta^T \cdot \beta) * (\gamma \cdot \gamma^T)]} \\
\beta &\leftarrow \beta * \frac{A \cdot [\alpha * (S \cdot \gamma^T)]}{\beta \cdot [(\alpha^T \cdot \alpha) * (\gamma \cdot \gamma^T)]} \\
\gamma &\leftarrow \gamma * \frac{[\alpha * (A^T \cdot \beta)]^T \cdot S}{[(\alpha^T \cdot \alpha) * (\beta^T \cdot \beta)]^T \cdot \gamma}
\end{aligned} \tag{15}$$

where '$*$' and '$--$' denote element-wise multiplication and division of matrices, while '$\cdot$' is ordinary matrix multiplication.

The PTF factorization (14') should be contrasted with our previous metaclustering approach (8) based on NMF:

$$S^{(i)}_{cg} \approx \sum_{k=1}^{n_c} \alpha^{(i)}_{ck} \cdot \gamma_{kg} \tag{8'}$$

It can be easily seen that whereas (14) groups biclusters by taking both the gene and the sample dimension into account, (8') may confuse two biclusters that have similar gene components (even if they have different sample supports). For example, (8') confuses biclusters 1 and 2 from Figure 2, while (14) is able to perfectly discriminate between the two despite the noise and the difference in intensity between the two biclusters.

After convergence of the PTF update rule, we normalize the rows of γ to unit norm ($||\gamma_k|| = 1$), as well as the columns of α such that $\sum_{i,c} \alpha^{(i)}_{ck} = r$ (r being the number of runs): [3]

---

[2] More precisely, we are dealing with the constrained optimization problem $\min C(\alpha,\beta,\gamma) = \frac{1}{2} \sum_{i,c,s,g} \left(A^{(i)}_{sc} S^{(i)}_{cg} - \sum_{k=1}^{n_c} \alpha^{(i)}_{ck} \beta_{sk} \gamma_{kg} \right)^2$ subject to $\alpha,\beta,\gamma \geq 0$.

[3] In order to be able to interpret β and γ as "unified" $A^{(i)}$ and $S^{(i)}$ respectively, we need to have $\sum_c \alpha^{(i)}_{ck} \approx 1$, i.e. $\sum_{i,c} \alpha^{(i)}_{ck} \approx r$, since $X \approx \sum_c A^{(i)}_{sc} \cdot S^{(i)}_{cg} \approx \sum_{k=1}^{n_c} \left(\sum_c \alpha^{(i)}_{ck} \right) \cdot \beta_{sk} \cdot \gamma_{kg}$.

$$\gamma_{kg} \mapsto \frac{1}{\|\gamma_k\|} \cdot \gamma_{kg}$$

$$\alpha_{(ic)k} \mapsto \frac{r}{\sum_{i',c'} \alpha_{(i'c')k}} \cdot \alpha_{(ic)k}$$

$$\beta_{sk} \mapsto \frac{\|\gamma_k\|}{r} \cdot \sum_{i,c} \alpha_{(ic)k} \cdot \beta_{sk}$$

and then run NMF initialized with (β, γ) to produce the final factorization $X \approx A \cdot S$.

6 Experimental Evaluation

We evaluated our algorithm on synthetic datasets[4] generated by continuous latent variable graphical models as in Figure 3. (The clusters corresponding to the latent variables L_k *overlap* in the variables X_i influenced by *several* L_k.) To generate *nonnegative bi*clusters as in Figure 2, we set L_k to nonzero values (drawn from the absolute of a normal distribution) only in certain subsets of samples. The observable variables X_i were affected by additive normal noise with a standard deviation $\sigma(\varepsilon_i) = \nu \cdot \sigma(X_i^{(0)})$ equal to a fraction ν of the standard deviation of the noise-free "signal" $X_i^{(0)}$. (It can be easily seen that this model is equivalent to $X = A^{(0)} \cdot S^{(0)} + \varepsilon$, where $X = (X_1 \ \ldots \ X_{ng})$ and $S^{(0)}{}_{kj}$ are the coefficients associated to the $L_k \rightarrow X_j$ edges.)

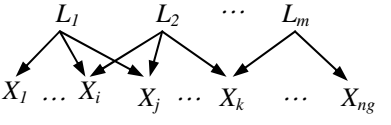

Fig. 3. Latent variable model for overlapping clusters

The Table below presents a comparison of various combinations of clustering and meta-clustering algorithms – columns of the Table correspond to clustering algorithms (k-means, fuzzy k-means [7] and NMF), while the rows are associated to meta-clustering algorithms (k-means, fuzzy k-means, NMF and PTF) with or without a final NMF step, as well as to the best individual clustering run (resampling). The figures in the Table represent average *matches* of the reconstructed clusters with the original ones, together with the associated *relative errors* (we display both averages and standard deviations for 10 runs of each metaclustering method with different input data X).

[4] The small dataset from Figure 2 with 21 samples and 35 genes allows a human analysis of the results.

Defining the *match* between two sets of possibly *overlapping* clusters is nontrivial. For each cluster C_1 from clustering 1, we determine the single cluster C_2 from clustering 2 into which it is best included, i.e. the one with the largest $|C_1 \cap C_2|/|C_1|$. We proceed analogously for the clusters C_2 from clustering 2. Then, for each cluster C_1 (from clustering 1), we determine its match $|C_1 \cap C_2|/|C_1 \cup C_2|$ with the *union* C_2 of clusters from clustering 2, for which C_1 is the best including cluster (as determined in the previous step). Similarly, we determine matches for clusters C_2 from clustering 2. The average match of the two clusterings is then the mean of all these matches (for all C_1 and all C_2).

The Table clearly demonstrates the necessity of using nonnegative decompositions like NMF (either as individual runs or as the final step) for obtaining reasonable results. Indeed, the best match *without* any nonnegative decompositions is 55% and the lowest relative error 0.2, whereas *with* nonnegative decompositions, we obtain a nearly perfect match (98%) with a relative error of 10^{-3}.

match (std match) relative error (std error)	Kmeans	fcm	NMF
kmeans(meta)	**0.53 (0.021)** 0.306 (0.032)	**0.55 (0.058)** 0.2 (0.018)	**0.81 (0.123)** 0.052 (0.033)
kmeans(meta) + NMF(final)	**0.62 (0.056)** 0.153 (0.059)	**0.63 (0.181)** 0.094 (0.046)	**0.9 (0.148)** 0.002 (0.001)
fcm(meta)	**0.51 (0.041)** 0.315 (0.054)	**0.53 (0.011)** 0.202 (0.019)	**0.92 (0.126)** 0.014 (0.004)
fcm(meta) + NMF(final)	**0.65 (0.178)** 0.092 (0.044)	**0.56 (0.024)** 0.112 (0.008)	**0.92 (0.126)** 0.002 (0)
NMF(meta)	**0.5 (0.032)** 0.313 (0.042)	**0.53 (0.009)** 0.194 (0.018)	**0.69 (0.008)** 0.027 (0.043)
NMF(meta) + NMF(final)	**0.59 (0.049)** 0.132 (0.016)	**0.55 (0.008)** 0.119 (0.012)	**0.74 (0.111)** 0.012 (0.025)
PTF(meta)	**0.49 (0.044)** 0.287 (0.023)	**0.53 (0.01)** 0.212 (0.019)	**0.98 (0.037)** 0.023 (0.006)
PTF(meta) + NMF(final)	**0.58 (0.04)** 0.122 (0.015)	**0.55 (0.011)** 0.116 (0.014)	**0.98 (0.043)** 0.001 (0)
Best clustering run (out of 10)	**0.49 (0.017)** 0.307 (0.011)	**0.53 (0.008)** 0.208 (0.018)	**0.76 (0.089)** 0.001 (0)

Note that *clustering* runs based on NMF are far superior to other methods. On the other hand, all tested *meta-clustering* algorithms perform reasonably well (with PTF faring best), especially in terms of relative error. However, as already discussed in Section 5, meta-clustering with NMF does not recover the clusters very well (average matches are around 74% versus virtually perfect matches for PTF (98%), 92% for fuzzy k-means and about 90% for k-means+NMF). NMF and PTF on NMF runs are also quite stable (the std of the match is 0.8% and 4% respectively).

Also note that although meta-clustering does not always outperform the best individual run in terms of *relative error*, it *does* outperform it in terms of the *match* with the original clusters (98% versus 76%).

We also considered larger problems in which the overlapping clusters can be discriminated by looking at the gene dimension only. As expected, in such cases the best results are obtained by a combination which uses NMF for meta-clustering: (NMF, NMF, NMF).

We also observed that k-means and fuzzy k-means are far inferior to NMF (as meta-clustering algorithms) in problems with a larger number of clusters. This is because, as the number of clusters increases, the fraction of perfectly reconstructed clusters *in a limited number of runs* decreases sharply. This makes meta-clustering algorithms like k-means or fuzzy k-means less effective, since these algorithms search for clusters that reoccur in a large fraction of runs. On the other hand, our approach using nonnegative decompositions looks for cluster prototypes out of which the clusters of all individual runs can be recomposed (recall Fig. 1) and therefore may behave well *even with a limited number of runs* (such as 10-20 in our experiments).

7 Related Work and Conclusions

Bradley and Fayyad [1] use *k-means* for meta-clustering a number of *k-means* runs on subsamples of the data for initializing a final k-means run. However, the use of a "hard" clustering approach like k-means in domains featuring *overlapping bi*clusters produces dramatically less accurate results than our approach using NMF or PTF for meta-clustering NMF runs (53% match and 30.6% error vs. 98% match and 0.1% error for our algorithm).[5]

The main technical contribution of this paper consists in showing how NMF and PTF can be used to solve the cluster correspondence problem for "*soft*" *bi*clustering algorithms such as NMF (which is significantly more involved than the cluster correspondence problem for "hard" algorithms and, as far as we know, has not been addressed before). The present approach is significantly different from other biclustering approaches – for example Cheng's biclustering [9] is based on a simpler additive model that is not scale invariant (problematic in the case of gene expression data). Our algorithm not only significantly outperforms all existing approaches (especially in terms of recovering the original clusters), but – more importantly – provides a conceptually elegant solution to the cluster correspondence problem. Furthermore, an initial application of the method to a large lung cancer dataset [10] proved computationally tractable and was able to perfectly recover the known histological classification of the various lung cancer types in the dataset. (For lack of space, we refer to the supplementary information at http://www.ai.ici.ro/ecml05/meyerson.pdf). The genomics applications will be the subject of future research.

Acknowledgements. I am grateful to Doina Tilivea who helped in the experimental evaluation of the algorithms.

References

1. Bradley P.S., Fayyad U.M. Refining Initial Points for K-Means Clustering, Proc. ICML-98, pp. 91-99.
2. Eisen M.B., P.T. Spellman, P.O. Brown, D. Botstein. Cluster analysis and display of genome-wide expression patterns, PNAS Vol.95, 14863-8, Dec. 1998.

[5] However, it is fair to say that [1] did not aim at improving the stability of clustering as we do, but at handling large datasets.

3. Hoyer P.O. Non-negative sparse coding. Neural Networks for Signal Processing XII, 557-565, Martigny, 2002.
4. Lee D.D., H.S. Seung. Learning the parts of objects by non-negative matrix factorization. Nature, vol. 401, no. 6755, pp. 788-791, 1999.
5. Lee D.D., H.S. Seung. Algorithms for non-negative matrix factorization. Proc. NIPS*2000, MIT Press, 2001.
6. Welling M., Weber M. Positive tensor factorization. Pattern Recognition Letters 22(12): 1255-1261 (2001).
7. Bezdek J.C. Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, 1981.
8. Bauer E. Kohavi R. An empirical comparison of voting classification algorithms: bagging, boosting, and variants. Machine Learning 36 (1999) 105-139.
9. Cheng Y. Church G. Biclustering of expression data. Proc. ISMB-2000, 93-103.
10. Bhattacharjee et al. Classification of human lung carcinomas by mRNA expression profiling reveals distinct adenocarcinoma subclasses. Proc. Natl. Acad. Sci. USA. 2001 Nov. 20;98(24):13790-5.