BIOINFORMATICS ORIGINAL PAPER

Vol. 24 no. 2 2008, pages 176–183
doi:10.1093/bioinformatics/btm562

Gene expression

Analysis of a Gibbs sampler method for model-based clustering of gene expression data

Anagha Joshi[1,2], Yves Van de Peer[1,2,*] and Tom Michoel[1,2]

[1]Department of Plant Systems Biology, VIB and [2]Department of Molecular Genetics, UGent, Technologiepark 927, 9052 Gent, Belgium

Received on July 10, 2007; revised on October 31, 2007; accepted on November 6, 2007

Advance Access publication November 22, 2007

Associate Editor: Martin Bishop

ABSTRACT

Motivation: Over the last decade, a large variety of clustering algorithms have been developed to detect coregulatory relationships among genes from microarray gene expression data. Model-based clustering approaches have emerged as statistically well-grounded methods, but the properties of these algorithms when applied to large-scale data sets are not always well understood. An in-depth analysis can reveal important insights about the performance of the algorithm, the expected quality of the output clusters, and the possibilities for extracting more relevant information out of a particular data set.

Results: We have extended an existing algorithm for model-based clustering of genes to simultaneously cluster genes and conditions, and used three large compendia of gene expression data for *Saccharomyces cerevisiae* to analyze its properties. The algorithm uses a Bayesian approach and a Gibbs sampling procedure to iteratively update the cluster assignment of each gene and condition. For large-scale data sets, the posterior distribution is strongly peaked on a limited number of equiprobable clusterings. A GO annotation analysis shows that these local maxima are all biologically equally significant, and that simultaneously clustering genes and conditions performs better than only clustering genes and assuming independent conditions. A collection of distinct equivalent clusterings can be summarized as a weighted graph on the set of genes, from which we extract fuzzy, overlapping clusters using a graph spectral method. The cores of these fuzzy clusters contain tight sets of strongly coexpressed genes, while the overlaps exhibit relations between genes showing only partial coexpression.

Availability: GaneSh, a Java package for coclustering, is available under the terms of the GNU General Public License from our website at http://bioinformatics.psb.ugent.be/software

Contact: yves.vandepeer@psb.ugent.be

Supplementary information: Supplementary data are available on our website at http://bioinformatics.psb.ugent.be/supplementary_data/anjos/gibbs

1 INTRODUCTION

Since the seminal paper by Eisen *et al.* (1998), now almost a decade ago, clustering forms the basis for extracting comprehensible information out of large-scale gene expression data sets. Clusters of coexpressed genes tend to be enriched for specific functional categories (Eisen *et al.*, 1998), share *cis*-regulatory sequences in their promoters (Tavazoie *et al.*, 1999) or form the building blocks for reconstructing transcription regulatory networks (Segal *et al.*, 2003).

A variety of heuristic clustering methods have been used, such as hierarchical clustering (Eisen *et al.*, 1998), *k*-means (Tavazoie *et al.*, 1999) or self-organizing maps (Tamayo *et al.*, 1999). Although these methods have had an enormous impact, their statistical properties are generally not well understood and important parameters such as the number of clusters are not determined automatically. Therefore, there has been a shift in attention towards model-based clustering approaches in recent years (Dahl, 2006; Fraley and Raftery, 2002; Medvedovic and Sivaganesan, 2002; Medvedovic *et al.*, 2004; Qin, 2006; Yeung *et al.*, 2001). A model-based approach assumes that the data is generated by a mixture of probability distributions, one for each cluster, and takes explicitly into account the noisyness of gene expression data. It allows for a statistical assessment of the resulting clusters and gives a formal estimate for the expected number of clusters. To infer model parameters and cluster assignments, standard statistical techniques such as Expectation Maximization or Gibbs sampling are used (Liu, 2002).

In this article, we use a novel model-based clustering method that builds upon the method recently introduced by Qin (2006). We address two key questions that have remained largely unanswered for model-based clustering methods in general, namely convergence of the Gibbs sampler for very large data sets, and non-heuristic reconstruction of gene clusters from the posterior probability distribution of the statistical model.

In the model used by Qin (2006), it is assumed that the expression levels of genes in one cluster are random samples drawn from a Gaussian distribution and expression levels of different experimental conditions are independent.We have extended this model to allow dependencies between different conditions in the same cluster. Medvedovic *et al.* (2004) used a multivariate normal distribution to take into account correlation among experimental conditions. Our approach consists of clustering the conditions within each gene cluster, assuming that the expression levels of the genes in one gene cluster for

*To whom correspondence should be addressed.

the conditions in one condition cluster are drawn from one Gaussian distribution. Hence, our model is a model for *coclustering* or *two-way clustering* of genes and conditions. The same statistical model was also used in our recent approach to reconstruct transcription regulatory networks (Michoel *et al*., 2007). The coclustering is carried out by a Gibbs sampler that iteratively updates the assignment of each gene, and within each gene cluster the assignment of each experimental condition, using the full conditional distributions of the model.

It is known that a Gibbs sampler may have poor mixing properties if the distribution being approximated is multimodal and it will then have a slow convergence rate (Liu, 2002). Previous studies of Gibbs samplers for model-based clustering have not reported convergence difficulties (Dahl, 2006; Medvedovic and Sivaganesan, 2002; Medvedovic *et al*., 2004). In those studies, only data sets with a relatively small number of genes (upto a few 100) (Medvedovic and Sivaganesan, 2002; Medvedovic *et al*., 2004), or a small number of experimental conditions (less than 10) (Dahl, 2006) were considered, and special sampling techniques such as reverse annealing (Medvedovic *et al*., 2004) or merge–split proposals (Dahl, 2006) were sufficient to generate a well mixing Gibbs sampler. We observe that for data sets of increasing size the correlation between two Gibbs sampler runs as well as the number of cluster solutions visited in one run after burn-in steadily decreases. This means that for large-scale data sets, the posterior distribution is very strongly peaked on multiple local modes. Since the peaks are so strong, we approximate the posterior distribution by averaging over multiple runs performed in parallel, each converging quickly to a single mode. By computing the correlation between different averages of the same number of runs, we are able to show that the number of distinct modes is relatively small and accurate approximations to the posterior distribution can be obtained with as few as 10 modes for around 6000 genes.

To identify the final optimal clustering, the traditional approach is to select out of all the clusterings visited by the Gibbs sampler the one that maximizes the posterior distribution [maximum a posteriori (MAP) clustering]. However, we show that for large data sets the differences in likelihood between the different local maxima are extremely small and statistically insignificant, such that the MAP clustering is as good as taking any local maximum at random. A GO (Ashburner *et al*., 2000) analysis of the different modes shows that also from the biological point of view any difference between the local modes is insignificant. Taking into account the full posterior distribution is more difficult since different clusterings may have a different number of clusters and the labeling of clusters is not unique [the label switching problem (Redner and Walker, 1984)]. The common solution to this problem is to consider pairwise probabilities for two genes being clustered together or not (Dahl, 2006; Medvedovic and Sivaganesan, 2002; Medvedovic *et al*., 2004). A major question that has not yet received a final answer is how to reconstruct gene clusters from these pairwise probabilities. Medvedovic and Sivaganesan (2002) and Medvedovic *et al*. (2004) use a heuristic hierarchical clustering on the pairwise probability matrix to form a final clustering estimate. Dahl (2006) introduces a least-squares method, which selects out of all clusterings visited by the Gibbs sampler the one which minimizes a distance function to the pairwise probability matrix. In both approaches, the probability matrix is reduced to a single hard clustering. This necessarily removes non-transitive relations between genes (such as a low probability for a pair of genes to be clustered together even though they both have relatively high probability to be clustered with the same third gene), which may nevertheless be informative and biologically meaningful.

We propose that the pairwise probability matrix reflects a *soft* or *fuzzy clustering* of the data, i.e. genes can belong to multiple clusters with a certain probability. To extract these fuzzy clusters from the pairwise probabilities, we use a method from pattern recognition theory (Inoue and Urahama, 1999). This method iteratively computes the largest eigenvalue and corresponding eigenvector of the probability matrix, constructs a fuzzy cluster with the eigenvector, and updates the probability matrix by removing from it the weight of the genes assigned to the last cluster. By only keeping genes that belong to one fuzzy cluster with very high probability, we obtain tight clusters that show higher functional coherence compared to standard clusters. Keeping also genes that belong with lower but still significant probability to multiple fuzzy clusters, we can tentatively identify multifunctional genes or relations between genes showing only partial coexpression. We show that our results are in good agreement with previous fuzzy clustering approaches to gene expression data (Gasch and Eisen, 2002). We believe that our fuzzy clustering method to summarize the posterior distribution will be of general interest for all model-based clustering approaches and solves the problems associated to heuristic clusterings of the pairwise probability matrix.

All our analyses are performed on three large-scale public compendia of gene expression data for *S. cerevisiae* (Gasch *et al*., 2000; Hughes *et al*., 2000; Spellman *et al*., 1998).

2 METHODS

2.1 Mathematical model

For an expression matrix with N genes and M conditions, we define a coclustering as a partition of the genes into K gene clusters G_k, together with for each gene cluster, a partition of the set of conditions into L_k condition clusters $\mathcal{E}_{k,l}$.We assume that all data points in a cocluster $\{(i,m): i \in G_k,\ m \in \mathcal{E}_{k,l}\}$ are random samples from the same normal distribution. This model generalizes the model used by Qin (2006), where the partition of conditions is always fixed at the trivial partition into singleton sets.

Given a set of means and precisions (μ_{kl},τ_{kl}), a coclustering $\mathcal{C}$ defines a probability density on data matrices $\mathcal{D}=(x_{im})$ by

$$p\big((\mathcal{D} \mid \mathcal{C}, \mu_{kl},\ \tau_{kl})\big) = \prod_{k=1}^{K}\prod_{l=1}^{L_k}\prod_{i\in\mathcal{G}_k}\prod_{m\in\mathcal{E}_{k,l}} p(x_{im} \mid \mu_{kl}, \tau_{kl}).$$

We use a uniform prior on the set of coclusterings with normal-gamma conjugate priors for the parameters μ_{kl} and τ_{kl}. Using Bayes' rule we find the probability of a coclustering $\mathcal{C}$ with parameters (μ_{kl}, τ_{kl}) given the data $\mathcal{D}$. Then we take the marginal probability over the parameters (μ_{kl}, τ_{kl}) to obtain the final probability of a coclustering $\mathcal{C}$ given the data $\mathcal{D}$, upto a normalization constant:

$$p(\mathcal{C} \mid \mathcal{D}) \propto \prod_{k=1}^{K}\prod_{l=1}^{L_k}\int\int p(\mu,\tau)\prod_{i\in\mathcal{G}_k}\prod_{m\in\mathcal{E}_{k,l}} p(x_{im} \mid \mu,\tau)\,\mathrm{d}\mu\mathrm{d}\tau, \qquad (1)$$

where $p(\mu,\tau)=p(\mu|\tau)p(\tau)$ with

$$p(\mu\,|\,\tau)=\left(\frac{\lambda_0\tau}{2\pi}\right)^{1/2}e^{-\frac{\lambda_0\tau}{2}(\mu-\mu_0)^2},\quad p(\tau)=\frac{\beta_0^{\alpha_0}}{\Gamma(\alpha_0)}\tau^{\alpha_0-1}e^{-\beta_0\tau},$$

$\alpha_0,\beta_0,\lambda_0>0$ and $-\infty<\mu_0<\infty$ being the parameters of the normal-gamma prior distribution. We use the values $\alpha_0=\beta_0=\lambda_0=0.1$ and $\mu_0=0.0$, resulting in a non-informative prior. We have compared the normal-gamma prior with other non-informative, conjugate priors, but found no difference in results (see Supplementary Material). The double integral in Equation 1 can be solved exactly in terms of the sufficient statistics $T_{kl}^{(n)}=\sum_{i\in\mathcal{G}_k,m\in\mathcal{E}_{kl}}x_{im}^n(n=0,1,2)$ for each cocluster. The log-likelihood or Bayesian score decomposes as a sum of cocluster scores:

$$S(\mathcal{C})=\log p(\mathcal{C}\mid\mathcal{D})=\sum_{k=1}^{K}\sum_{l=1}^{L_k}S_{kl},\tag{2}$$

with

$$S_{kl}=-\frac{1}{2}T_{kl}^{(0)}\log(2\pi)+\frac{1}{2}\log\left(\frac{\lambda_0}{\lambda_0+T_{kl}^{(0)}}\right)-\log\Gamma(\alpha_0)+\log\Gamma(\alpha_0+\frac{1}{2}T_{kl}^{(0)})+\alpha_0\log\beta_0-(\alpha_0+\frac{1}{2}T_{kl}^{(0)})\log\beta_1$$

and

$$\beta_1=\beta_0+\frac{1}{2}\Big[T_{kl}^{(2)}-\frac{(T_{kl}^{(1)})^2}{T_{kl}^{(0)}}\Big]+\frac{\lambda_0\big(T_{kl}^{(1)}-\mu_0T_{kl}^{(0)}\big)^2}{2(\lambda_0+T_{kl}^{(0)})T_{kl}^{(0)}}.$$

2.2 Gibbs sampler algorithm

We use a Gibbs sampler to sample coclusterings from the posterior distribution. The algorithm iteratively updates the assignment of genes to gene clusters, and for each gene cluster, the assignment of conditions to condition clusters as follows:

(1) Initialization: randomly assign N genes to a random K_0 number of gene clusters, and for each cluster, randomly assign M conditions to a random $L_{k,0}$ number of condition clusters.

(2) For N cycles, remove a random gene i from its current cluster. For each gene cluster k, calculate the Bayesian score $S(C_{i\to k})$, where $C_{i\to k}$ denotes the coclustering obtained from $\mathcal{C}$ by assigning gene i to cluster k, keeping all other assignments of genes and conditions equal, as well as the probability $S(C_{i\to 0})$ for the gene to be alone in its own cluster. Assign gene i to one of the possible $K+1$ gene clusters, where K is the current number of gene clusters, according to the probabilities $Q_k\propto e^{S(C_{i\to k})}$ normalized such that $\sum_k Q_k=1$.

(3) For each gene cluster k, for M cycles, remove a random condition m from its current cluster. For each condition cluster l, calculate the Bayesian score $S(C_{k,m\to l})$. Assign condition m to one of the possible L_k+1 clusters, where L_k is the current number of condition clusters for gene cluster k, according to the probabilities $Q_l\propto e^{S(C_{k,m\to l})}$ normalized such that $\sum_l Q_l=1$.

(4) Iterate steps 2 and 3 until convergence. One iteration is defined as executing steps 2 and 3 consecutively once, and hence consists of $N+K\times M$ sampling steps (with K the number of gene clusters after step 1 of that iteration).

This coclustering algorithm simulates a Markov chain that satisfies detailed balance with respect to the posterior distribution i.e. after a sufficient number of iterations, the probability to visit a particular coclustering $\mathcal{C}$ is given exactly by $p(\mathcal{C}|\mathcal{D})$. The expectation value of any real function f with respect to the posterior distribution can be approximated by averaging over the iterations of a sufficiently long Gibbs sampler run:

$$E(f)=\sum_{\mathcal{C}}f(\mathcal{C})p(\mathcal{C}\mid\mathcal{D})\approx\frac{1}{T}\sum_{t=T_0+1}^{T_0+T}f(\mathcal{C}_t)\tag{3}$$

where C_t is the coclustering visited at iteration t and T_0 is a possible burn-in period. We say that the Gibbs sampler has converged if two runs starting from different random initializations return the same averages for a suitable set of test functions f. More precisely, if $\{f_n\}$ is a set of test functions, define $a_n=E_1(f_n)$ the average of f_n in the first Gibbs sampler run, and $b_n=E_2(f_n)$ the average of f_n in the second Gibbs sampler run. We define a correlation measure $\rho\ (0\le\rho\le1)$ between two runs as

$$\rho=\frac{|\sum_n a_nb_n|}{\sqrt{(\sum_n a_n^2)(\sum_n b_n^2)}}.\tag{4}$$

Full convergence is reached if $\rho=1$.

2.3 Fuzzy clustering

To keep track of the gene clusters, independent of the (varying) number of clusters or their labeling, we consider functions

$$f_{ij}(\mathcal{C})=\begin{cases}1 & \text{if gene } i \text{ and } j \text{ belong to the same gene cluster in } \mathcal{C}\\ 0 & \text{otherwise}\end{cases}\tag{5}$$

In general, the posterior distribution is not concentrated on a single coclustering and the matrix $F=(E(f_{ij}))$ of expectation values [see Equation (3)] consists of probabilities between 0 and 1. To quantify this fuzzyness, we use an entropy measure

$$H_{\text{fuzzy}}=\frac{1}{N^2\ln 2}\sum_{ij}h(F_{ij}),\tag{6}$$

where N is the dimension of the square matrix F and

$$h(q)=-q\ln(q)-(1-q)\ln(1-q)\text{ for }0\le q\le 1.$$

For a hard clustering ($F_{ij}=0$ or 1 for all i,j), $H_{\text{fuzzy}}=0$, and for a maximally fuzzy clustering ($F_{ij}=0.5$ for all i,j), $H_{\text{fuzzy}}=1$. In reality, the matrix F is very sparse (most gene pairs will never be clustered together), so H_{fuzzy} remains small even for real fuzzy clusterings.

We assume that a fuzzy gene–gene matrix F is produced by a fuzzy clustering of the genes, i.e. we assume that each gene i has a probability p_{ik} to belong to each cluster k, such that $\sum_k p_{ik}=1$. To extract these probabilities from F we use a graph spectral method (Inoue and Urahama, 1999), originally developed for pattern recognition and image analysis, modified here to enforce the normalization conditions on p_{ik}. A fuzzy cluster is represented by a column vector $w=(w_1,\ldots,w_N)^T$, with w_i the weight of gene i in this cluster, normalized such that $\|w\|^2=w^Tw=\sum_i w_i^2=1$ The cohesiveness of the cluster with respect to the gene–gene matrix F is defined as $w^TFw=\sum_{ij}w_iF_{ij}w_j$. By the Rayleigh–Ritz theorem,

$$\max_{w\neq0}\frac{w^TFw}{w^Tw}=v_1^TFv_1=\lambda_1,$$

where λ_1 is the largest eigenvalue of F and v_1 the corresponding (normalized) eigenvector. Hence, the maximally cohesive cluster in F is given by the eigenvector of the largest eigenvalue. By the Perron–Frobenius theorem, this eigenvector is unique and all its entries are non-negative. We can then define the membership probabilities to cluster 1 by $p_{i1}=v_{1,i}/\max_j(v_{1,j})$. Hence the gene with the highest weight in v_1 is considered the prototypical gene for this cluster, and it will not belong to any other cluster. The probability p_{i1} measures to what extent other genes are coexpressed with this prototypical gene. To find the next most

cohesive cluster, we remove from F the information already contained in the first cluster by setting

$$F_{ij}^{(2)} = \sqrt{1-p_{i1}}F_{ij}\sqrt{1-p_{j1}},$$

and compute the largest eigenvalue and corresponding (normalized) eigenvector v_2 for this matrix. The prototypical gene for this cluster may already have some probability assigned to the previous cluster, so we define the membership probabilities to the second cluster by

$$p_{i2} = \min\left(\frac{v_{2,i}}{\max_j(v_{2,j})}(1-p_{i_{\max}1}), 1-p_{i1}\right).$$

Here $i_{max} = \arg\max_j(v_{2,j})$ is the prototypical gene for the second cluster, and we take the 'min' to ensure that $\sum_k p_{ik}$ will never exceed 1.

This procedure of reducing F and computing the largest eigenvalue and corresponding eigenvector to define the next cluster membership probabilities is iterated until one of the following stopping criteria is met:

(1) All entries in the reduced matrix $F^{(k)}$ reach 0, i.e. for all genes, $\sum_{k'<k} p_{ik'} = 1$, and we have completely determined all fuzzy clusters and their membership probabilities.

(2) The largest eigenvalue of the reduced matrix $F^{(k)}$ has rank >1. In this case the eigenvector is no longer unique and need no longer have non-negative entries, so we cannot make new cluster membership probabilities out of it. This may happen if the (weighted) graph defined by connecting gene pairs with non-zero entries in $F^{(k)}$ is no longer strongly connected (Perron–Frobenius theorem).

To compute one or more of the largest eigenvalues and eigenvectors for large sparse matrices such as F and its reductions $F^{(k)}$ we use efficient sparse matrix routines, such as for instance implemented in the Matlab® function `eigs`.

2.4 Data sets

We use three large compendia of gene expression data for budding yeast:

(1) Gasch *et al.* (2000) data set: expression in 173 stress-related conditions.

(2) Hughes *et al.* (2000) data set: compendium of expression profiles corresponding to 300 diverse mutations and chemical treatments.

(3) Spellman *et al.* (1998) data set: 77 conditions for alpha factor arrest, elutriation and arrest of a cdc15 temperature-sensitive mutant.

We select the genes present in all three data sets (6052 genes) and, to be as unbiased as possible, no further postprocessing is done. We use SynTReN (Van den Bulcke *et al.*, 2006) to generate simulated data sets with varying number of conditions for a synthetic transcription regulatory network with 1000 genes (see also Supplementary Material).

2.5 Functional coherence

To estimate the overall biological relevance of the clusters, we use a method that calculates the mutual information between clusters and GO attributes (Gibbons and Roth, 2002). For each GOslim attribute, we create a cluster–attribute contingency table where rows are clusters and columns are attribute status ('*Yes*') if the gene possesses the attribute, '*No*' if it is not known whether the gene possesses the attribute). The total mutual information is defined as the sum of mutual informations between clusters and individual GO attributes:

$$MI = \sum_A H(\mathcal{C}) + H(A) - H(\mathcal{C}, A) \quad (7)$$

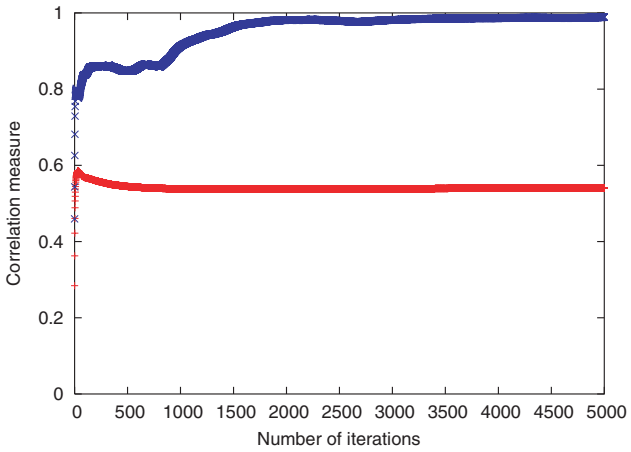

Fig. 1. Trace plot of the correlation measure ρ between two different Gibbs sampler runs as a function of the number of iterations, for a small data set (100 genes, 10 conditions, top curve) and a large data set (1000 genes, 173 conditions, bottom curve). Both data sets are subsets of the Gasch *et al.* data set.

where $\mathcal{C}$ is a clustering of the genes, A is a GO attribute and H is Shannon's entropy, $H = -\sum_i p_i \log(p_i)$, and the p_i are probabilities obtained from the contingency tables.

3 RESULTS AND DISCUSSION

3.1 Convergence of the Gibbs sampler algorithm

We study convergence using the test functions f_{ij}, which indicate if gene i and j are clustered together or not [see Equation (5) in the Methods section] and compute the correlation measure ρ between different runs for this set of functions (see Equation (4) in the Methods section). In addition to the correlation measure, we also compute the entropy measure H_{fuzzy} (see Equation (6) in the Methods section). This parameter summarizes the 'shape' of the posterior distribution: a value of 0 corresponds to hard clustering that implies that the distribution is completely supported on a single solution, the more positive H_{fuzzy} is, the more the distribution is supported on multiple solutions.

In the analysis below, we use subsets from the Gasch *et al.* data set with a varying number of genes and conditions and perform multiple Gibbs sampler runs with a large number of iterations. One iteration involves a reassignment of all genes and all conditions in all clusters, and hence involves $N + M \times K$ sampling steps in the Gibbs sampler, where N is the number of genes, M the number of conditions and K the number of clusters at that iteration (typically $K \sim \sqrt{N}$).

First we consider a very small data set (100 genes, 10 conditions). We start two Gibbs sampler runs in parallel and compute the correlation measure ρ at each iteration, see Figure 1. In this case, ρ approaches its maximum value $\rho = 1$ in less than 5000 iterations and the Gibbs sampler generates a well mixing chain that can easily explore the whole space. Non-zero values of the entropy measure H_{fuzzy} (0.105 ± 0.003) indicate

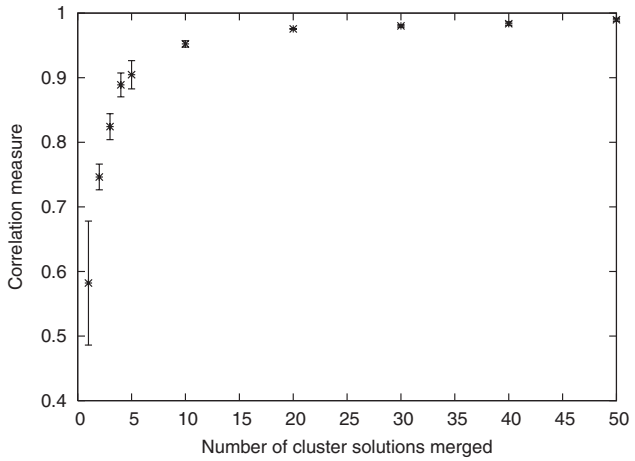

Fig. 2. Correlation measure ρ between different averages of the same number of local maxima for a data set of 1000 genes and 173 conditions (subset of the Gasch *et al*. data set).

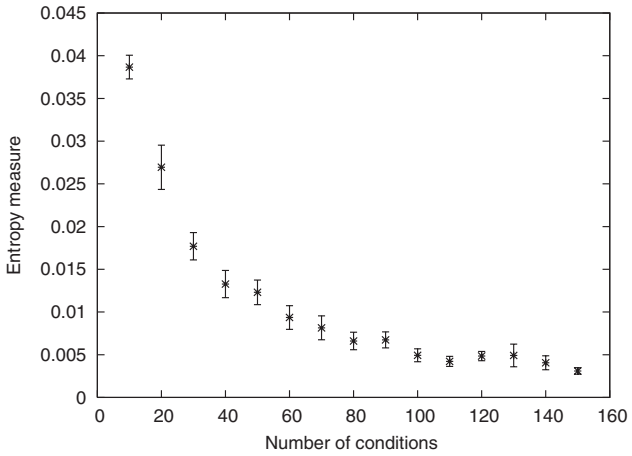

Fig. 3. Entropy measure H_{fuzzy} for data sets with 1000 genes and varying number of conditions (subsets of the Gasch *et al*. data set).

that the posterior distribution is supported on multiple clusterings of the genes.

Next, we run the Gibbs sampler algorithm on a data set with 1000 genes and all 173 conditions. Unlike in the previous situation we observe that the correlation between two Gibbs sampler runs saturates well below 1 (see Fig. 1). Hence, the Gibbs sampler does not converge to the posterior distribution in one run.We can gain further understanding for the lack of convergence by looking in more detail at a single Gibbs sampler run. It turns out that the correlation measure between two successive iterations reaches 1 very rapidly and remains unchanged afterwards (see Supplementary Fig. 2). Since each iteration involves a large number of sampling steps (i.e. a large number of possible configuration changes), this implies that the Gibbs sampler very rapidly finds a local maximum of the posterior distribution from which it can no longer escape. We conclude that the posterior distribution is supported on multiple local maxima that overlap only partially, and with valleys in between that cannot be crossed by the Gibbs sampler. These local maxima all have approximately the same log-likelihood (see for instance the small variance in Fig. 4 below) and are therefore all equally meaningful. The probability ratio between peaks and valleys is so large (exponential in the size of the data set) that an accurate approximation to the posterior distribution is given by averaging over the local maxima only. Those can be uncovered by performing multiple independent runs, each converging very quickly on one of the maxima, and there is no need for special techniques to also sample in between local maxima. The number of local maxima (Gibbs sampler runs) necessary for a good approximation can be estimated as follows. We perform 150 independent Gibbs sampler runs and compute for each the pairwise gene–gene clustering probability matrix F (see Methods section). For each $k = 1, \ldots, 50$, we take two non-overlapping sets of k solutions and compute the average of their pairwise probability matrices F. Then, we compute the correlation measure ρ between those two averages. This is repeated several times, depending on the number of non-overlapping sets that can be chosen from the pool of 150 solutions. If for a given k the correlation is always 1, then there are at most k local maxima. Figure 2 shows that as k increases, the correlation quickly reaches close to this perfect value 1. This implies that the number of local maxima is not too large and a good approximation to the posterior distribution can be obtained in this case already with 10 to 20 solutions. Supplementary Figure 1 shows an example of hard clusters formed as a result of a single run and fuzzy clusters formed by merging the result of 10 independent runs.

In Figure 3, we keep the same 1000 genes and select an increasing number of conditions. As the data set increases, the entropy measure H_{fuzzy} decreases, meaning the clusters become increasingly hard. Simultaneously, the correlation measure ρ decreases from about 0.85 to 0.55 (see Supplementary Fig. 3). We conclude that the depth of the valleys between different local maxima of the posterior distribution increases with the size of the data set, and it becomes increasingly more difficult for the Gibbs sampler to escape from these maxima and visit the whole space in one run.

3.2 Analysis of whole genome data sets

If we run the Gibbs sampler algorithm on the three whole genome yeast data sets, we are in the situation where the algorithm very rapidly gets stuck in a local maximum. In Figure 4, we plot the average Bayesian log-likelihood score [see Equation (2) in the Methods section] for 10 different Gibbs sampler runs for the Spellman *et al*. data set. The rapid convergence of the log-likelihood shows that the Gibbs sampler reaches the local maxima very quickly and the low variance shows that the different local maxima are all equally likely. The average over 10 runs of the GO mutual information score (see Equations (7) in the Methods section) shows the same rapid convergence and small variance (see Supplementary Fig. 6), implying that the different maxima are biologically equally meaningful according to this score. The correlation between different averages of 10 Gibbs sampler runs reaches 0.85, a value we consider high enough for a good approximation of the

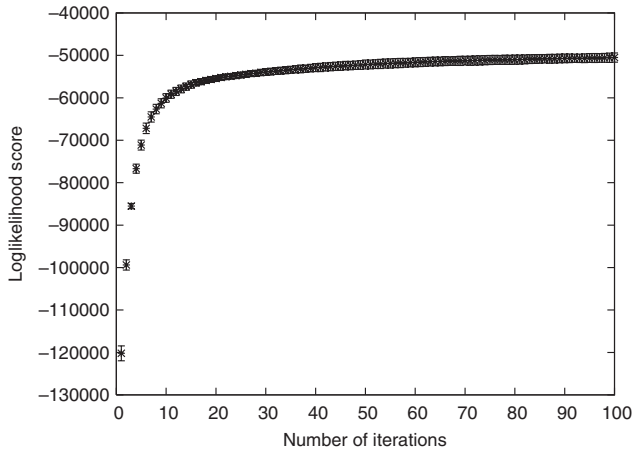

Fig. 4. Trace plot of the average log-likelihood score and SD for 10 Gibbs sampler runs for the Spellman *et al*. data set.

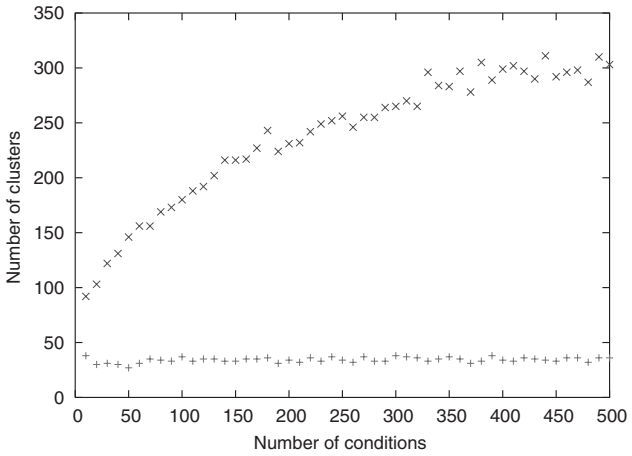

Fig. 5. Number of gene clusters for a simulated data set with 1000 genes and a varying number of conditions, for two-way clustering [top data points (×)] and one-way clustering [bottom data points (+)].

Table 1. One-way clustering, averages for 10 different Gibbs sampler runs

Data set	Average K	Average log-likelihood score	Average MI
Gasch *et al*.	52.9 (2.6)	$-6.101\ (0.014) \times 10^5$	1.771 (0.031)
Hughes *et al*.	14.9 (0.5)	$2.530\ (0.002) \times 10^6$	0.588 (0.044)
Spellman *et al*.	49.7 (2.2)	$-7.183\ (0.037) \times 10^4$	1.491 (0.032

Table 2. Two-way clustering, averages for 10 different Gibbs sampler runs

Data set	Average K	Average log-likelihood score	Average MI
Gasch *et al*.	84.5 (2.5)	$-5.586\ (0.012) \times 10^5$	1.912 (0.033)
Hughes *et al*.	85.5 (2.7)	$2.798\ (0.004) \times 10^6$	1.511 (0.045)
Spellman *et al*.	65.4 (4.2)	$-5.112\ (0.011) \times 10^4$	1.612 (0.032)

posterior distribution.The other two data sets show precisely the same behavior (see Supplementary Figs 4 and 5).

3.3 Two-way clustering *versus* one-way clustering

Our coclustering algorithm extends the CRC algorithm of Qin (2006) by also clustering the conditions for each cluster of genes (*'two-way clustering'*), instead of assuming they are always independent (*'one-way clustering'*). We compare the clustering of genes for the three yeast data sets using both methods, by computing the average number of clusters inferred (K), the average log-likelihood score and the average GO mutual information score for 10 independent runs of each algorithm. The results are tabulated in Tables 1 and 2. For all three data sets, both the log-likelihood score and the GO mutual information score are higher (better) for our method. The increase in GO mutual information score is especially significant in case of the Hughes *et al*. data set. This data set has very few overexpressed or repressed values and if each condition is considered independent, there are very few distinct profiles that result in the formation of very few clusters (∼15 for 6052 genes). Also clustering the conditions gives more meaningful results since differentially expressed conditions form separate clusters from one large background cluster of non-differentially expressed conditions.

For simulated data sets, clusters are defined as sets of genes sharing the same regulators in the synthetic regulatory network, and the true number of clusters is known. Here, we consider a gene network whose topology is subsampled from an *Escherichia coli* transcriptional network (Van den Bulcke *et al*., 2006) with 1000 genes, of which 105 transcription factors, and 286 clusters. For two-way clustering, as we increase the number of conditions in the simulated data set, more clusters are formed and the number of clusters saturates close to the true number (see Fig. 5). For one-way clustering, addition of conditions does not affect the inferred number of clusters which is anorder of magnitude smaller than the true number (see Fig. 5). For two-way clustering, due to the clustering of conditions, the number of model parameters is reduced, and greater statistical accuracy can be achieved, even when the number of genes in a cluster becomes small. The correlation measure ρ between true clusters and inferred clusters also shows ahigher value for two-way clustering over one-way (see Supplementary Fig. 8).

Unlike for simulated data sets, the inferred number of clusters does not depend much upon the number of conditions for real biological data sets (see Supplementary Fig. 7), i.e. even if more conditions are added, the algorithm does not generate more clusters. This is because in simulated data, every addition of a condition adds new information, but for real data sets that might not be the case. In order to get the true clusters from the expression data, we do not only need more conditions but also

Table 3. Number of genes clustered and number of genes belonging to multiple clusters with different membership probability cutoff values

Data set	0.1	0.3	0.5
Gasch *et al.*	6045 (4356)	4062 (344)	1781 (0)
Hughes *et al.*	6052 (4554)	3959 (34)	2254 (0)
Spellman *et al.*	6052 (5187)	3158 (139)	1255 (0)

that each new condition contributes information different from the information already available from the previous conditions. This might be a reason why the algorithm clusters 6052 genes in only ~80 clusters (see Table 2).

3.4 Fuzzy clusters

Our algorithm returns a summary of the posterior distribution in the form of a gene–gene matrix whose entries are the probabilities that a pair of genes is clustered together. To convert these pairwise probabilities back to clusters, we use a graph spectral method as explained in the Methods section. The method produces fuzzy overlapping clusters, where each gene i belongs to each fuzzy cluster k with a probability p_{ik}, such that $\sum_k p_{ik} = 1$. The size of a fuzzy cluster k is defined as $\sum_i p_{ik}$. The algorithm iteratively produces new fuzzy clusters until all the information in the pairwise matrix is converted into clusters (first stopping criterium, see Methods section), or until the mathematical conditions underlying the algorithm cease to hold (second stopping criterium, see Methods section). We applied the algorithm to pairwise probability matrices for each of the three data sets, obtained by averaging over 10 different Gibbs sampler runs. For the Gasch *et al.* and Hughes *et al.* data sets, full fuzzy clustering is achieved with 500 fuzzy clusters (all 6052 genes have total assignment probability $\sum_k p_{ik} > 0.98$). For the spellman *et al.* data set, the second stopping criterium is met after producing 321 fuzzy clusters

In general, we observe that the algorithm first produces one very large fuzzy cluster corresponding to an average expression profile that almost all genes can relate to. This cluster is of no interest for further analysis. Then, it produces a number of fuzzy clustersof varying size, which show interesting coexpression profiles and are useful for further analysis. For the three data sets considered here, this number is around 100, consistent with the average number of clusters in different Gibbs sampler runs (see Table 2). The remaining fuzzy clusters are typically very small and consist mostly of noise. Like the very first cluster, they are of no interest for further analysis.

Since every gene belongs to every cluster, we use a probability cutoff to remove from each cluster the genes that belong to it with a very small probability. The smaller the cutoff, the more genes belong to a cluster, which results into more fuzzy clusters and vice versa. Table 3 shows the total number of genes assigned to at least one fuzzy cluster with different cutoff values and in brackets the number of genes assigned to at least two fuzzy clusters.

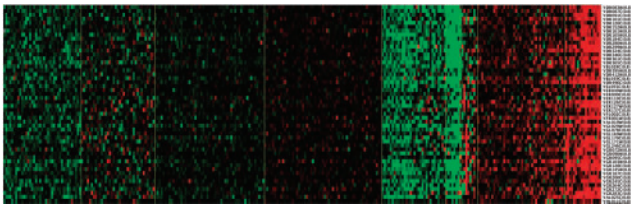

Fig. 6. Ribosomal genes form a tight cluster in the Hughes *et al.* data set. (Due to space constraints only the first few genes are shown; for the complete figure, see the Supplementary Material.)

Fig. 7. Fourgenes GAL1, GAL2, GAL7 and GAL10 form a tight cluster showing conditional coexpression in the Gasch *et al.* data set.

The goal of merging different Gibbs sampler solutions and forming fuzzy clusters is to extract additional information out of a data set that is not captured by a single hard clustering solution. This can be achieved in two ways. First, by obtaining tight clusters of few but highly coexpressed genes with a high probability cutoff. Second, by characterizing genes that belong to multiple clusters with a significant probability.

For all three data sets, at a probability cutoff of 0.5, we get a subset of genes that belong to only one cluster with high probability. Table 3 shows that each data set retains at least 20% of its genes. These are sets of strongly coexpressed genes that cluster together in almost every hard cluster solution. Ribosomal genes show such a strong coexpression pattern in all the three data sets where most genes belong to this cluster with a probability close to 1 (see Fig. 6). At least 75% of all the genes in cluster 2 (Gasch *et al.*, data), cluster 3 (Hughes *et al.*, data) and cluster 2 (Spellman *et al.*, data) are located in ribosome.

Local but very strong coexpression patterns can also be detected by our method. Cluster 15 of the Gasch *et al.* data set consists of only 4 genes clustered together with probability 1 (see Fig. 7). These four genes, GAL1, GAL2, GAL7 and GAL10, are enzymes in the galactose catabolic pathway and respond to different carbon sources during steady state. They are strongly upregulated when galactose is used as a carbon source (second experiment cluster in Fig. 7) and strongly downregulated with any other sugar as a carbon source (first experiment cluster in Fig. 7). In every hard cluster solution, these 4 genes are clustered together along with other genes. By merging these hard cluster solutions to form fuzzy clusters, we get a tight but more meaningful cluster with only 4 genes.

Table 3 shows that many genes belong to two or more clusters with a significant probability. For the Gasch *et al.* data set, we find similar observations as in Gasch and Eisen (2002). Cluster 27 contains genes localized in endoplasmic reticulum (ER) and induced under dithiothreitol (DTT) stress like FKB2, JEM1, ERD2, ERP1, ERP2, RET2, RET3, SEC13, SEC21, SEC24 and others. Cluster 34 contains genes repressed under nitrogen stress and stationary state. Twenty percent of the genes in cluster 27 also belong to cluster 34 with a significant membership. These include genes encoding for ER vesicle coat

proteins like RET2, RET3, SEC13 and others that are induced under DTT stress as well as repressed under nitrogen stress and stationary state. Also RIO1, an essential serine kinase, belongs to two clusters with a significant probability. It clusters with genes involved in ribosomal biogenesis and assembly (Gasch *et al*., data cluster 3) as well as with genes functioning as generators of precursor metabolites and energy (Gasch *et al*., data cluster 7). We find similar observations for the Hughes *et al*. and Spellman *et al*. data sets. Genes CLN1, CLN2 and other DNA synthesis genes like CLB6, which are known to be regulated by SBF during S1 phase (Koch *et al*., 1996) belong to cluster 19 (Spellman *et al*., data). They also belong with significant probability to cluster 4 (Spellman *et al*., data). More than one-third of the genes in cluster 4 are predicted to be cell-cycle regulated genes.

4 CONCLUSION

We have developed an algorithm to simultaneously cluster genes and conditions and sample such coclusterings from a Bayesian probabilistic model. For large data sets, the model is supported on multiple equivalent local maxima. The average of these local maxima can be represented by a matrix of pairwise gene–gene clustering probabilities and we have introduced a new method for extracting fuzzy, overlapping clusters from this matrix. This method is able to extract information out of the data set that is not available from a single, hard clustering.

ACKNOWLEDGEMENTS

We thank Steven Maere and Vanessa Vermeirssen for helpful discussions. Early Stage Marie Curie Fellowship to A.J.; Postdoctoral Fellowship of the Research Foundation Flanders (Belgium) to T.M.

Conflict of Interest: none declared.

REFERENCES

Ashburner,M. *et al*. (2000) Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nat. Genet*., **25**, 25–29.

Dahl,D.B. (2006) Model-based clustering for expression data viaa Dirichlet process mixture model. In Do,K.-A. *et al*. (eds.) *Bayesian inference for gene expression and proteomics*. Cambridge University Press, New York, pp. 201–218.

Eisen,M.B. *et al*. (1998) Cluster analysis and display of genome-wide expression patterns. *Proc. Natl Acad. Sci. USA*., **95**, 14863–14868.

Fraley,C. and Raftery,A.E. (2002) Model-based clustering, discriminant analysis, and density estimation. *J. Am. Stat. Assoc*., **97**, 611–631.

Gasch,A.P. and Eisen,M.B. (2002) Exploring the conditional coregulation of yeast gene expression through fuzzy k-means clustering. *Genome Biol*, **3**, RESEARCH0049.

Gasch,A.P. *et al*. (2000) Genomic expression programs in the response of yeast cells to environmental changes. *Mol. Biol. Cell*, **11**, 4241–4257.

Gibbons,F.D. and Roth,F.P. (2002) Judging the quality of gene expression-based clustering methods using gene annotation. *Genome Res*., **12**, 1574–1581.

Hughes,T.R. *et al*. (2000) Functional discovery via a compendium of expression profiles. *Cell*, **102**, 109–126.

Inoue,K. and Urahama,K. (1999) Sequential fuzzy cluster extraction by a graph spectral method. *Pattern Recognit. Lett*., **20**, 699–705.

Koch,C. *et al*. (1996) Switching transcription on and off during the yeast cell cycle: ClnCdc28 kinases activate bound transcription factor SBF Swi4/Swi6 at start, whereas Clb/Cdc28 kinases displace it from the promoter in G2. *Genes Dev*., **10**, 129–141.

Liu,J.S. (2002) *Monte Carlo Strategies in Scientific Computing*. Springer, New York.

Medvedovic,M. and Sivaganesan,S. (2002) Bayesian infinite mixture model based clustering of gene expression profiles. *Bioinformatics*, **18**, 1194–1206.

Medvedovic,M. *et al*. (2004) Bayesian mixture model based clustering of replicated microarray data. *Bioinformatics*, **20**, 1222–1232.

Michoel,T. *et al*. (2007) Validating module network learning algorithms using simulated data. *BMC Bioinformatics*, **8** (Suppl. 2), S5.

Qin,Z.S. (2006) Clustering microarray gene expression data using weighted Chinese restaurant process. *Bioinformatics*, **22**, 1988–1997.

Redner,R.A. and Walker,H.F. (1984) Mixture densities, maximum likelihood, and the EM algorithm. *SIAM Review*, **26**, 195–239.

Segal,E. *et al*. (2003) Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nat. Genet*., **34**, 166–167.

Spellman,P.T. *et al*. (1998) Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. *Mol. Biol. Cell*, **9**, 3273–3297.

Tamayo,P. *et al*. (1999) Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation. *Proc. Natl Acad. Sci. USA*., **96**, 2907–2912.

Tavazoie,S. *et al*. (1999) Systematic determination of genetic network architecture. *Nat. Genet*., **22**, 281–285.

Van den Bulcke,T. *et al*. (2006) SynTReN: a generator of synthetic gene expression data for design and analysis of structure learning algorithms. *BMC Bioinformatics*, **7**, 43.

Yeung,K.Y. *et al*. (2001) Model-based clustering and data transformations for gene expression data. *Bioinformatics*, **17**, 977–987.