Wang and Wang *EURASIP Journal on Bioinformatics and Systems Biology* 2013, **2013**:5
http://bsb.eurasipjournals.com/content/2013/1/5

EURASIP Journal on Bioinformatics and Systems Biology
a SpringerOpen Journal

RESEARCH Open Access

Hierarchical Dirichlet process model for gene expression clustering

Liming Wang[1] and Xiaodong Wang[2*]

Abstract

Clustering is an important data processing tool for interpreting microarray data and genomic network inference. In this article, we propose a clustering algorithm based on the hierarchical Dirichlet processes (HDP). The HDP clustering introduces a hierarchical structure in the statistical model which captures the hierarchical features prevalent in biological data such as the gene express data. We develop a Gibbs sampling algorithm based on the Chinese restaurant metaphor for the HDP clustering. We apply the proposed HDP algorithm to both regulatory network segmentation and gene expression clustering. The HDP algorithm is shown to outperform several popular clustering algorithms by revealing the underlying hierarchical structure of the data. For the yeast cell cycle data, we compare the HDP result to the standard result and show that the HDP algorithm provides more information and reduces the unnecessary clustering fragments.

1 Introduction

The microarray technology has enabled the possibility to monitor the expression levels of thousands of genes in parallel under various conditions [1]. Due to the high-volume nature of the microarray data, one often needs certain algorithms to investigate the gene functions, regulation relations, etc. Clustering is considered to be an important tool for analyzing the biological data [2-4]. The aim of clustering is to group the data into disjoint subsets, where in each subset the data show certain similarities to each other. In particular, for microarray data, genes in each clustered group exhibit correlated expression patterns under various experiments.

Several clustering methods have been proposed, most of which are distance-based algorithms. That is, a distance is first defined for clustering purpose and then the clusters are formed based on the distances of the data. Typical algorithms in this category include the K-means algorithm [5] and the self-organizing map (SOM) algorithm [6]. These algorithms are based on simple rules, and they often suffer from robustness issue, i.e., they are sensitive to noise which is extensive in biological data [7]. For example, the SOM algorithm requires user to provide number of clusters in advance. Hence, incorrect estimation of the parameter may provide wrong result.

Another important category of clustering methods is the model-based algorithms. These algorithms employ a statistical approach to model the structure of clusters. Specifically, data are assumed to be generated by some mixture distribution. Each component of the mixture corresponds to a cluster. Usually, the parameters of the mixture distribution are estimated by the EM algorithm [8]. The finite-mixture model [9-11] assumes that the number of mixture components is finite and the number can be estimated using the Bayesian information criterion [12] or the Akaike information criterion [13]. However, since the estimation of the number of clusters and the estimation of the mixture parameters are performed separately, the finite-mixture model may be sensitive to the different choices of the number of clusters [14].

The infinite-mixture model has been proposed to cope with the above sensitivity problem of the finite-mixture model. This model does not assume a specific number of components and is primarily based on the Dirichlet processes [15,16]. The clustering process can equivalently be viewed as a Chinese restaurant process [17], where the data are considered as customers entering a restaurant. Each component corresponds to a table with infinite capacity. A new customer joins a table according to the current assignment of seats.

*Correspondence: wangx@ee.columbia.edu
[2]Department of Electrical Engineering, Columbia University, New York, NY 10027, USA
Full list of author information is available at the end of the article

Hierarchical clustering (HC) is yet another more advanced approach especially for biological data [18], which groups together the data with similar features based on the underlying hierarchical structure. The biological data often exhibit hierarchical structure, e.g., one cluster may highly be overlapped or could be embedded into another cluster [19]. If such hierarchical structure is ignored, the clustering result may contain many fragmental clusters which could have been combined together. Hence, for biological data, such HC has its advantages to many traditional clustering algorithms. The performances of such HC algorithms depend highly on the quality of the data and the specific agglomerative or divisive ways the algorithms use for combining clusters.

Traditional clustering algorithms for microarray data usually assign each gene with a feature vector formed by the expressions in different experiments. The clustering is carried out for these vectors. It is well known that many genes share different levels of functionalities [20]. The resemblances of different genes are commonly represented at different levels of perspectives, e.g., at the cluster level instead of individual gene level. In other words, The relationships among different genes may vary during different experiments. In Figure 1, we illustrate the gene hierarchical structures for microarray data. Genes group A and B may show close relationship to genes group C in some experiments. While the genes group D shows correlations to groups A, B, and C in other experiments. The group D obviously has a hierarchical relationships to other gene groups. In this case, we desire to have a HC algorithm recognizing the gene resemblances not at the single gene level but at the higher cluster level, to avoid unnecessary fragmental clusters that impede the proper interpretation of the biological information. Such a HC algorithm may also provide new information by taking the hierarchical similarities into account.

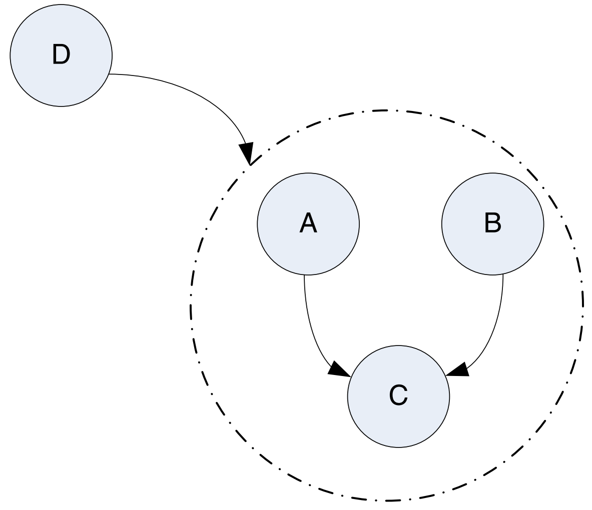

Figure 1 Illustration of gene hierarchical structures in microarray data.

In this article, we propose a model-based clustering algorithm for gene expression data based on the hierarchical Dirichlet process (HDP) [21]. The HDP model incorporates the merits of both the infinite-mixture model and the HC. The hierarchical structure is introduced to allow sharing data among related clusters. On the other hand, the model uses the Dirichlet processes as the nonparametric Bayesian prior, which do not assume a fixed number of clusters *a priori*.

The remainder of the article is organized as follows. In Section 2, we introduce some necessary mathematical background and formulate the HC problem as a statistical inference problem. In Section 3, we derive a Gibbs sampler-based inference algorithm based on the Chinese restaurant metaphor of the HDP model. In Section 4, we provide experimental results of the proposed HDP algorithm for two applications, regulatory network segmentation and gene expression clustering. Finally, Section 5 concludes the article.

2 System model and problem formulation

As in any model-based clustering method, it is assumed that the gene expression data are random samples from some underlying distributions. All data in one cluster are generated by the same distribution. For most existing clustering algorithms, each gene is associated with a vector containing the expressions in all experiments. The clustering of the genes is based on their vectors. However, such approach ignores the fact that genes may show different functionalities under various experiment conditions, i.e., different clusters may be formed under different experiments. In order to cope with this phenomenon, we treat each expression separately. More specifically, we allow different expressions of the same individual gene to be generated by different statistical models.

Suppose that for the mircoarray data, there are N genes in total. For each gene, we conduct M experiments. Let g_{ji} denote the expression of the ith gene in the jth experiment, $1 \leq i \leq N$, and $1 \leq j \leq M$. For each g_{ji}, we associate a latent membership variable z_{ji}, which indicates the cluster membership of g_{ji}. That is, if genes i and i' are in the same cluster under the conditions of experiments j and j', we have $z_{ji} = z_{j'i'}$. Note that z_{ji} is supported on a countable set such as $\mathbb{N}$ or $\mathbb{Z}$. For each g_{ji}, we associate a coefficient $\theta_{z_{ji}}$, whose index is determined by its membership variable z_{ji}. In order to have a Bayesian approach, we also assume that each coefficient θ_k is drawn independently from a prior distribution G_0

$$\theta_k \sim G_0, \tag{1}$$

where k is determined by z_{ji}.

The membership variable $\mathbf{z} = \{z_{ji}\}_{j,i}$ has a discrete joint distribution

$$\mathbf{z} \sim \pi. \tag{2}$$

Note that in this article, the bold-face letter always refers to a set formed by the elements with specified indices.

We assume that each g_{ji} is drawn independently from a distribution $F(\theta_{z_{ji}})$

$$g_{ji} \sim F\left(\theta_{z_{ji}}\right), \tag{3}$$

where $\theta_{z_{ji}}$ is a coefficient associated with g_{ji} and F is a distribution family such as the Gaussian distribution family. In summary, we have the following model for the expression data

$$\begin{aligned} \theta_k \quad &\sim G_0 \\ \mathbf{z} \quad &\sim \pi \\ g_{ji}|z_{ji}, \theta_k \ &\sim F\left(\theta_{z_{ji}}\right). \end{aligned} \tag{4}$$

The above model is a relatively general one which can induce many previous models. For example, in all Bayesian approaches, all variables are assigned with proper priors. It is very popular to use the mixture model as the prior, which models the data generated by a mixture of distributions, e.g., a linear combination of a family of distributions such as Gaussian distributions. Each cluster is generated by one component in the mixture distribution given the membership variable [14]. The above approach corresponds to our model if we assume that π is finitely supported and F is Gaussian.

The aim for clustering is to determine the posterior probability of the latent membership variables given the observed gene expressions

$$P(\mathbf{z}|\mathbf{g}), \tag{5}$$

where $\mathbf{g} = \{g_{ji}\}_{j,i}$.

As a clustering algorithm, the final result is given in the forms of clusters. Each gene has to be assigned to one and only one cluster. Once we have the inference result in (5), we can apply the maximum *a posterior* criterion to obtain an estimate of membership variable $\hat{z}_{\cdot i}$ for the ith gene as

$$\hat{z}_{\cdot i} = \arg_a \max \sum_j P(z_{ji} = a|\mathbf{g}). \tag{6}$$

We note that in case one is interested in finding other related clusters for one gene, we can simply use the inferred distribution to membership variable to obtain this information.

2.1 Dirichlet processes and infinite mixture model

Instead of assuming a fixed number of clusters *a priori*, one can assume infinite number of clusters to avoid the estimation accuracy problem on the number of clusters as we mentioned earlier. Correspondingly in (4), the prior π is an infinite discrete distribution. Again as in the Bayesian fashion, we will introduce priors for all parameters. The Dirichlet process is one such prior. It can be viewed as a random measure [15], i.e., the domain of this process (viewed as a measure) is a collection of probability measures. In this section, we will give a brief introduction to the Dirichlet process which serves as the vital prior part in our HDP model.

Recall that the Dirichlet distribution $\mathcal{D}(u_1, \ldots, u_K)$ of order K on a $(K-1)$-simplex in $\mathbb{R}^{K-1}$ with parameter $u_1, \ldots, u_K$ is given by the following probability density function

$$\mathcal{D}(x_1, \ldots, x_{K-1}; u_1, \ldots, u_K) = \frac{\Gamma\left(\sum_{i=1}^{K} u_i\right)}{\prod_{i=1}^{K} \Gamma(u_i)} \prod_{i=1}^{K} {x_i}^{u_i - 1} \tag{7}$$

where $\sum_{i=1}^{K} x_i = 1, u_i > 0, i = 1, \ldots, K$, and $\Gamma(\cdot)$ is the Gamma function. Since every point in the domain is a discrete probability measure, the Dirichlet distribution is a random measure in the finite discrete probability space.

The Dirichlet processes are the generalization of the Dirichlet distribution into the continuous space. There are various constructive or non-constructive definitions of Dirichlet processes. For simplicity, we use the following non-constructive definition.

Let (X, σ, μ_0) be a probability space. A Dirichlet process $D(\alpha_0, \mu_0)$ with parameter $\alpha_0 > 0$ is defined as a random measure: for any non-trivial finite partition $(\chi_1, \ldots, \chi_r)$ of X with $\chi_i \in \sigma$, we have the random variable

$$(\mathcal{G}(\chi_1), \ldots, \mathcal{G}(\chi_r)) \sim \mathcal{D}(\alpha_0\mu_0(\chi_1), \ldots, \alpha_0\mu_0(\chi_r)), \tag{8}$$

where $\mathcal{G}$ is drawn from $D(\alpha_0, \mu_0)$.

The Dirichlet processes can be characterized in various ways [15] such as the stick-breaking construction [22] and the Chinese restaurant process [23]. The Chinese restaurant process serves as a visualized characterization of the Dirichlet process.

Let $x_1, x_2, \ldots$ be a sequence of random variables drawn from the Dirichlet process $D(\alpha_0, \mu_0)$. Although we do not have the explicit formula for D, we would like to know the conditional probability of x_i given $x_1, \ldots, x_{i-1}$. In the Chinese restaurant model, the data can be viewed as customers sequentially entering a restaurant with infinite number of tables. Each table corresponds to a cluster with unlimited capacity. Each customer x_i entering the restaurant will join in the table already taken with equal probability. In addition, the new customer may sit in a new table with probability proportional to α_0. Tables that have already been occupied by customers tend to gain more and more customers.

One remarkable property of the Dirichlet process is that although it is generated by a continuous process, it is discrete (countably many) almost surely [15]. In other words, almost every sample distribution drawn from the Dirichlet process is a discrete distribution. As a consequence, the Dirichlet process is suitable to serve as a non-parametric prior of the infinite mixture model.

The Dirichlet mixture model uses the Dirichlet process as a prior. The model in (4) can then be represented as follows:

$$g_{ji}|z_{ji}, \theta_k \sim F(\theta_{z_{ji}}); \tag{9}$$

θ_k is generated by the measure μ_0

$$\theta_k \sim \mu_0; \tag{10}$$

$\{z_{ji}\}$ is generated by a Dirichlet process $D(\alpha_0, \mu_0)$

$$\{z_{ji}\} \sim D(\alpha_0, \mu_0). \tag{11}$$

Recall that $D(\alpha_0, \mu_0)$ is discrete almost everywhere, which corresponds to the indices of the clusters.

2.2 HDP model

Biological data such as the expression data often exhibit hierarchical structures. For example, although clusters can be formed based on similarities, some clusters may still share certain similarities among themselves at different levels of perspectives. Within one cluster, the genes may share similar features. But on the level of clusters, one cluster may share some similar feature with some other clusters. Many traditional clustering algorithms typically fail to recognize such hierarchical information and are not able to group these similar clusters into a new cluster, producing many fragments in the final clustering result. As a consequence, it is difficult to interpret the functionalities and meanings of these fragments. Therefore, it is desirable to have an algorithm that is able to cluster among clusters. In other words, the algorithm should be able to cluster based on multiple features at different levels. In order to capture the hierarchical structure feature of the gene expressions, we now introduce the hierarchical model to allow clustering at different levels. The clustering algorithm based on the hierarchical model not only reduces the number of cluster fragments, but also may reveal more details about the unknown functionalities of certain genes as the clusters sharing multiple features.

Recall that in the statistical model (11), the clustering effect is induced by the Dirichlet process $D(\alpha_0, \mu_0)$. If we need to take into account different level of clusters, it is natural to introduce a prior with clustering effect to the base measure μ_0. Again in this case, the Dirichlet process can serve as such prior. The intuition is that given the base measure, the clustering effect is represented through a Dirichlet process on the single gene level. By the Dirichlet process assumption on the base measure, the base measure also exhibits the clustering effect, which leads to clustering at cluster level. We simply set the prior to the base measure μ_0 as

$$\mu_0 \sim D_1(\alpha_1, \mu_1), \tag{12}$$

where $D_1(\alpha_1, \mu_1)$ is another Dirichlet process. In this article, we use the same letter for the measure, the distribution it induces, and the corresponding density function as long as it is clear from the context. Moreover, we could extend the hierarchies to as many levels as we wish at the expense of complexity of the inference algorithm. The desired number of hierarchies can be determined by the prior biological knowledge. In this article, we focus on a two-level hierarchy.

As a remark, we would like to point out the connection and difference on the "hierarchy" in the proposed HDP method and traditional HC [4]. Both the HDP and HC algorithms can provide HC results. The hierarchy in the HDP method is manifested by the Chinese restaurant process which will be introduced later, where the data sit in the same table can be viewed as the first level and all tables sharing the same dish can be viewed as the second level. While the hierarchy in the HC is obtained by merging existing clusters based on their distances. However, its specific merging strategy is heuristic and is irreversible for those merged clusters. Hierarchy formed in this fashion often may not reflect the true structure in the data since various hierarchical structures can be formed by choosing different distance metrics. However, the HDP algorithm captures the hierarchical structure at the model level. The merging is carried out automatically during the inference. Therefore, it naturally takes the hierarchy into consideration.

In summary, we have the following HDP model for the data:

$$\begin{aligned}
\mu_0 &\sim D_1(\alpha_1, \mu_1) \\
\{z_{ji}\}|\mu_o, \alpha_0 &\sim D(\alpha_0, \mu_0) \\
\alpha_0, \alpha_1 &\sim \Gamma(a, b) \\
\theta_k &\sim \mu_1 \\
g_{ji}|z_{ji}, \theta_k &\sim F(\theta_{z_{ji}}),
\end{aligned} \tag{13}$$

where a and b are some fixed constants. We assume that F and μ_1 are conjugate priors. In this article, F is assumed to be the Gaussian distribution and μ_1 is the inverse Gamma distribution.

3 Inference algorithm

It is intractable to get the closed-form solution to the inference problem (5). In this section, we develop a Gibbs sampling algorithm for estimating the posterior distribution in (5). At each iteration l, we draw a sample $z_{ji}^{(l)}$ sequentially from the distribution:

$$P\left(z_{ji}^{(l)}|z_{11}^{(l)}, z_{12}^{(l)}, \ldots, z_{j(i-1)}^{(l)}, z_{j(i+1)}^{(l-1)}, \ldots, z_{MN}^{(l-1)}, \mathbf{g}\right). \tag{14}$$

Under regularity conditions, the distribution of $\{z_{ji}^{(l)}\}_{j,i}$ will converge to the true posterior distribution in (5) [24]. The proposed Gibbs sampling algorithm is similar to the HDP inference algorithm proposed in [21], since both the Gibbs algorithms use the Chinese restaurant metaphor which we will elaborate later. However, because of the differences in modeling, we still need to provide details for the inference algorithm based on our model.

3.1 Chinese restaurant metaphor

The Chinese restaurant model [23] is a visualized characterization for interpreting the Dirichlet process. Because there is no explicit formula to describe the Dirichlet process, we will employ the Chinese restaurant model for HDP inference instead of directly computing the posterior distribution in (5). We refer to [23,25] for the proof and other details of the equivalence between the Chinese restaurant metaphor and the Dirichlet processes.

In the Chinese restaurant metaphor for the HDP model (13), we view $\{z_{ji}\}$ as customers entering a restaurant sequentially. The restaurant has infinite number of rows and columns of tables which are labeled by t_{ji}. Each z_{ji} will associate to one and only one table in the jth row. We use $\phi(z_{ji})$ to denote the column index of the table in the jth row taken by z_{ji}, i.e., z_{ji} will sit at table $t_{j\phi(z_{ji})}$. If it is clear from the context, we will use ϕ_{ji} in short for $\phi(z_{ji})$. The index of the random variable θ_k in (13) is characterized by a menu containing various dishes. Each table picks one and only one dish from the menus $\{m_k\}_{k=1,2,\ldots}$, which are drawn independently from the base measure μ_1. g_{ji} is drawn independently according to the dish it chooses through the distribution $F(\cdot)$ as in (13). We denote $\lambda(t_{ji})$ as the index of the dish taken by table t_{ji}, i.e., table t_{ji} chooses dish $m_{\lambda(t_{ji})}$. As before, we may write λ_{ji} in short of $\lambda(t_{ji})$. In summary, customer z_{ji} will sit at table $t_{j\phi_{ji}}$ and enjoy dish $m_{\lambda_{j\phi_{ji}}}$. The HDP is reflected in this metaphor such that the customers choose the tables as well as the dishes in a Dirichlet process fashion. The customers sitting at the same table are classified into one cluster. Moreover, the customers sitting at different tables but ordering the same dish will also be clustered into the same group. Hence, the clustering effect is performed at the cluster level, i.e., we allow "clustering among clusters". In Figure 2, we show an illustration of the Chinese restaurant metaphor. The different patterns of shades represent different clusters. We also introduce two useful counter variables: c_{ji} denotes the number of customers sitting at table t_{ji}; d_{jk} counts the number of tables in row j serving dish m_k.

Using the Chinese restaurant metaphor, instead of inferring z_{ji}, we can directly infer ϕ_{ji} and λ_{ji}. The membership variable z_{ji} is completely determined by $\lambda(t_{j\phi(z_{ji})})$. That is, $z_{ji} = z_{j'i'}$ if and only if $\lambda(t_{j\phi(z_{ji})}) = \lambda(t_{j\phi(z_{j'i'})})$. As we pointed out before, the specific values of the membership variable z_{ji} are not relevant to the clustering as long as z_{ji} is supported on a countable set. Hence, we could simply let

$$z_{ji} = \lambda\left(t_{j\phi(z_{ji})}\right). \tag{15}$$

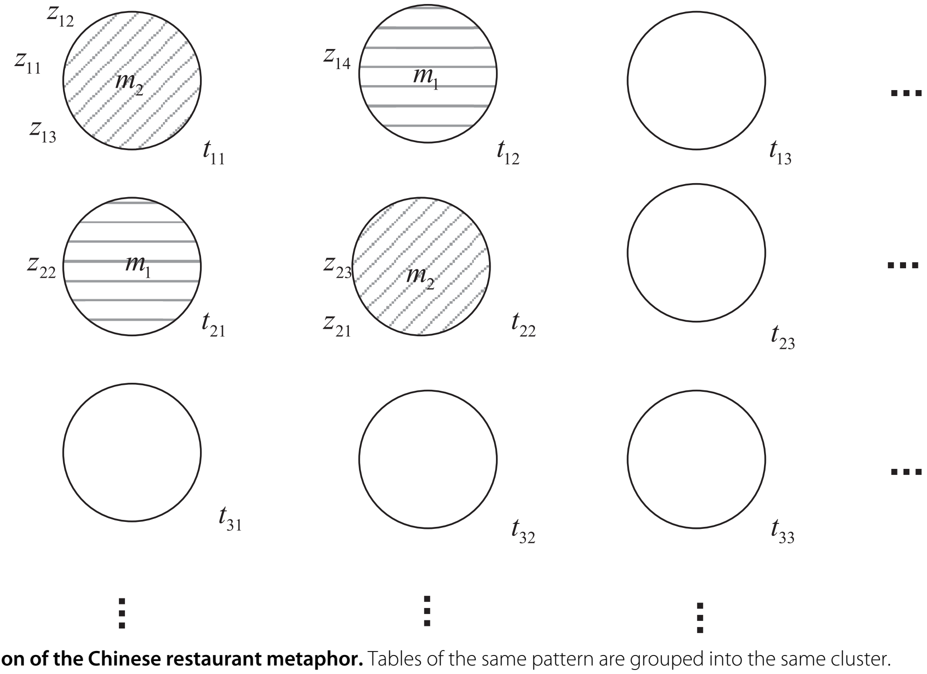

Figure 2 Illustration of the Chinese restaurant metaphor. Tables of the same pattern are grouped into the same cluster.

According to [25], we have the following conditional probabilities for the HDP model

$$\phi_{ji}|\phi_{j1},\ldots,\phi_{ji-1},\alpha_0,\mu_0 \sim \sum_{m=1}^{\sum_k d_{jk}} \frac{c_{jm}}{i-1+\alpha_0}\delta_{t_{j\phi_{ji}}} + \frac{\alpha_0}{i-1+\alpha_0}\mu_0, \tag{16}$$

where $\sum_k d_{jk}$ calculates the number of tables taken in the rth row and $\delta_{(.)}$ is the Kronecker delta function. The interpretation of (16) is that customer z_{ji} chooses a table already taken with equal probability. In addition, z_{ji} may choose a new table with probability proportional to α_0.

By the hierarchical assumption, the distribution of the dish chosen at an occupied table is another Dirichlet process. We have the following conditional distribution of the dishes

$$\lambda_{j\phi_{ji}}|\lambda_{1\phi_{11}},\ldots,\lambda_{j\phi_{j(i-1)}},\alpha_1,\mu_1 \sim \sum_{k=1}^{K_{ji}} \frac{\sum_j d_{jk}}{\sum_{jk} d_{jk}+\alpha_1}\delta_{m_k} + \frac{\alpha_1}{\sum_{jk} d_{jk}+\alpha_1}\mu_1, \tag{17}$$

where $\sum_j d_{jk}$ counts the number of tables serving dish m_k; $\sum_{jk} d_{jk}$ counts the number of tables serving dishes; K_{ji} denotes the net number of dishes served till λ_{ji}'s coming by counting only once each dish that has been served multiple times.

3.2 A Gibbs sampler for HDP inference

Instead of sampling the posterior probability in (5), we will sample $\boldsymbol{\phi} = \{\phi_{11},\phi_{12},\ldots\}$ and $\boldsymbol{\lambda} = \{\lambda_{11},\lambda_{12},\ldots\}$ from the following posterior distribution

$$P(\boldsymbol{\phi},\boldsymbol{\lambda}|\mathbf{g}). \tag{18}$$

We can calculate the related conditional probabilities as follows.

If a is a value that has been taken before, the conditional probability of $\phi_{ji} = a$ is given by

$$P\left(\phi_{ji} = a|\boldsymbol{\phi}_{ji}^c,\boldsymbol{\lambda},\boldsymbol{\theta},\alpha_1,\alpha_0,\mu_1,\mathbf{g}\right) \propto c_{ja}f_{\lambda_{ja}}(g_{ji}|\mathbf{g}_{ji}^c), \tag{19}$$

where $\boldsymbol{\theta} = \{\theta_{ji}\}_{j,i}$ and $\boldsymbol{\lambda} = \{\lambda_{ji}\}_{j,i}$. The superscript c denotes the complement of the variables in its category, i.e., $\mathbf{g}_{ji}^c = \{g_{j'i'}\}_{(j',i')\neq(j,i)}$ and $\boldsymbol{\phi}_{ji}^c = \{\phi_{j'i'}\}_{(j',i')\neq(j,i)}$. $f_{\lambda_{ja}}\left(g_{ji}|\mathbf{g}_{ji}^c\right)$ denotes the conditional density of g_{ji} given all other data generated according to menu $m_{\lambda_{ja}}$, which can be calculated as

$$f_{\lambda_{ja}}\left(g_{ji}|\mathbf{g}_{ji}^c\right) = \frac{\int \prod_{\lambda_{j'\phi_{j'i'}}=\lambda_{ja}} F(g_{j'i'}|\theta)\mu_1(\theta)d\theta}{\int \prod_{j'i'\neq ji,\lambda_{j'\phi_{j'i'}}=\lambda_{ja}} F(g_{j'i'}|\theta)\mu_1(\theta)d\theta}. \tag{20}$$

The numerator of (20) is the joint density of the data which are generated by the same dish. By the assumption that $g_{j'i'}$ are conditionally independent given the chosen dish, we have the conditional density of the data in the product form. The denominator is the joint density excluding the specific g_{ji} term. The integrals in (20) can either be calculated using the numerical method or using the Monte Carlo integration. For example, in order to calculate the following integral $\int_a^b f(x)p(x)dx$, where $p(x)$ is a density function, we can draw samples $x_1,x_2,\ldots,x_n$ from $p(x)$ and approximate the integral by $\int_a^b f(x)p(x)dx = E_{p(x)}[f(x)] \approx \frac{1}{n}\sum_{i=1}^n f(x_i)$. To calculate (20), we view $\mu_1(\cdot)$ as $p(\cdot)$ and $F(g_{j'i'}|\cdot)$ as $f(\cdot)$.

On the other hand, if a is a new value then we have

$$P\left(\phi_{ji} = a|\boldsymbol{\phi}_{ji}^c,\boldsymbol{\lambda},\boldsymbol{\theta},\alpha_1,\alpha_0,\mathbf{g}\right) \propto \alpha_0 \left[\sum_{k=1}^{K_{ja}} \frac{\sum_j d_{jk}}{\sum_{jk} d_{jk}+\alpha_1} f_k\left(g_{ji}|\mathbf{g}_{ji}^c\right) + \frac{\alpha_1}{\sum_{jk} d_{jk}+\alpha_1}\int F(g_{ji}|\theta)\mu_1(\theta)d\theta\right]. \tag{21}$$

We also have the following conditional probabilities for λ_{ji}. If a is used before, we have

$$P\left(\lambda_{j\phi_{ji}} = a|\boldsymbol{\phi},\boldsymbol{\lambda}_{j\phi_{ji}}^c,\boldsymbol{\theta},\alpha_1,\alpha_0,\mathbf{g}\right) \propto \left(\sum_j d_{ja}\right) f_a\left(g_{ji}|\mathbf{g}_{ji}^c\right); \tag{22}$$

otherwise we have

$$P\left(\lambda_{j\phi_{ji}} = a|\boldsymbol{\phi},\boldsymbol{\lambda}_{j\phi_{ji}}^c,\boldsymbol{\theta},\alpha_1,\alpha_0,\mathbf{g}\right) \propto \alpha_1 \int F(g_{ji}|\theta)\mu_1(\theta)d\theta. \tag{23}$$

The derivations of (19), (21), (22), and (23) are given in Appendix.

Before we present the Gibbs sampling algorithm, we recall the Metropolis–Hastings (M–H) algorithm [26] for drawing samples from a target distribution whose density function $f(x)$ is only known up to a scaling factor, i.e., $f(x) \propto p(x)$. To draw samples from $f(x)$, we make use of some fixed conditional distribution $q(x_2|x_1)$ that satisfies $q(x_2|x_1) = q(x_1|x_2)$, $\forall x_1,x_2$. The M–H algorithm proceeds as follows.

- Start with an arbitrary value x_0 with $p(x_0) > 0$.
- For $l = 1,2,\ldots$
  - Given the previous sample x_{l-1}, draw a candidate sample $x^\star$ from $q(x^\star|x_{l-1})$.
  - Calculate $\beta = \frac{p(x^\star)}{p(x_{l-1})}$. If $\beta \geq 1$ then accept the candidate and let $x_l = x^\star$. Otherwise accept it

with probability β, or reject it and accept the previous sample with probability $1 - \beta$.

After a "burn-in" period, say l_0, the samples $\{x_l\}_{l>l_0}$ follow the distribution $f(x)$.

We now summarize the Gibbs sampling algorithm for the HDP inference as follows.

- Initialization: randomly assign the indices $\boldsymbol{\phi}^{(0)} = \left\{\phi_{11}^{(0)}, \phi_{12}^{(0)}, \ldots\right\}$ and $\boldsymbol{\lambda}^{(0)} = \left\{\lambda_{11}^{(0)}, \lambda_{12}^{(0)}, \ldots\right\}$. Note that once we have all the indices, the counters $\{c_{ji}\}$ and $\{d_{jk}\}$ are also determined.
- For $l = 1, 2, \ldots, l_0 + L$,
  - Draw samples of $\left\{\phi_{ji}^{(l)}\right\}$ from their posteriors
$$P\left(\phi_{ji}^{(l)} = a|\boldsymbol{\phi}_{ji}^{(l-1)c}, \boldsymbol{\lambda}^{(l-1)}, \alpha_1^{(l-1)}, \alpha_0^{(l-1)}, \mathbf{g}\right) \tag{24}$$
given by (19) and (21) using the M–H algorithm. We view the probability in (24) as the target density and choose $q(\cdot|\cdot)$ to be a distribution supported on $\mathbb{N}$. For example, we can use $q(i|j) = \frac{j}{(j+1)^i}$, $i, j \in \mathbb{N}$.
  - Draw samples of $\left\{\lambda_{j\phi_{ji}^{(l)}}^{(l)}\right\}$ from their posteriors
$$P\left(\lambda_{j\phi_{ji}^{(l)}}^{(l)} = a|\boldsymbol{\phi}^{(l)}, \boldsymbol{\lambda}_{j\phi_{ji}^{(l)}}^{(l-1)c}, \alpha_1^{(l-1)}, \alpha_0^{(l-1)}, \mathbf{g}\right) \tag{25}$$
given by (22) and (23) using M–H algorithm. We view the probability in (25) as the target density and use $q(\cdot|\cdot)$ as specified in the previous step.
  - Since $P(\alpha_0|\boldsymbol{\phi}, \boldsymbol{\lambda}, \alpha_1, \mathbf{g}) = P(\alpha_0)$ and $P(\alpha_1|\boldsymbol{\phi}, \boldsymbol{\lambda}, \alpha_0, \mathbf{g}) = P(\alpha_1)$, simply draw samples of $\alpha_0^{(l)}$ and $\alpha_1^{(l)}$ from their prior Gamma distributions.
- Using the samples after the "burn-in" period $\left\{\boldsymbol{\phi}^{(l)}, \boldsymbol{\lambda}^{(l)}\right\}_{l=l_0+1}^{l_0+L}$ to calculate $\hat{P}(\boldsymbol{\phi}, \boldsymbol{\lambda}|\mathbf{g})$, which is given by
$$\hat{P}\left(\phi_{ji} = a, \lambda_{j\phi_{ji}} = b\right) = \frac{\sum_{l=l_0+1}^{l_0+L} \mathbf{1}\left\{\phi_{ji}^{(l)} = a, \lambda_{j\phi_{ji}^{(l)}}^{(l)} = b\right\}}{L}, \tag{26}$$
where $\mathbf{1}(\cdot)$ is the indicator function. Determine the membership distribution $P(\mathbf{z}|\mathbf{g})$ from the inferred joint distribution $\hat{P}(\boldsymbol{\phi}, \boldsymbol{\lambda}|\mathbf{g})$ by $P(z_{ji} = a|\mathbf{g}) = \sum_b \hat{P}(\lambda_{jb} = a|\mathbf{g}, \phi_{ji} = b)\hat{P}(\phi_{ji} = b|\mathbf{g})$.
- Calculate the estimation of clustering index $\hat{z}_{\cdot i}$ for the ith gene by $\hat{z}_{\cdot i} = \arg_a \max \sum_j P(z_{ji} = a|\mathbf{g})$.

3.3 A numerical example

In this section, we provide a simple numerical example to illustrate the proposed Gibbs sampler. Let us consider the case $N = M = 2$, i.e., there are 2 genes and 2 experiments. Assume that the expressions are as $g_{11} = 0, g_{12} = 1, g_{21} = -1$, and $g_{22} = 2$. We assume $\mu_1(\theta) \sim \mathcal{N}(0,1)$ and $F(g_{ji}|\theta) \sim \mathcal{N}(\theta, 1)$. For initialization, we set $\phi_{11}^{(0)} = 1, \phi_{12}^{(0)} = 2, \phi_{21}^{(0)} = 3, \phi_{22}^{(0)} = 4$; $\lambda_{1\phi_{11}^{(0)}}^{(0)} = 1, \lambda_{1\phi_{12}^{(0)}}^{(0)} = 1, \lambda_{2\phi_{21}^{(0)}}^{(0)} = 2, \lambda_{2\phi_{22}^{(0)}}^{(0)} = 2$, and $\alpha_0^{(0)} = \alpha_1^{(0)} = 1$.

We first show how to draw sample from $P\left(\phi_{11}^{(1)}|\boldsymbol{\phi}_{11}^{(0)c}, \boldsymbol{\lambda}^{(0)}, \alpha_1^{(0)}, \alpha_0^{(0)}, \mathbf{g}\right)$ by the M–H algorithm. Given the initial value, assume that $q(\cdot|\cdot)$ returns $\phi_{11} = 3$ as a candidate sample. By (19), we have $P\left(\phi_{11}^{(1)} = 1|\boldsymbol{\phi}_{11}^{(0)c}, \boldsymbol{\lambda}^{(0)}, \alpha_1^{(0)}, \alpha_0^{(0)}, \mathbf{g}\right) \propto c_{11} f_{\lambda_{11}}\left(g_{11}|\mathbf{g}_{11}^c\right)$, where $c_{11} = 1$ and $\lambda_{11} = 1$. We also have

$$\begin{aligned} f_1\left(g_{11}|\mathbf{g}_{11}^c\right) &= \frac{\int \prod_{\lambda_{j'\phi_{j'i'}}=1} F(g_{j'i'}|\theta)\mu_1(\theta)d\theta}{\int \prod_{(j',i')\neq(1,1),\lambda_{j'\phi_{j'i'}}=1} F(g_{j'i'}|\theta)\mu_1(\theta)d\theta} \\ &= \frac{\int F(g_{11}|\theta)F(g_{12}|\theta)\mu_1(\theta)d\theta}{\int F(g_{12}|\theta)\mu_1(\theta)d\theta} \approx 0.22971. \end{aligned} \tag{27}$$

Note that the above integral can be calculated either numerically or by using the Monte Carlo integration method.

By (21) and using the specific values of the variables, we obtain

$$\begin{aligned} &P\left(\phi_{11}^{(1)} = 3|\boldsymbol{\phi}_{11}^{(0)c}, \boldsymbol{\lambda}^{(0)}, \alpha_1^{(0)}, \alpha_0^{(0)}, \mathbf{g}\right) \\ &\propto \alpha_0 \left[\sum_{k=1}^{K_{11}} \frac{\sum_j d_{jk}}{\sum_{jk} d_{jk} + \alpha_1} f_k\left(g_{11}|\mathbf{g}_{11}^c\right)\right. \\ &\qquad \left. + \frac{\alpha_1}{\sum_{jk} d_{jk} + \alpha_1} \int F(g_{11}|\theta)\mu_1(\theta)d\theta\right] \end{aligned} \tag{28}$$

with $K_{11} = 1$, $\sum_j d_{j1} = 2$, $\sum_{jk} d_{jk} = 4$, $\alpha_0 = \alpha_1 = 1$. Plugging in these values, we have

$$\begin{aligned} &P\left(\phi_{11}^{(1)} = 3|\boldsymbol{\phi}_{11}^{(0)c}, \boldsymbol{\lambda}^{(0)}, \alpha_1^{(0)}, \alpha_0^{(0)}, \mathbf{g}\right) \\ &\propto \frac{2}{5} f_1\left(g_{11}|\mathbf{g}_{11}^c\right) + \frac{1}{5}\int F(g_{11}|\theta)\mu_1(\theta)d\theta \approx 0.1483. \end{aligned} \tag{29}$$

Since $\beta = \frac{0.1483}{0.22971} \approx 0.6456 < 1$, we should accept this candidate sample $\phi_{11} = 3$ with a probability of 0.6456. After the burn-in period, say the sample returned by the M–H algorithm is $\phi_{11} = 4$, then we update $\phi_{11}^{(1)} = 4$ and move on to draw samples of the remaining variables ϕ_{12}, ϕ_{21}, and ϕ_{22}.

Assuming that we obtain samples of $\boldsymbol{\phi}^{(1)}$ as $\phi_{11}^{(1)} = 4, \phi_{12}^{(1)} = 1, \phi_{21}^{(1)} = 1, \phi_{22}^{(1)} = 2$. We next draw the sample $\boldsymbol{\lambda}^{(1)}$. Given the initial value $\lambda_{1\phi_{11}^{(1)}} = 1$ and $q(\cdot|\cdot)$ returns $\lambda_{1\phi_{11}^{(1)}} = 3$ as a candidate sample. By (22), we obtain $P\left(\lambda_{1\phi_{11}^{(1)}}^{(1)} = 1|\boldsymbol{\phi}^{(1)}, \boldsymbol{\lambda}_{1\phi_{11}^{(1)}}^{(0)c}, \alpha_1^{(0)}, \alpha_0^{(0)}, \mathbf{g}\right) \propto \left(\sum_j d_{j1}\right) f_1\left(g_{11}|\mathbf{g}_{11}^c\right)$. Furthermore, we have $\sum_j d_{j1} = 2$ and $f_1\left(g_{11}|\mathbf{g}_{11}^c\right) \approx 0.22971$ as calculated before.

By (23), we obtain $P\left(\lambda_{1\phi_{11}}^{(1)} = 3|\boldsymbol{\phi}^{(1)}, \boldsymbol{\lambda}_{1\phi_{11}}^{(0)c}, \alpha_1^{(0)}, \alpha_0^{(0)}, \mathbf{g}\right) \propto \alpha_1 \int F(g_{11}|\theta)\mu_1(\theta)d\theta$. Moreover, we have $\alpha_1 = 1$ and $\int F(g_{11}|\theta)\mu_1(\theta)d\theta \approx 0.28208$ as calculated before. So we have $\beta = \frac{0.28208}{2*0.22971} \approx 0.614 < 1$. After the burn-in period, assume that the M–H algorithm returns a sample $\lambda_{1\phi_{11}^{(1)}} = 2$, then update $\lambda_{1\phi_{11}^{(1)}}^{(1)} = 2$ and move on to sample the remaining λ variables as well as α_0 and α_1.

After the burn-in period of the whole Gibbs sampler, we can calculate the posterior joint distribution $P(\boldsymbol{\phi}, \boldsymbol{\lambda}|\mathbf{g})$ from the samples and determine the clusters following the last two steps in the proposed Gibbs sampling algorithm.

4 Experimental results

The HDP clustering algorithm proposed in this article can be employed for gene expression analysis or as a segmentation algorithm for gene regulatory network inference. In this section, we first introduce two performance measures for clustering, the Rand Index (RI) [27] and the Silhouette Index (SI) [28]. We compare the HDP algorithm to the support vector machine (SVM) algorithm for network segmentation on synthetic data. We then conduct various experiments on both synthetic and real datasets including the AD400 datasets [29], the yeast galactose datasets [30], yeast sporulation datasets [31], human fibroblasts serum datasets [32], and yeast cell cycle data [33]. We compare the HDP algorithm to the Latent Dirichlet allocation (LDA), MCLUST, SVM, K-means, Bayesian Infinite Mixture Clustering (BIMC) the HC [4,14,34-37] based on the performance measures and the functional relationships.

4.1 Performance measures

In order to evaluate the clustering result, we utilize two measures: RI [27] and SI [28]. The first index is used when a ground truth is known in *priori* and the second index is to measure the performance without any knowledge of the ground truth.

The RI is a measure of agreement between two clustering results. It takes a value between 0 and 1. The higher is the score, the higher agreements it indicates.

Let A denote the datasets with a total number of n elements. Given two clustering results $X = \{X_1, \ldots, X_S\}$ and $Y = \{Y_1, \ldots, Y_T\}$ of A, i.e., $A = \bigcup_{i=1}^{S} X_i = \bigcup_{j=1}^{T} Y_j$ and $X_i \bigcap X_j = \emptyset$, $Y_i \bigcap Y_j = \emptyset$ for $i \neq j$. For any pair of elements (a, b) in A, we say they are in the same set under a clustering result if a and b are in the same cluster. Otherwise we say they are in different sets. Note that there are totally $\binom{n}{2}$ pairs of elements. We define the following four counting numbers: Z_1 denotes the number of pairs that are both in the same set in X and Y; Z_2 denotes the number of pairs that are both in different sets in X and Y; Z_3 denotes the number of pairs that are in the same set in X and in different sets in Y; and Z_4 denotes the number of pairs that are in different sets in X and in the same set in Y. The RI is then given by

$$\text{RI} = \frac{Z_1 + Z_2}{Z_1 + Z_2 + Z_3 + Z_4}. \tag{30}$$

Due to the lack of the ground truth in most real applications, we utilize the SI to evaluate the clustering performance. The SI is a measure by calculating the average width of all data points, which reflects the compactness of the clustering. Let x denote the average distance between a point p in a cluster and all other points within that cluster. Let y be the minimum average distance between p and other clusters. The Silhouette distance for p is defined as

$$s(p) = \frac{y - x}{\max\{x, y\}}. \tag{31}$$

The SI is the average Silhouette distance among all data points. The value of SI lies in $[-1, 1]$ and higher score indicates better performance.

4.2 Network segmentation on synthetic data

In regulatory network inference, due to the large size of the network, it is often useful to perform a network segmentation. The segmented sub-networks usually have much less number of nodes than the original network, leading to faster and more accurate analysis of the original network [38]. Clustering algorithms can be employed for such segmentation purpose. However, traditional clustering algorithms often provide segmentation results either too fine or too coarse, i.e., the resulting sub-networks

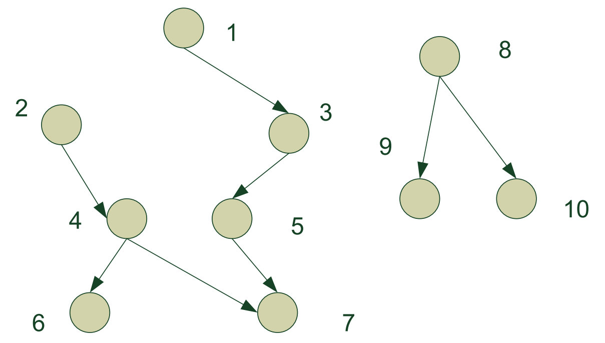

Figure 3 The synthetic network structure.

Table 1 Clustering performance of LDA, SVM, MCLUST, K-means, HC, and HDP on the AD400 data

Algorithm	RI	SI	Number of clusters
LDA	0.931	0.553	10.0
SVM	0.929	0.493	11
MCLUST	0.942	0.583	10
K-means	0.895	0.457	10
HC	0.916	0.348	9
BIMC	0.935	0.571	10.0
HDP	0.947	0.577	10.0

either contain too few genes or two many genes. In addition, the hierarchical structure of the network cannot be discovered by those algorithms. Thanks to its hierarchical model assumption, the HDP algorithm can provide better segmentation results. We demonstrate the segmentation application of HDP on a synthetic network and compare to the SVM algorithm which is widely used for clustering and segmentation.

The network under consideration is shown in Figure 3. We assume that the distributions for all nodes are Gaussian. The directed links indicate that the parent nodes are the priors of the child nodes. Disconnected nodes are mutually independent. We generate the data in the following way. Nodes 1, 2, and 8 are generated independently by Gaussian distributions of unit variance with means 1, 2, and 3, respectively. Nodes 3, 4, 5, 6, 9, and 10 are generated independently by unit variance Gaussian distributions with means determined by their respective parent nodes. Node 7 is generated by a Gaussian distribution with mean determined by node 4 and variance determined by absolute value of node 5. The network contains two isolated segments with one segment containing nodes 1–7 and the other containing nodes 8–10. The HDP algorithm is applied to this network and segments the network into three clusters. Nodes 2, 4, 6 form one cluster; nodes 1, 3, 5, 7 form another cluster; and nodes 8, 9, 10 form the third

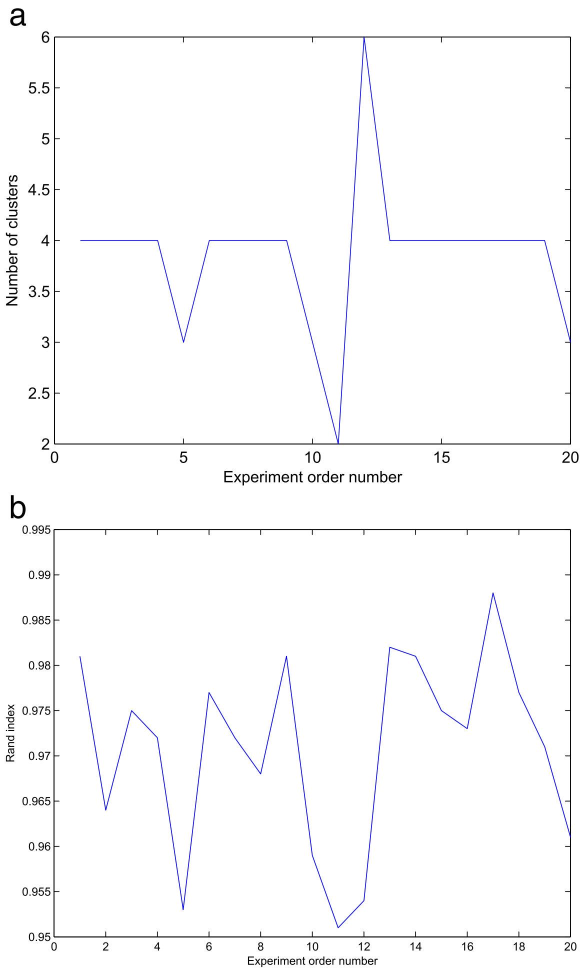

Figure 4 Plots of the HDP results in 20 experiments. (a) Plot of number of clusters in 20 experiments. **(b)** Plot of rand index in 20 experiments.

one. The SVM algorithm on the other hand produces two clusters, one containing nodes 1–7 and the other containing nodes 8–10. As one can see, the network obviously contains two hierarchies in the left segment, i.e., nodes 1–7 of the network. The SVM fails to recognize the hierarchies and provides a result coarser than that given by the HDP algorithm.

4.3 AD400 data

The AD400 is a synthetic dataset proposed in [29], which is used to evaluate the clustering algorithm performance. The dataset is constituted by 400 genes with 10 time points. As the ground truth, the AD400 dataset has 10 clusters with each one containing 40 genes.

For randomized algorithms as LDA, BIMC, HDP, we average the results over 20 runs of the algorithms. We compare the HDP algorithm to other widely used algorithms such as LDA, SVM, MCLUST, K-means, BIMC, and HC. The results are presented in Table 1. As we can see, the HDP algorithm has the similar performance of the MCLUST algorithm. While the HDP generally performs better than other widely used algorithms.

4.4 Yeast galactose data

We conduct experiment on the yeast galactose data, which consists of 205 genes. The true number of clusters based on the functional categories is 4 [39]. We calculate the RI index between different clustering results to the result in [39], which is regarded as the standard benchmark. The LDA model is a generative probabilistic model for document classifications [34], which also uses Dirichlet distribution as a prior. We adapt the LDA model to the yeast galactose data to compare the proposed HDP algorithm. Since the LDA and HDP methods are randomized algorithms, we run the algorithms 20 times and use the average for the final score. In Figure 4, we illustrate the performances of each experiments for the HDP method. The performances of the algorithms under consideration are listed in Table 2.

It is seen that the HDP algorithm performs the best among the three algorithms. Unlike the MCLUST and LDA algorithms which produce more clusters than 4, the average number of clusters given by the HDP algorithm is very closed to the "true" value 4. Compared to the SVM method, the HDP algorithm produces a result that is more similar to the "ground truth", i.e., with the highest RI value.

Table 2 Clustering performance of LDA, MCLUST, SVM, and HDP on the yeast galactose data

Algorithm	Rand index	Number of clusters
LDA	0.942	6.3
SVM	0.954	5
MCLUST	0.903	9
HDP	0.973	3.8

Table 3 Clustering performance of LDA, MCLUST, K-means, HC, BIMC, and HDP on the yeast sporulation data

Algorithm	SI	Number of clusters
LDA	0.586	6.2
MCLUST	0.577	6
K-Means	0.324	8
HC	0.392	7
BIMC	0.592	6.1
HDP	0.673	6.0

4.5 Yeast sporulation data

The yeast sporulation dataset consists of 6,118 genes with 7 times points which were obtained during the sporulation process [31]. We pre-processed the dataset by applying a logarithmic transform and removing the data whose expression levels did not have significant changes. After the pre-process, the data have 513 genes left. In Table 3, we compare the HDP clustering result to LDA, MCLUST, K-Means, BIMC, and HC. For randomized algorithms such as LDA, BIMC, and HDP, we average the scores by running the algorithm 20 times.

From Table 3, we can see that the HDP has the highest SI score. It suggests that the clustering results provided by HDP are more compact and less separated than results from other algorithms. The K-means and HC algorithm suggest higher number of clusters. However, their SI scores indicate that their clusters are not as tight as other algorithms.

4.6 Human fibroblasts serum data

The human fibroblasts serum data consists of 8,613 genes with 12 time points [32]. Again a logarithmic transform has been applied to the data and genes without significant changes have been removed. The remaining dataset has 532 genes.

In Table 4, we show the performance of the HDP algorithm and other various algorithms. It has been shown

Table 4 Clustering performance of LDA, MCLUST, K-means, HC, BIMC, and HDP on the human fibroblasts serum data

Algorithm	SI	Number of clusters
LDA	0.298	9.4
MCLUST	0.382	6
K-Means	0.324	7
HC	0.313	5
BIMC	0.418	7.3
HDP	0.452	6.4

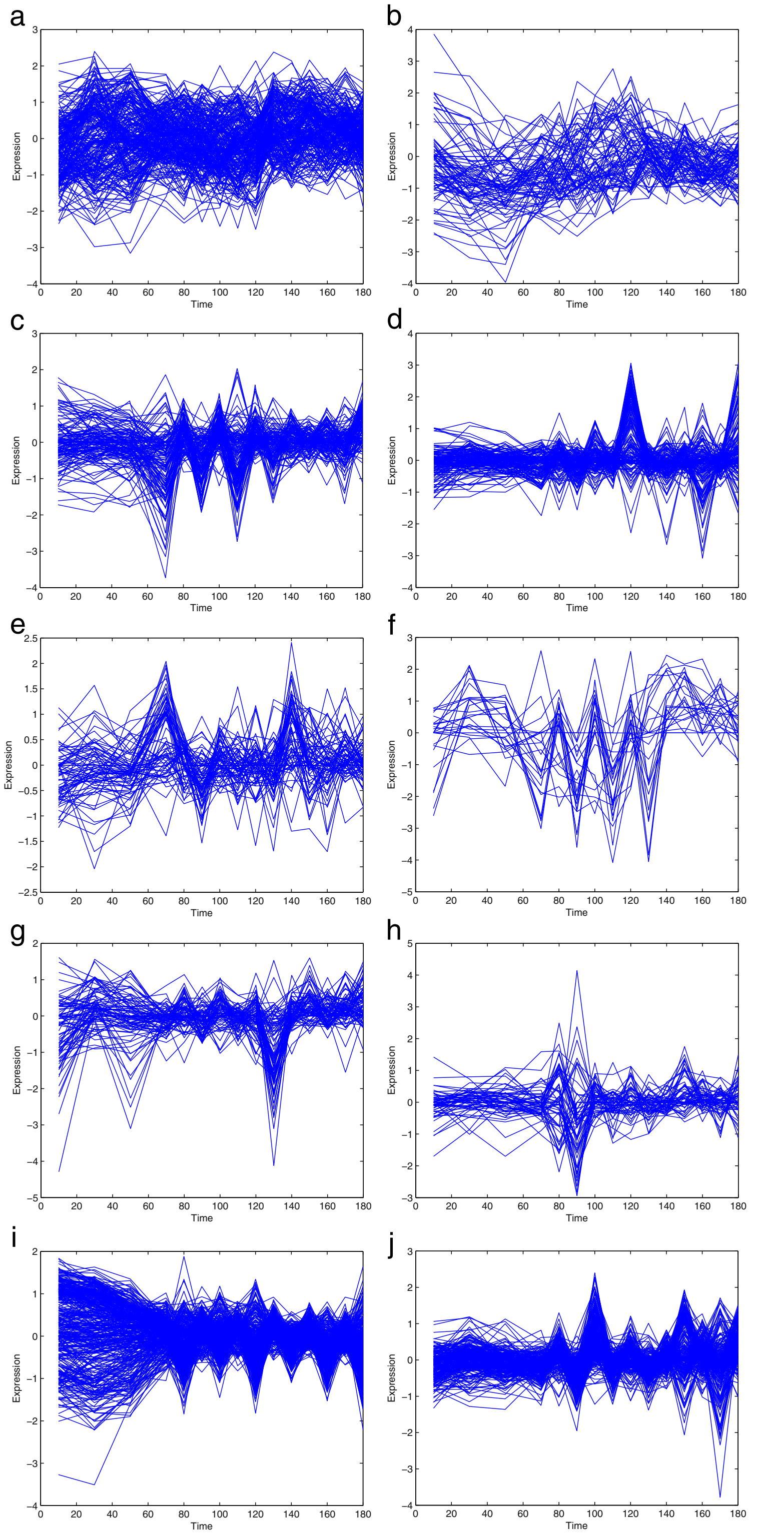

Figure 5 Plots of all HDP clusters for yeast cell cycle data. **(a)** Plot of Cluster 1, containing 261 genes. **(b)** Plot of Cluster 2, containing 86 genes. **(c)** Plot of Cluster 3, containing 135 genes. **(d)** Plot of Cluster 4, containing 144 genes. **(e)** Plot of Cluster 5, containing 76 genes. **(f)** Plot of Cluster 6, containing 25 genes. **(g)** Plot of Cluster 7, containing 88 genes. **(h)** Plot of Cluster 8, containing 60 genes. **(i)** Plot of Cluster 9, containing 381 genes. **(j)** Plot of Cluster 10, containing 259 genes.

Table 5 Numbers of newly discovered genes in various functional categories by the proposed HDP clustering algorithm

Function categories	Number of newly discovered genes
Cell cycle and DNA processing	20
Protein synthesis	25
Protein fate	4
Cell fate	12
Transcription	8
Unclassified protein	57

that the clustering results by the HDP algorithm are the compactest among those algorithms. The LDA algorithm suggests 9.4 clusters with the lowest SI score, which indicates that some of its clusters can be further tightened. HC provides a result consisting of five clusters. However, the SI score of the HC result is not the highest, which suggests its clustering may not be well formed.

4.7 Yeast cell cycle data

We next apply the proposed HDP clustering algorithm on the yeast cell *Saccharomyces cerevisiae* cycle dataset [2,40]. The data are obtained by synchronizing and collecting the mRNAs from cells at 10-min intervals over the course of two cell cycles. It has been used widely for testing the performances of clustering algorithm [2,14,41]. The expression data have been taken logarithmic transform and lie in the interval $[-2, 2]$. We pre-processed the data to remove those which did not change significantly over time. We also removed those data whose means are below a small threshold. After the pre-processing, there are 1,515 genes left. We then apply the HDP algorithm and obtain 10 clusters in total. The plots of the clusters are shown in Figure 5.

We resort to the MIPS database [42] to determine the functional categories for each cluster. The inferred functional category of a cluster is the category shared by the majority of the member elements. After applying the cell-cycle selection criterion in [2], we find that there are 126 genes identified by proposed HDP algorithm but not discovered in [2]. We list in Table 5 the numbers of newly discovered genes in various functional categories. We also observe that parts of the newly discovered unclassified genes belong to clusters with classified categories. Given the hierarchical characteristic of the HDP algorithm, it may suggest multiple descriptions of those genes that might have been overlooked before.

Note that in [14] a Bayesian model with infinite number of clusters is proposed based on the Dirichlet process. The model in [14] is a special case of the HDP model proposed in this article when there is only one hierarchy. In terms of discovering new gene functionalities, we find that the performances of the two algorithms are similar, as the method in [14] discovered 106 new genes compared to the result in [2]. However, by taking the hierarchical structure into account, the total number of clusters found by the HDP algorithm is significantly smaller than that given in [14] which is 43 clusters. The SI score for BIMC and HDP are 0.321 and 0.392, respectively. The HDP clustering consolidates many fragmental clusters, which may provide an easier way to interpret the clustering results.

In Table 6, we list the new genes discovered by the HDP algorithm which are not found in [2].

5 Conclusions

In this article, we have proposed a new clustering approach based on the HDP. The HDP clustering explicitly models the hierarchical structure in the data that is prevalent in biological data such as gene expressions. We have developed a statistical inference algorithm for the proposed HDP model based on the Chinese restaurant metaphor and the Gibbs sampler. We have applied the proposed HDP clustering algorithm to both regulatory network segmentation and gene expression clustering. The HDP algorithm is shown to reveal more structural information of the data compared to popular algorithms such as SVM and MCLUST, by incorporating the hierarchical knowledge into the model.

Table 6 List of newly discovered genes in various functional categories

Function categories	Genes
Cell cycle and DNA processing	YBL051c YBR136w YBL016w YDR200c YBR274w YDR217c YLR314c YJL074c YJL095w YDR052c YDL126c YCL016c YDL188c YAL040c YEL019c YER122c YLR035c YLR055c YML032c YMR078c
Protein synthesis	YDR091c YGL103w YBR118w YBL057c YBR101c YBR181c YDL083c YDL184c YDR012w YDR172w YGL105w YGL129c YJL041w YJL125c YJR113c YLR185w YPL037c YPL048w YLR009w YHL001w YHL015w YHR011w YHR088w YDR450w YEL034w
Protein fate	YAL016w YBL009w YBR044c YDL040c
Cell fate	YAL040c YDL006w YDL134c YIL007c YJL187c YDL029w YDL035c YCR002c YBL105c YCR089w YER114c YEL023c
Transcription	YAL021c YBL022c YCL051w YDR146c YIL084c YJL127c YJL164c YJL006c

Appendix

Derivation of formula (19) and (21)

$$
\begin{aligned}
&P\left(\phi_{ji}=a|\boldsymbol{\phi}^c_{ji},\boldsymbol{\lambda},\boldsymbol{\theta},\alpha_1,\alpha_0,\mu_1,\mathbf{g}\right)\\
&\quad=\frac{P\left(g_{ji},\phi_{ji}=a|\boldsymbol{\phi}^c(z_{ji}),\boldsymbol{\lambda},\boldsymbol{\theta},\alpha_1,\alpha_0,\mu_1,\mathbf{g}^c_{ji}\right)}{P\left(g_{ji}|\boldsymbol{\phi}^c_{ji},\boldsymbol{\lambda},\boldsymbol{\theta},\alpha_1,\alpha_0,\mu_1,\mathbf{g}^c_{ji}\right)} &(32)\\
&\quad\propto P\left(g_{ji},\phi_{ji}=a|\boldsymbol{\phi}^c_{ji},\boldsymbol{\lambda},\boldsymbol{\theta},\alpha_1,\alpha_0,\mu_1,\mathbf{g}^c_{ji}\right) &(33)\\
&\quad\propto P\left(g_{ji}|\boldsymbol{\phi},\boldsymbol{\lambda},\boldsymbol{\theta},\alpha_1,\alpha_0,\mu_1,\mathbf{g}^c_{ji}\right)\\
&\qquad P\left(\phi_{ji}=a|\boldsymbol{\phi}^c_{ji},\boldsymbol{\lambda},\boldsymbol{\theta},\alpha_1,\alpha_0,\mu_1,\mathbf{g}^c_{ji}\right) &(34)
\end{aligned}
$$

By (16), if a has appeared before, we have

$$P\left(\phi_{ji}=a|\boldsymbol{\phi}^c_{ji},\boldsymbol{\lambda},\boldsymbol{\theta},\alpha_1,\alpha_0,\mu_1,\mathbf{g}^c_{ji}\right)\propto c_{ja}. \qquad (35)$$

Otherwise we have

$$P\left(\phi_{ji}=a|\boldsymbol{\phi}^c_{ji},\boldsymbol{\lambda},\boldsymbol{\theta},\alpha_1,\alpha_0,\mu_1,\mathbf{g}^c_{ji}\right)\propto \alpha_0. \qquad (36)$$

If a has appeared before, by the assumption the data are conditionally independent, we also have

$$P\left(g_{ji}|\boldsymbol{\phi},\boldsymbol{\lambda},\boldsymbol{\theta},\alpha_1,\alpha_0,\mu_1,\mathbf{g}^c_{ji}\right)=f_{\lambda_{ja}}\left(g_{ji}|\mathbf{g}^c_{ji}\right), \qquad (37)$$

where $f_{\lambda_{ja}}(g_{ji}|\mathbf{g}^c_{ji})$ can be calculated by the Bayes' formula:

$$f_{\lambda_{ja}}\left(g_{ji}|\mathbf{g}^c_{ji}\right)=\frac{\int\prod_{\lambda_{j'\phi_{j'i'}}=\lambda_{ja}}F(g_{j'i'}|\theta)\mu_1(\theta)d\theta}{\int\prod_{(j',i')\neq(j,i),\lambda_{j'\phi_{j'i'}}=\lambda_{ja}}F(g_{j'i'}|\theta)\mu_1(\theta)d\theta}. \qquad (38)$$

Combining (35) and (37), we have (19).

If a has not appeared before, by (17), we have

$$
\begin{aligned}
&P\left(g_{ji}|\boldsymbol{\phi},\boldsymbol{\lambda},\boldsymbol{\theta},\alpha_1,\alpha_0,\mu_1,\mathbf{g}^c_{ji}\right)\\
&=\sum_{k=1}^{K_{ja}}\frac{\sum_j d_{jk}}{\sum_{jk}d_{jk}+\alpha_1}f_k\left(g_{ji}|\mathbf{g}^c_{ji}\right)+\frac{\alpha_1}{\sum_{jk}d_{jk}+\alpha_1}\int F(g_{ji}|\theta)\mu_1(\theta)d\theta, \qquad (39)
\end{aligned}
$$

Combining (36) and (39), we have (21).

Derivation of (22) nd (23)

$$
\begin{aligned}
&P\left(\lambda_{j\phi_{ji}}=a|\boldsymbol{\phi},\boldsymbol{\lambda}^c_{j\phi_{ji}},\boldsymbol{\theta},\alpha_1,\alpha_0,\mu_1,\mathbf{g}\right)\\
&\quad=\frac{P\left(g_{ji},\lambda_{j\phi_{ji}}=a|\boldsymbol{\phi},\lambda^c_{j\phi_{ji}},\boldsymbol{\theta},\alpha_1,\alpha_0,\mu_1,\mathbf{g}^c_{ji}\right)}{P\left(g_{ji}|\boldsymbol{\phi},\boldsymbol{\lambda},\boldsymbol{\theta},\alpha_1,\alpha_0,\mu_1,\mathbf{g}^c_{ji}\right)} &(40)\\
&\quad\propto P\left(g_{ji},\lambda_{j\phi_{ji}}=a|\phi,\boldsymbol{\lambda}^c_{j\phi_{ji}},\boldsymbol{\theta},\alpha_1,\alpha_0,\mu_1,\mathbf{g}^c_{ji}\right) &(41)\\
&\quad\propto P\left(g_{ji}|\boldsymbol{\phi},\boldsymbol{\lambda},\boldsymbol{\theta},\alpha_1,\alpha_0,\mu_1,\mathbf{g}^c_{ji}\right)\\
&\qquad P\left(\lambda_{j\phi_{ji}}=a|\boldsymbol{\phi},\boldsymbol{\lambda}^c_{j\phi_{ji}},\boldsymbol{\theta},\alpha_1,\alpha_0,\mu_1,\mathbf{g}^c_{ji}\right) &(42)
\end{aligned}
$$

By (17), if a has appeared before, we have

$$P\left(\lambda_{j\phi_{ji}}=a|\boldsymbol{\phi},\boldsymbol{\lambda}^c_{j\phi_{ji}},\boldsymbol{\theta},\alpha_1,\alpha_0,\mu_1,\mathbf{g}^c_{ji}\right)\propto\sum_j d_{ja}. \qquad (43)$$

Otherwise we have

$$P\left(\lambda_{j\phi_{ji}}=a|\boldsymbol{\phi},\boldsymbol{\lambda}^c_{j\phi_{ji}},\boldsymbol{\theta},\alpha_1,\alpha_0,\mu_1,\mathbf{g}^c_{ji}\right)\propto\alpha_1. \qquad (44)$$

If a is used before, we have

$$P\left(g_{ji}|\boldsymbol{\phi},\boldsymbol{\lambda},\boldsymbol{\theta},\alpha_1,\alpha_0,\mu_1,\mathbf{g}^c_{ji}\right)=f_a\left(g_{ji}|\mathbf{g}^c_{ji}\right). \qquad (45)$$

Otherwise, the customer chooses a new table. The data are generated from F based on a sample from μ_1. We have

$$P\left(g_{ji}|\boldsymbol{\phi},\boldsymbol{\lambda},\boldsymbol{\theta},\alpha_1,\alpha_0,\mu_1,\mathbf{g}^c_{ji}\right)=\int F(g_{ji}|\theta)\mu_1(\theta)d\theta. \qquad (46)$$

Combining (43), (44), (45), and (46), we have (22) and (23).

Competing interests

The authors declare that they have no competing interests.

Author details

[1] Department of Electrical & Computer Engineering, Duke University, Durham, NC 27708, USA. [2] Department of Electrical Engineering, Columbia University, New York, NY 10027, USA.

Received: 17 October 2012 Accepted: 11 March 2013
Published: 12 April 2013

References

1. M Schena, D Shalon, R Davis, P Brown, Quantitative monitoring of gene expression patterns with a complementary DNA microarray. Science. **270**(5235), 467–470 (1995)

2. R Cho, M Campbell, E Winzeler, L Steinmetz, A Conway, L Wodicka, T Wolfsberg, A Gabrielian, D Landsman, D Lockhart, A genome-wide transcriptional analysis of the mitotic cell cycle. Mol. Cell. **2**, 65–73 (1998)
3. J Hughes, P Estep, S Tavazoie, G Church, Computational identification of cis-regulatory elements associated with groups of functionally related genes in Saccharomyces cerevisiae. J. Mol. Biol. **296**(5), 1205–1214 (2000)
4. M Eisen, P Spellman, P Brown, D Botstein, Cluster analysis display of genome-wide expression patterns. Proc. Natl. Acad. Sci. **95**(25), 14863–14868 (1998)
5. J MacQueen, in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1. Some methods for classification and analysis of multivariate observations (University of California Press, California, 1967), pp. 281–297
6. T Kohonen, *Self-Organization and Associative Memory*. (Springer, New York, 1988)
7. D Jiang, C Tang, A Zhang, Cluster analysis for gene expression data: a survey. IEEE Trans Knowledge Data Eng. **16**(11), 1370–1386 (2004)
8. A Dempster, N Laird, D Rubin, Maximum likelihood from incomplete data via the EM algorithm. J. R. Stat. Soc. Ser. B (Methodological). **39**, 1–38 (1977)
9. G McLachlan, D Peel, *Finite Mixture Models*. (Wiley-Interscience, New York, 2000)
10. C Fraley, A Raftery, Model-based clustering, discriminant analysis, and density estimation. Am. J. Stat. Assoc. **97**(458), 611–631 (2002)
11. K Yeung, C Fraley, A Murua, A Raftery, W Ruzzo, Model-based clustering and data transformations for gene expression data. Bioinformatics. **17**(10), 977–987 (2001)
12. G Schwarz, Estimating the dimension of a model. Ann. Stat. **6**(2), 461–464 (1978)
13. H Akaike, A new look at the statistical model identification. IEEE Trans. Autom. Control. **19**(6), 716–723 (1974)
14. M Medvedovic, S Sivaganesan, Bayesian infinite mixture model based clustering of gene expression profiles. Bioinformatics. **18**(9), 1194–1206 (2002)
15. T Ferguson, A Bayesian analysis of some nonparametric problems. Ann. Stat. **1**(2), 209–230 (1973)
16. R Neal, Markov chain sampling methods for Dirichlet process mixture models. J. Comput. Graph. Stat. **9**(2), 249–265 (2000)
17. J Pitman. Some developments of the Blackwell-MacQueen urn scheme. *Lecture Notes-Monograph Series* (1996), pp. 245–267
18. L Kaufman, P Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis, vol. 39*, (1990)
19. D Jiang, J Pei, A Zhang, in *Proceedings of Third IEEE Symposium on Bioinformatics and Bioengineering*. DHC: a density-based hierarchical clustering method for time series gene expression data (IEEE, Bethesda, 2003), pp. 393–400
20. J Piatigorsky, *Gene Sharing and Evolution: The Diversity of Protein Functions*. (Harvard University Press, Cambridge, 2007)
21. Y Teh, M Jordan, M Beal, D Blei, Hierarchical Dirichlet processes. J. Am. Stat. Assoc. **101**(476), 1566–1581 (2006)
22. J Sethuraman, A constructive definition of Dirichlet priors. Stat. Sinica. **4**, 639–650 (1991)
23. D Aldous. Exchangeability and related topics. *École d'Été de Probabilités de Saint-Flour XIII* (1985), pp. 1–198
24. G Casella, E George, Explaining the Gibbs sampler. Am. Stat. **46**(3), 167–174 (1992)
25. D Blackwell, J MacQueen, Ferguson distributions via Pólya urn schemes. Ann. Stat. **1**(2), 353–355 (1973)
26. S Brooks, Markov chain Monte Carlo method and its application. J. R. Stat. Soc. Ser. D (The Statistician). **47**, 69–100 (1998)
27. L Hubert, P Arabie, Comparing partitions. J. Classif. **2**, 193–218 (1985)
28. PJ Rousseeuw, Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J. Comput. Appl. Math. **20**, 53–65 (1987)
29. KY Yeung, WL Ruzzo, Principal component analysis for clustering gene expression data. Bioinformatics. **17**(9), 763–774 (2001)
30. K Yeung, M Medvedovic, R Bumgarner, Clustering gene-expression data with repeated measurements. Genome Biol. **4**(5), R34 (2003)
31. S Chu, J DeRisi, M Eisen, J Mulholland, D Botstein, PO Brown, I Herskowitz, The transcriptional program of sporulation in budding yeast. Science. **282**(5389), 699–705 (1998)
32. VR Iyer, MB Eisen, DT Ross, G Schuler, T Moore, JC Lee, JM Trent, LM Staudt, J Hudson, MS Boguski, et al., The transcriptional program in the response of human fibroblasts to serum,. Science. **283**(5398), 83–87 (1999)
33. P Spellman, G Sherlock, M Zhang, V Iyer, K Anders, M Eisen, P Brown, D Botstein, B Futcher, Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. Mol. Biol. Cell. **9**(12), 3273 (1998)
34. D Blei, A Ng, M Jordan, Latent Dirichlet allocation. J. Mach. Learn. Res. **3**, 993–1022 (2003)
35. C Fraley, A Raftery, MCLUST: software for model-based cluster analysis. J. Classif. **16**(2), 297–306 (1999)
36. T Furey, N Cristianini, N Duffy, D Bednarski, M Schummer, D Haussler, Support vector machine classification and validation of cancer tissue samples using microarray expression data. Bioinformatics. **16**(10), 906–914 (2000)
37. S Tavazoie, JD Hughes, MJ Campbell, RJ Cho, GM Church, et al., Systematic determination of genetic network architecture. Nat. Genetics. **22**, 281–285 (1999)
38. F Chung, L Lu. *Complex Graphs and Networks*. CBMS Lecture Series, vol. 107 (American Mathematical Society, Providence, 2006)
39. M Ashburner, C Ball, J Blake, D Botstein, H Butler, J Cherry, A Davis, K Dolinski, S Dwight, J Eppig, et al., Gene ontology: tool for the unification of biology. Nat. Genet. **25**, 25–29 (2000)
40. Stanford University. Yeast cell cycle datasets. http://genome-www.stanford.edu/cellcycle/data/rawdata
41. A Lukashin, R Fuchs, Analysis of temporal gene expression profiles: clustering by simulated annealing and determining the optimal number of clusters. Bioinformatics. **17**(5), 405–414 (2001)
42. H Mewes, D Frishman, U Guldener, G Mannhaupt, K Mayer, M Mokrejs, B Morgenstern, M Munsterkotter, S Rudd, B Weil, MIPS: a database for genomes and protein sequences. Nucleic Acids Res. **30**, 31–34 (2002)

doi:10.1186/1687-4153-2013-5
Cite this article as: Wang and Wang: **Hierarchical Dirichlet process model for gene expression clustering.** *EURASIP Journal on Bioinformatics and Systems Biology* 2013 **2013**:5.