Biomedical Engineering and Computational Biology

REVIEW

Computational Methodologies for Analyzing, Modeling and Controlling Gene Regulatory Networks

Zahra Zamani, Amirhossein Hajihosseini and Ali Masoudi-Nejad

Laboratory of Systems Biology and Bioinformatics (LBB), Institute of Biochemistry and Biophysics, and COE in Biomathematics, University of Tehran, Tehran, Iran. Corresponding author email: amasoudin@ibb.ut.ac.ir

Abstract: Molecular biology focuses on genes and their interactions at the transcription, regulation and protein level. Finding genes that cause certain behaviors can make therapeutic interventions more effective. Although biological tools can extract the genes and perform some analyses, without the help of computational methods, deep insight of the genetic function and its effects will not occur. On the other hand, complex systems can be modeled by networks, introducing the main data as nodes and the links in-between as the transactions occurring within the network. Gene regulatory networks are examples that are modeled and analyzed in order to gain insight of their exact functions. Since a cell's specific functionality is greatly determined by the genes it expresses, translation or the act of converting mRNA to proteins is highly regulated by the control network that directs cellular activities. This paper briefly reviews the most important computational methods for analyzing, modeling and controlling the gene regulatory networks.

Keywords: gene regulatory networks, network modeling, computational modeling, network biology, gene expression, classification, clustering, molecular biology

Biomedical Engineering and Computational Biology 2010:2 47–62

doi: 10.4137/BECB.S5594

This article is available from http://www.la-press.com.

Introduction

Since the advent of the human genome project, molecular biology—starting in post World War II—has grown rapidly. Today large amounts of data and uncertainty are major problems in complex systems such as dynamic networks which make them hard to be analyzed. These systems can be modeled by networks like metabolic networks, signaling networks and gene regulatory networks, all having fundamental elements that build the network (such as proteins, organs or nucleic acids) and certain properties that emerge as the interaction of the network structure. Computational and mathematical techniques can create a model and by simulating it, capture the dynamic behaviors of these networks.

The main focus of this work will be on gene regulatory networks. The input data for these networks are gene expression data obtained from various technologies for data mining the gene expression datasets. This data has to be first preprocessed. Normalization can then be preformed to remove the potential noise. Although this removes any irrelevant datasets, sometimes some expression levels may be corrupted or missing so that imputing the missing data is done at this stage. Next, gene regulatory networks will be identified from the data. The model created now has to be examined to find perturbations or external controls leading to desirable states of the model. At the end, a new formal framework is introduced which will have the benefits of the methods overviewed in this survey.

Preprocessing the Input Data

The input data for the regulatory network is extracted from various technologies. The microarray experiment has been a better choice among the other methods so far.[1] The gene expression datasets carry variations and noise during the measuring steps which may lead to uncertain or even incorrect analysis of the data.

In the preprocessing phase, the image is scanned and image-processing methods are used to reveal the data within an image. This process is as follows: the spots corresponding to genes are identified, their boundaries are determined and the color intensity of each spot is measured and compared to the background. Background subtraction is usually performed using mathematical models like the Gaussian process. To reduce the noise in the final intensity, target detection algorithms are used to eliminate all weak intensities that fall below a certain value and then detect whether it falls within a grid or not using detection methods like Mann-Whitney or fixed threshold. Background correction methods for more sophisticated models use other algorithms like non parametric, Mixture of Gaussian and Code-book based techniques.[1] Backgrounds having fast variations cannot be accurately modeled with just a few Gaussians, so non-parametric methods are used for estimating background probabilities at each pixel from many recent samples over time using Kernel density estimation and the codebook (CB) background subtraction algorithm was intended to sample values over long times, without making parametric assumptions.

The gene expression can now be represented by an expression matrix

$$M = \{g_{ij} | 1 \leq i \leq n, 1 \leq j \leq m\},$$

where g_{ij} is the expression level of the ith gene with the jth condition. Scaling this data is then performed. The Log transformation method replaces each matrix value with $log_2(g_{ij})$. Mean/Median-center adjustment subtracts the mean/median of rows/columns from all values in the rows/columns such that the mean/median of each row/column is zero. It is recommended to use the median method[2] since it is more robust to noise. Deviation adjustment multiplies all row/column values with a factor so that the deviation of each row/column is one. Also, to prevent redundancy, the rows/columns are checked to ensure that no two genes have the same data.

Normalization

Normalization of gene expression data is an adjustment process for difference in labeling and detection efficiencies for the fluorescent labels and for differences in the quantity of initial RNA from the two samples examined in the assay.[3] Some techniques presented here are the most common.

Total intensity assumes the same mRNA in both experiments so that a normalization factor is used to re-scale the intensity for each gene in the array. The sum of control (A_i) and experiment (B_i) is computed according to the below equation and by multiplying this to the raw data, it is normalized:

$$N_{total} = \frac{\sum_{i=1}^{Narray} A_i}{\sum_{i=1}^{Narray} B_i}.$$

In the *globalization method* for each array, all measured values are divided by their sum (or average).[4] The mean approach uses the below equation:

$$N_{total} = \frac{1/Narray \sum_{i=1}^{Narray} A_i}{1/Narray \sum_{i=1}^{Narray} B_i}.$$

Intensity-dependent methods use the dye as the bias and display the data in a ratio-intensity plot (MA-plot):[5]

$$Log\ ratio: M = log_2 A_i - log_2 B_i,$$

$$Log - intensity: A = \frac{log_2 A_i + log_2 B_i}{2}.$$

One of the methods appropriate for linear relations is *rank invariant*.[6] Let r_{gv} be the rank of gene g in the control array v and r_{gi} the rank of g in i. To determine if a gene is rank invariant, the absolute value of the change in relative rank (r) has to be:

$$\frac{r_{gi} - r_{gv}}{r_{gv}} < 0.05.$$

Another common method is *Cubic-spline normalization* for nonlinear relationships between samples or groups of samples. This method divides the intensity distribution into a group of quantiles (q) consisting of a similar number of gene intensities:

$$q_i = \frac{i - 0.5}{N},\ N = \max\left(15, \frac{N_{probes}}{100}\right).$$

Housekeeping genes don't have a significant amount of regulation. This causes an iterative process that normalizes the mean expression ratio to one and calculates confidence limits that can be used to identify differentially expressed genes.[3] A method that considers most genes not only regulated ones is *centralization*. In this method, if $l^*_{g,k}$ is considered as the true expression level of gene g in the kth sample and $m_{g,k}$ is considered as the measured value or the signal intensity, then we have:

$$m_{g,k} = b_{g,k} + c_k d_g l^*_{g,k} + e^{res}.$$

Here, $b_{g,k}$ is the background noise, $c_k d_g$ is the proportionality between measured intensity and true number of mRNA copies and e^{res} is the error. Then the quotient $q^*_{i,j}$ is estimated according to:[4]

$$q_g = \frac{m_{g,k_i} - b_{g,k_i}}{m_{g,k_j} - b_{g,k_j}},\ g \in G_{i,j}.$$

The number of mRNA copies of each particular gene is considered as Poisson distribution instead of the normal distribution, since it arises naturally when events are counted that occur independently and with constant probability. Now, from the resulting matrix of pairwise quotients, the background intensity (μ) and the spread intensity ((), an optimally consistent scaling of the samples (s_i, s_j) is computed and is then used for scaling the data:

$$s = argmin \sum_{i,j=1}^{n} \left(\frac{\log s_i / s_j - \mu_{i,j}}{\sigma_{i,j}} \right)^2.$$

In a recent method, the expression values of majority of genes are considered to be constant between two microarray experiments.[7] In this method, assuming f_{1i} and f_{2i} are fluorescence intensities, the probability that proper expression value is e_{1i} for two samples is:

$$P_1(e_{1i}|f_{1i}, k) = \frac{1}{\sqrt{2\pi\sigma_1}} \exp\left(-\frac{e_{1i} - f_{1i}^2}{2\sigma_1^2} \right).$$

Therefore, the probability that expression value of gene i is constant between two experiments is to be:

$$P_1(e_{1i} = e_{2i}|f_{1i}, f_{2i}, k) = \frac{1}{\sqrt{2\pi\sigma_2}} \exp\left(-\frac{\{(f_{1i} - f_{2i}) - k\}^2}{2\sigma^2} \right).$$

The adjustment factor that maximizes the expected number of genes whose expression values are constant between two experiments is as follows:

$$k = argmax\left(\sum P_1(e_{1i} = e_{2i}|f_{1i}, f_{2i}, k) \right).$$

To briefly summarize this section, the various normalization techniques that are both commonly used and more efficient, are described. Normalizing the data adds more reliability on the raw datasets and makes them ready for further analysis.

Missing value imputation

Missing values occur for diverse reasons, including insufficient resolution, image corruption, due to dust or scratches on the slide or as a result of the robotic methods used to create them.

In *singular value decomposition* (SVD),[8] the k most significant basic functions or eigengenes from V are selected and a missing value j in gene i is estimated by first regressing this gene against the k eigengenes and then using the coefficients of the regression to reconstruct j from a linear combination of the k eigengenes. Each missing value in A is estimated and then the procedure is repeated on the newly obtained matrix until the total change in the matrix falls below the empirically determined threshold. Since this method is dependent on learning, the *k-nearest neighbor method* is preferred that computes the Euclidean distance between the genes not missing in the matrix and finds the k-nearest.

Regression[9] is for fitting multivariate Gaussian means and covariance's in the presence of missing data and the imputed values come as a by-product. It removes the rows of the matrix which have missing values in column j, fits the regression of the clean column j on all the other columns and uses the coefficients from the regression to make predictions at the missing locations in column j. The method could be generalized by using regression methods other than linear regression, such as regression trees, or iterative processes to perform the imputation for each column. Both SVD and KNN surpass the commonly used row average method and KNN appears to provide a more robust and sensitive method for missing value estimation as SVD.[9] *Missing log2* is another common method where transformed data that are often replaced by zeros or, less often, by an average expression over the row, or row average.

Integrative Missing Value Estimation method (iMISS) incorporates information of multiple reference microarray datasets to improve missing data imputation.[10]

GOimpute is the first algorithm that exploits the functional similarities embedded in the GeneOntology (GO) databases along with the expression similarities to facilitate the neighbor gene selection.[11]

An imputation framework called *HAImpute*[12] is used to integrate histone acetylation information to improve the estimation accuracy of the missing values in gene expression datasets.

Fixed Rank Approximation Algorithm (FRAA) is another iterative method that is briefly introduced below:

For ($p = 0$ *to iter* $- 1$)

1. Compute $A_p := G_p^T G_p$ and find an orthonormal set of eigenvectors for A_p, $v_{p,l}$, ..., $v_{p,m}$.
2. G_{p+1} is a solution to the minimum problem:

$$\min_{x \in X} \sum_{i=l+1}^{m} \sigma_i(x)^2 = \sum_{i=l+1}^{m} \sigma_i(G)^2.$$

IFRAA[13] has been introduced as a combination of FRAA and a good clustering algorithm to complete the missing data and then group the data to a small number of clusters of data with similar characteristics. Here, $f_l(X)$ or the objective function is the sum of the squares of all but the first l singular values of a matrix. The minimum of $f_l(X)$ is considered on the set X, which is the set of all possible choices of matrices $X = (x_{i,j})^{n,m}$, $i,j = 1$ such that $x_{ij} = g_{ij}$ if the entry g_{ij} is known.

Another computational approach is *Bayesian principal component analysis* (BPCA). The main steps are: principal component regression, Bayesian estimation, and expectation maximization (EM)-like repetitive algorithm. In PC regression, the missing part y_{miss} in the expression vector y is estimated from the observed part y_{obs} by using the PCA result. Let w_{obs} and w_{miss} be parts of each principal axis w_l, corresponding to the observed and missing parts, respectively, in y. By minimizing the residual error we obtain:

$$x = (W^{obsT} W^{obs})^{-1} W^{obsT} y^{obs}.$$

Using x, the missing part is estimated as:

$$y^{miss} = W^{miss} x.$$

Next, a Bayesian estimation obtains the posterior distribution of θ and X, according to the Bayes theorem:

$$p(\theta, X|Y) \propto p(Y, X|\theta)p(\theta),$$
$$P(\theta|\alpha) \equiv p(u, W, \tau|\alpha) = p(\mu, \tau)p(\tau)\prod_{j=1}^{k} p(w_j|\tau, \alpha_j).$$

Least squares estimates (LLS), iterative analysis of variance methods, Gaussian mixture clustering (GMC), collateral missing value estimation (CMVE) and projection on convex sets (POCS) are the other methods that are less common and not expressed here. This section reviewed the most common approaches in preprocessing gene expression data. Comparing the methods in this section is greatly dependent on the type of datasets used. Statistical, large, complex data use more recent methods described, while other types can use the first methods mentioned in this survey. This preprocesses data is now ready to be analyzed more deeply.

Analysis of Gene Expression Data

Although the microarray technology opened new horizons in finding the expression level of genes, the data still has high dimension in the number of genes, and noise is unavoidable during the extraction phase. In order to reduce the dimensions in gene data, we use methods for analyzing these data. Finding out what the gene expression levels stand for in each problem can help reduce the irrelevant data and is also more helpful to biologists. The most common methods are classified as below.

Classification

Classification has the goal of determining whether an object belongs to a certain class. Classification techniques can be used in microarray analysis to predict sample phenotypes based on gene expression patterns. A *predictor* or *classifier* for K classes partitions the space of gene expression data into K disjoint and exhaustive subsets, $A_1, \ldots, A_k$. Predictors are built from past experience. Such observations comprise the *learning set* (LS),

$$L = \{(x_1, y_1), \ldots, (x_{nL}, y_{nL})\}.$$

Predictors may then be applied to a *test set* (TS), $T = \{x_1, \ldots, x_{nT}\}$ to predict the class y_i. In this section, some of these discrimination methods are examined more closely.

In *Bayes Approach*, if class priors ($\prod_k$) and class conditional densities $p_k(x)$ are known, Bayes theorem gives the posterior probability $p(k|x)$ of class k given feature vector x:

$$p(k|x) = \frac{\prod_k p_k(x)}{\sum_t \prod_t p_t(x)}.$$

Bayes rule identifies those genes that are most likely to confer high classifier accuracy. Here we model each class as a set of Gaussian distributions, one for each gene from the training samples of that class.[14]

Dimensional Reduction[15] is a solution that preselects a subset of genes likely to be predictive and then investigates in depth the relationship between these and the phenotype of interest.

Principal component analysis is used to reduce the dimensionality of a dataset. This is done by projecting the data of a dimensionality N onto the eigenvectors of their covariance matrix with, usually, the largest M eigenvalues taken

$$PC_i = \sum_{j=1}^{M} b_{ij} X_j,$$

where X_j is the jth original attribute, and b_{ij} are the linear factors (eignvectors) which are chosen so as to make the variance as large as possible.

Information Gain Based Feature Ranking defines the entropy of the class before and after observing the attribute, respectively:

$$H(C) = -\sum_{c \in C} p(c) \log_2 p(c),$$
$$H(C|X) = -\sum_{x \in X} p(x) \sum_{c \in C} p(c|x) \log_2 p(c|x),$$

where x is a feature value and c is a class label. Basic methods are then used to classify the data using these two techniques.

Classifier Evaluation is used because of the probabilistic nature of the predictions and can be done using data other than those used to develop the classifier.

The *nearest centroid classifier* computes a centroid given by the average expression levels of the samples in the class, and then assigns new samples to the class whose centroid is nearest.

Top-scoring pair (TSP) classifier looks for pairs of genes such that gene 1 is greater than gene 2 in class A and smaller in class B.

Nearest-neighbors classifiers assign samples to classes by matching the gene expression profile to that of samples whose class is known:

- Use distance function (Euclidean, Mahalanobis, Manhattan) to define k nearest neighbors for x.
- Predict class by majority vote (most common among neighbors).

Support vector machines (SVMs) operate by finding a hyper-surface in the space of gene expression profiles, that will split the groups so that there is the largest distance between the hyper-surface and the nearest of the points in the groups. By maximizing the margin, this method avoids over fitting. If the datasets are not linearly separable, the data is mapped into higher dimensional space.

Discriminate analysis optimally partitions a space of expression profiles into subsets that are highly predictive of the phenotype of interest, for example by maximizing the ratio of between-classes variance to within-class variance. The criterion to be maximized is:

$$J(w) = \frac{|m_1 - m_2|^2}{s_1^2 + s_2^2},$$

where $m_k = w^T m_k$ is the projection of mean m of class k and

$$s_k^2 = \sum_{y \in rank} (y - m_k)^2,$$

is the scatter for projected samples y of classes.

Classification trees recursively partition the space of expression profiles at the root into subsets at the leaves that are highly predictive of the phenotype of interest. Rules can be derived from the tree by following a path from the root to a leaf. Two most common methods are C4.5 and CART which recursively grow a tree top-down. The decision tree algorithm is well known for its robustness, learning time complexity and more importantly, being more precise than other classification methods such as SVM and discriminate analysis.

Clustering based approach can be used to classify DNA array data where training and test samples are mixed together and clustered without supervision to produce a specified number of clusters.[16] The 'G-S algorithm' is applicable to datasets with two classes and uses the training data to compute a mean and standard deviation for each gene's level of expression.

In *Model based classification*,[17] rules are defined according to the situation examined individually to extract information, or used in groups to predict membership in some class or to predict the values of a "target" attribute (high-risk patients, and tumor stage). Results are extracted using software (ie, HAMB) which automatically selects the feature set and parameters for each of the target attributes using search and heuristics as well as pruning and grouping similar rules.

An *ensemble method* combines multiple classifiers built on a set of re-sampled training datasets or generated from various classification methods on a training dataset. Binding three different supervised machine learning techniques, namely C4.5, bagged and boosted decision trees shows better results compared to each single method learning by itself.[18]

Boosting aggregates models produced by a 'weak learner' into an effective classifier. The final aggregate classifies a sample according to votes from its models, each weighted according to its accuracy on the data with which it was trained. This method is computationally expensive since the entire training dataset must be examined to train the weak learner during each iteration.

Bagging[19] aims to manipulate the training data by randomly replacing the original T training data by N items:

For each iteration $i = 1, ..., j$

- Select subset t from training examples T.
- Generate classifier $C_i(x)$ from t.
- The final classifier $C^*(x)$ is formed by aggregating the j classifiers.
- To classify an instance x, a vote for class y is recorded by every classifier $C_i(x) = y$.

The *AdaBoost*[20] is an alternative method to influence the training data:

Assigns an equal weight for instance x_i, for each iteration $i = 1, ..., j$

- Generate classifier $C_i(x)$ with minimizing the weighted error over the instances x.
- Update the weight of x_i.

The final classifier $C^*(x)$ is formed by a weighted vote of the individual $C_i(x)$ according to its accuracy on the weighted training set.

A simple method is the *fuzzy inference*[21] that has the theoretical advantage not needing to be retrained when using measurements obtained from a different type of microarray. For each gene g in question, we define fuzzy membership functions $g_u, g_n, g_d: U \to [0,1]$ as follows. Let $ramp: R^3 \to [0,1]$ be defined as

$$ramp(x,v,w) = \begin{cases} 0 & if\ x \le v, \\ 1 & if\ x \ge w, \\ \dfrac{x-v}{w-v} & otherwise. \end{cases}$$

Associate with each gene g a triplet

$$(g_{min}, g_{median}, g_{max}) \in R^3,$$

such that

$$g_{min} \le g_{median} \le g_{max}.$$

Using this, we define:

$$g_u(x) = ramp(g(x), g_{median}, g_{max}),$$
$$g_d(x) = 1 - ramp(g(x), g_{min}, g_{median}),$$
$$g_n(x) = 1 - max(g_u(x), g_d(x)).$$

An application $r(x)$ of a rule r = (D', c) to an element x, is defined as:

$$r(x) = min_{(g,l)\in D'}(g_l(x)).$$

Given the associated thresholds t_c, we define:

$$class_R(x) = \left\{ c | c_R^{t_c}(x) = max_{c'\in C}\left(c'^{t_{c'}}_R(x) \right) \right\}.$$

All the classification algorithms explained have been used in various experiments to analyze the gene expression datasets. Whether one method is superior over another depends on the dataset, evaluation algorithm and the cross-validation method used to classify the data. This is a topic mainly discussed in the statistical machine learning area.

Differentially expressed genes

In DEGs, subsets of genes that have a significant expression level are used to group genes into up-regulated (having high measures in the treatment experiment compared to the control) and down-regulated groups. The types of experiments include: *single-slide* (expression levels on one slide) and *multiple-slide* (different slides). For single-slide methods, the first ideas relied on fold increase/decrease cut-offs to identify differentially expressed genes using certain threshold (mean, variation or log intensities). Recent methods turned to *probabilistic modeling* and differ mainly in the distributional assumptions they make for (*Red;Green*). Each *(R,G)* is normally and independently distributed with constant CV.[22] *Hierarchical model*[23] (Gamma-Gamma-Bernoulli model) is suggested for $(R;G)$ to identify differentially expressed genes based on the posterior odds of change (functions of $R + G$ and RG) under this hierarchical model. The following statistical methods can be used to identify expression levels:[24]

Average difference:

$$\frac{1}{8}\left(\sum_{i=1}^{8} T_i - \sum_{i=1}^{8} C_i \right) = \bar{T} - \bar{C}.$$

T-statistic:

$$t = \frac{\bar{T} - \bar{C}}{S_p \sqrt{\dfrac{1}{n_T} + \dfrac{1}{n_C}}}.$$

Empirical Bayes Method:

$$t^* = \frac{\bar{T} - \bar{C}}{\left(a + S_p\right) \sqrt{\dfrac{1}{n_T} + \dfrac{1}{n_C}}}.$$

Mixture models: Normal mixture model for log-ratios in two colour arrays for DE genes and non-DE genes.

Error models:

$$T = \mu_T + \mu_T \varepsilon + \delta, C^* = \mu_{C^*} + \mu_{C^*} \varepsilon' + \delta',$$
$$log(n_{ijkg}) = \mu + A_t + D_j + V_k + G_g + (AC)_{ig} + (VC)_{kg} + \varepsilon_{ijkg},$$

(A = Arrays, D = Dye, V = T-C, G = Gene, VG = Term of interest and value).

Identification of DEGs from multiple microarray analysis data is done by considering clusters of data and finding the different expression levels within the clusters. A common approach performs a *hypothesis test* for each gene. These tests use the null hypothesis

$H = 0$ and $H = 1$ otherwise. A random variable is used to show the probability of H:

$$p-value(x) = \inf_{\Gamma_\alpha : x \in \Gamma_\alpha} \left\{ Prob\left(X_g \in \Gamma_\alpha \mid H_g = 0 \right) \right\}.$$

Standard hypothesis tests like the linear models, linear regression and ANOVA can be applied here. To improve the performance of the current method, variance estimates are changed by penalizing or controlling those using learning approaches such as *Bayesian models*. An improved random search algorithm is introduced for identifying DEG combinations to generate candidate gene combinations of a given size.[25] Cross validation is used to provide replication stability of the search procedure:

Repeat v times for v optimal sets:

- Randomly draw u_1 samples from one group of arrays and u_2 from other.
- Leave out the selected arrays; find optimal subset of genes using data from remaining group.

A *t-like score*[26] is calculated for each gene. Genes with scores greater than an adjusted threshold are said to have altered expressions under the treatment. Permutations are used to calculate the false discovery rate. Genes that are assigned into a cluster with a mean t value, significantly different from zero, are declared to be differentially expressed.

In *Hierarchical mixture model*,[27] the expected expression values of the ith gene under the control and the treatment are assumed to arise from a mixture of three distributions. The cluster of genes with $\mu_1 i! = \mu_2 i$ are declared as DEG.

The *linear model and empirical Bayes method*[28] is a hierarchical linear model to describe expression levels. A moderated t-statistic with extra information borrowed from other genes, is calculated for each gene. Adjusted P-values are calculated for genes based on the moderated t-statistic to achieve control.

A new *Bayesian mixture model*[29] approach is introduced to classify genes into one of three clusters, corresponding to clusters of down-regulated, neutral, and up-regulated genes, respectively. The Bayesian method is implemented via the Markov Chain Monte Carlo (MCMC) algorithm. The cluster means of down- and up-regulated genes are sampled from truncated normal distributions whereas the cluster mean of the neutral genes is set to zero. Genes assigned into significant clusters are differentially expressed.

Clustering

Clustering is essential in data mining structures and identifying patterns in underlying data. Cluster analysis seeks to partition a given dataset into groups based on specified features so that the data points within a group are more similar to each other than the points in different groups. Clustering is an example of *unsupervised classification* so it doesn't rely on predefined classes and training examples. Genes can be grouped in clusters based on their expression patterns which are called *gene-based clustering* where genes are treated as the objects, while the samples are the features.[30] Samples can be partitioned into homogeneous groups, corresponding to macroscopic phenotype, such as clinical syndromes or cancer types and is defined as *sample-based clustering*. Current molecular biology also reveals that only a small subset of genes participates in cellular process and so *subspace clustering* captures clusters formed by a subset of genes across a subset of samples.

The *K-means algorithm*[31] partitions the dataset into k disjoint subsets to optimize the following objective function:

$$E = \sum_{i=1}^{K} \sum_{O \in C_i} |O - \mu_i|^2,$$

where O is a data object in cluster and μ_i is the centroid of C_i. This algorithm is simple and fast. In addition, it not only is sensitive to noise but also needs the optimal value for k.

The *"soft" k-means clustering* is done via an *Expectation Maximization* (EM) algorithm where each cluster represented by a distribution (eg, a Gaussian), the E step determines how likely it is that each cluster generates each instance and the M step adjusts the cluster parameters to maximize likelihood of instances.

Deterministic Annealing[32] uses the same cost function, but rather than minimizing it for a fixed value of clusters K, it performs a statistical mechanics type analysis using a maximum entropy principle as its starting point. The resulting free energy is a complex function of the number of centroids and their locations. Minimization is done by lowering the temperature variable.

Self-organizing maps (SOM)[33] are based on a single layer neural network. Each data object acts as a training sample to fit the distributions of the input dataset. Clusters are identified by mapping all data points to the output neurons. This method is more robust to noise but it needs the number of clusters and the network structure.

SOMs are combined with *Particle Swarm Optimization* (PSO) to cluster gene data.[34] To find an optimal or near-optimal solution to the problem, PSO updates the current generation of particles using the information about the best solution obtained by each particle and the entire population. The weights are trained by SOM in the first stage. In the second stage, they are optimized by PSO. Each particle has current velocity, current position, the best position discovered by the particle so far and the best position discovered by the particle and its neighbors so far. The *n*th component of the new velocity and the new position is updated:

$$V_{i,n}(t+1) = wV_{i,n}(t) + c_1\left(G_i(t) - X_{i,n}(t)\right) + c_2\left(l_{i,n}(t) - X_{i,n}(t)\right),$$
$$X_{i,n}(t+1) = X_{i,n}(t) + V_{i,n}(t+1).$$

Hierarchical clustering generates a hierarchical series of nested clusters which can be graphically represented by a tree. The branches record the formation of the clusters and the similarity between the clusters. If the tree is built in a top-down manner, divisive algorithms are used to decide how to split clusters at each step. The distance is computed using any on the following methods:

1. Single link: distance of two most similar instances
$$dist(c_u, c_v) = \min\{dist(a, b) | a \in c_u, b \in c_v\}.$$
2. Complete link: distance of two least similar instances
$$dist(c_u, c_v) = \max\{dist(a, b) | a \in c_u, b \in c_v\}.$$
3. Average link: average distance of between instances
$$dist(c_u, c_v) = \text{avg}\{dist(a, b) | a \in c_u, b \in c_v\}.$$

A *deterministic annealing algorithm* is used where starting by two clusters, the probability of a gene belonging to a cluster is computed by the Gaussian model, and the cluster centroid is:[52]

$$\frac{\Sigma_k \overrightarrow{g_k} P_j(\overrightarrow{g_k})}{\Sigma_k P_j(\overrightarrow{g_k})}.$$

The iterative algorithm EM is then used until each cluster contains one gene.

Agglomerative algorithms are used when the approach is bottom-up where the different measures of *cluster proximity* derive various merge strategies.

Density-based hieratical clustering[35] considers a cluster as a high-dimensional dense area, where data objects are "attracted" with each other. At the "core" part of the dense area, objects are crowded closely with each other, and thus have high density. Hierarchal clustering is sensitive to noise and is also greedy and might fall in a local minimum.

Fuzzy logic is used to produce a probability vector for each observation. A hard cluster is determined by assigning an observation to a group which has the highest probability. Usually the Manhattan distance which is more robust than the Euclidean distance, is used.[36]

The *naïve Bayesian clustering*[37] has been introduced along with a new extension in using context-specific method in the clustering. It uses the Gaussian model and the EM learning algorithm for clustering. The probability of each cluster is computed and compared to the other clusters. The joint distribution will be in the form:

$$P(X_1, \ldots, X_N | G) = \left(\prod_{i \in G} P(X_i)\right) \times \sum_k \left(P(C = k) \prod_{i \in G} P(X_i | C = k)\right).$$

A binding site has high probability in promoter regions of genes in these two categories, and low probability of appearing in the promoter region of all other genes:

$$P(X_i | C = k) = \{P(X_i | C = K)k = k_j \in L_i | P(X_i | k \notin L_i) otherwise\}.$$

A scoring function is used as the optimization problem and the maximum likelihood method is used for model M and dimension D:

$$\log P(D|M) = \log P(D|M, \vec{\theta}_M) - \frac{1}{2}\log MdimM + O(1).$$

The method detects the number of clusters and is robust to random variables.

Graph-theoretical clustering techniques convert the problem of clustering a dataset into graph theoretical problems like finding *minimum cut* as in CLICK[38] or *maximal cliques* in CAST[39] of the proximity graph. CLICK recursively divides the graph in two, using minimum weight cut computations, until the sum of the weights of the discarded vertices is minimum. An EM method is used to build two similarity distributions.

The second has the difficulty of determining a "good" value for the global parameter for the clique. First a non-clustered point is chosen and placed into its own cluster and then an iterative step is used as follows:

- If distance from any non-clustered point is less than threshold, add point to the cluster.
- If distance from any point in the cluster to the other points is greater than threshold, remove point from cluster.

A *minimum spanning tree* (MST)[40] is used to cluster the gene data. The idea is to partition the main tree into K subtrees so that the total edge-distance of all K subtrees is minimized. The work presents three algorithms based on this theory. First, it intends to capture the intuition that two data points with a short edge-distance should belong to the same cluster and data points with a long edge-distance should belong to different clusters and hence be cut. Second, it attempts to partition the minimum spanning tree T into K subtrees to optimize a more general objective function than the previous one:

$$\sum_{i=1}^{K}\sum_{d\in T_i}\rho(d, center(T_i)).$$

For each pair of adjacent clusters it goes through all tree edges within the merged cluster to find the edge to cut. A more global algorithm is presented as the third algorithm which representatives are the results of the optimization process and attempts to partition the tree into K subtrees and simultaneously to select K representatives in such a way to optimize the objective function.

In a more interesting case, MST's are used along with *genetic algorithms* to overcome less promising local optima and find more optimal solutions. First, the MST is computed using Prim's algorithm. Then, for each edge in the MST, a mutual neighborhood value is calculated that will be used in the local search. Next, the fitness function is estimated using the following:

$$min\sum_{i=1}^{k}\left(\left(\sum_{x_i,x_j\in C_i}\frac{d^2(x_i,x_j)}{|C_i|}\right)+p\right),$$

where C_i denotes the number of cluster members, k represents the number of clusters and p is a term to penalize results. The local search is followed: for each individual, a list of deleted and non-deleted edges is created. Selection for the GA is applied twice during the main loop; for variation (recombination and mutation) individuals are randomly selected; to determine parents' selection for survival is performed on a pool consisting of all parents of the current generation and the offspring. As the mutation operator, a simple modified point mutation is applied.

The *p-quasi complete linkage clustering* is used.[41] Since the p-quasi complete graph problem is NP-complete, an approximate algorithm for finding maximal p-quasi complete subgraphs is introduced. The algorithm initially starts clustering from each vertex as a cluster, and repeatedly adds adjacent vertices to the existing subgraphs while the p-quasi completeness condition holds.

Dynamical Clustering[42] is a partitional iterative algorithm that optimizes the best fitting between classes and their representation using a predefined number of classes. It works on two alternates steps: an allocation step, where all individuals are allocated to the class with the prototype with lower dissimilarity, followed by a representation step where a prototype is constructed for each class.

Model-based clustering approaches[43] provide a statistical framework to model the cluster structure of gene expression data. The dataset is assumed to come from a finite mixture of underlying probability distributions, with each component corresponding to a different cluster. The goal is to estimate two parameters of L such that the result below is maximized:

$$L_{mix}(\theta,\Gamma)=\sum_{i=1}^{k}\gamma_r^i f_i(x_r|\theta_i),$$

where N is the number of data objects, k denotes the number of components, x_r represents the data object, f shows the density function, θ_i denotes the model parameter and γ_r is the hidden parameter. Usually, the parameters are estimated by the EM algorithm.

Coupled two-way clustering[44] provides a heuristic to avoid brute-force enumeration of all possible combinations. Only subsets identified as stable clusters in previous iterations are candidates for the next iteration. The main motivation is to increase the signal to noise ratio of the expression data.

The *plaid model*[45] regards gene expression data as a sum of multiple "layers", where each layer may represent the presence of a particular biological process with only a subset of genes and a subset of samples involved.

The *bicluster concept*[46] is to model a block, along with a score called the mean-squared residue to measure the coherence of genes and conditions in the block. The score for two gene pairs is as follows:

$$H(G',S') = \frac{1}{|G'||S'|}\sum_{i\in G', j\in S'}(w_{ij} - \eta_i S' - \eta G'_j + \eta G' S').$$

Since bi-clustering is NP, it needs approximate solutions: Hill-climbing, graph hashing technique, evolutionary algorithms, simulated annealing and simple divide-and-conquer are some of the most important algorithms used in bi-clustering gene data.

Triclustering[47] is mining coherent clusters in three-dimensional gene expression datasets. Tricluster relies on graph-based approach to mine valid clusters and merge/delete clusters having large overlaps.

This section briefly reviewed the current state of art in clustering as a main approach to analyzing gene data. The approaches mentioned all have high performance in the datasets used and since clustering is an unsupervised learning method, only the dataset and the different type of algorithm determine the efficiency of the clustering approach. There is no unique method defined as the best clustering approach and the results greatly depend on the parameters explained in each of the methods. In the next section these data will be used to find a network model.

Modeling gene expression data

Increase in the amount of data produced in the future will need automated ways of discovering patterns and the structure of the underlying causal network that is assumed to generate the observed data. The inference of discrete models from gene expression data is not trivial because of high dimension, continuous valued expression data, lack of large number of experiments coarse and uneven sampling, but these methods can help getting insights in the cell machinery.

The most straightforward way to model a network is to view it as a *Directed Graph*.[48] The vertices correspond to the genes, while the edges denote interactions among them. A number of operations on graphs can be carried out to make biologically relevant predictions about a regulatory system. A search for paths between two genes may reveal missing regulatory interactions or provide clues about redundancy in the network.

In the *Boolean Network Model*,[49] expression levels of genes are represented as Boolean variables. The operations of BNs are usually represented by diagrams where input nodes are genes at the current time step t and output nodes are genes at the next time step $t + 1$. In order to obtain a BN from a time series of gene data, the following algorithm is performed:

- Randomly construct a random n, K, BN
- repeat
  - select initial state $s(0)$ randomly
  - $s \leftarrow s(0)$
  - Repeat
    - if Is the *Network* uniquely identified then Return
    - $s \leftarrow$ state of *Network* following s
- until a cycle is detected

Real biological systems are continuous and non-binary, however, owing to its simplicity; the proposed algorithm can be extended for them. Random Boolean Networks (RBN)[50] are simplified models of BNs. Glass networks are continuous time networks. In a network with N nodes, each with k inputs, we define the activation rate of node i as below where f_i is a BN of inputs:

$$\frac{dx_i}{dt} = -\tau_i x_i + f_i\left(X_{i_1}(t), X_{i_2}(t), \ldots, X_{i_k}(t)\right).$$

In *Probabilistic Boolean Networks*,[51] the transition is almost always stochastic. For any gene g, assume there are l_g number of Boolean functions: $f_1(g)$, …., $f_{l_g}(g)$ with selection probabilities: $c_1^{(g)},\ldots c_{l_g}^{(g)}$ and the transition probability is any given $S(t)$ to any $S(t+1)$. We can easily create a stochastic matrix in the state space of genes. This stochastic matrix is Markov chain and the analysis of the dynamics of the network can be done using MC theory. A context-sensitive PBN which is mainly used in biological context consists of a set of n nodes and a set of vector-valued functions, called predictor functions. Suppose that the activity level of gene i at time step k is denoted by $x_i(k)$. The overall expression levels of all the genes in the network at time step k is given by the row vector

$$x(k) = [x_1(k), \ldots, x_n(k)].$$

The expected immediate reward earned in state i, when control u is selected, is defined by:

$$\bar{r}(i,u) = \sum_j p(u) r(i,u,j).$$

The infinite expected total discounted reward, given the policy p and the initial state i, is:

$$J_\Pi(i) = \lim_{N\to\infty} E\left\{ \sum_{t=0}^{N-1} \lambda^t r(i, \mu_t(i), j) \right\},$$

where vector j is the value function. We seek a policy that maximizes the value function for each state i:[52]

$$J^*(i) = \max_{\pi\in\Pi} J_\pi(i) \text{ for all } i \in S.$$

Markov Chain Models[53,54] are constructed directly from gene expression data. Mathematical modeling tools that allow estimation of steady-state behavior in biological systems would be useful for examining two ubiquitous forms of biological system behavior. The first is homeostasis, the ability of cells to maintain their ongoing processes within the narrow ranges compatible with survival, and the second is a switch-like functionality that allows cells to rapidly transition within limited process segments between meta-stable states. The definition of the Markov chain between a state at step t and the state at step $t+1$ and the transition rule is depicted as bellow:[53]

$$g \downarrow$$

Based on the transition rule, the transition probability between any two arbitrary states of the Markov chain is as follows:

$$Pr\{S^{(t)} \to S^{(t+1)}\} = \prod_{i=1}^{n} C_l^{g_l^{(t+1)}}.$$

The simulation algorithm used in this study is summarized as follows:

- Randomly initialize $S(0)$.
- Start Markov chain from $S(t)$ to $S(t+1)$ based on transition rule:
  - If perturbation flag = true derive $S(t+1)$ using perturbation rule.
  - else use known conditional probabilities to derive $S(t+1)$.
- Start to collect sample from $S(T+1)$ to $S(T+\mathrm{N})$.

In *Linear/Quasi Linear Models*,[55] each gene is modeled as a linear combination of other genes. Being quasi linear means applying a non-linear function (eg, sigmoid) on the linear combination. This is done to prevent unbounded growing of expression values. The transcription response of gene i to $r_i(t)$ is calculated with a dose-response or "squashing" function:

$$x_i(t+1) = \frac{1}{1+e^{-(\alpha_i r_i(t)+\beta_i)}},$$

where $r_i(t)$ is the state of gene i, a_i and b_i are constants defining shape of response curve for i. Using Neural Networks methods, the equations are further simplified:

$$u_i(t+1) = m_i x_i(t+1) = \frac{m_i}{1+e^{-\Sigma z_{ij} u_j(t)}}.$$

Differential Equations, including ODE, PDE and DDE, model the rate of change of the concentration for genes. A mathematical model for genetic regulatory networks with time delays can be given as follows.[56] Let $m(t) \in R^n$ and $p(t) \in R^n$ be the concentrations of mRNA and proteins respectively. Then the gene regulatory network can be described by the following system of ordinary differential equations:

$$\begin{cases} \dot{m}(t) = -K_m m(t) + c(p(t,\tau_p)), \\ \dot{p}(t) = -K_p p(t) + d(m(t,\tau_m)), \end{cases}$$

where $K_m = diag(K_{m1}, \ldots, K_{mn}) \in R^{n\times n}$ and $K_p = diag(K_{p1}, \ldots, K_{pn}) \in R^{n\times n}$ are positive real diagonal matrices that represent the degradation rates for mRNAs and proteins respectively. $\tau_m \in R^n$ and $\tau_p \in R^n$ are positive real vectors indicating the time delays for mRNA and proteins respectively, and

$$\begin{cases} m(t, \tau_m) = (m_1(t - \tau_{m1}), \ldots, m_n(t - \tau_{mn})), \\ p(t, \tau_p) = (p_1(t - \tau_{p1}), \ldots, p_n(t - \tau_{pn})). \end{cases}$$

Also, $c(p(t)) \in R^n$ and $d(m(t)) \in R^n$ are nonlinear functions. It is worth noting that the first equation of the above system describes the transcription process while the second one implies the translation process.

Two important categories in modeling gene regulatory networks including deterministic and stochastic modeling approaches are discussed and compared with each other. In particular, the role of principles from feedback control theory in understanding biological behaviors such as mono- and multistability of the system, existence and stability of oscillations, noise resistance, etc. has been shown. Also, a variety of networks characteristics that are in direct relationship with the presence of both positive and negative feedback loops in gene regulatory networks, are discussed.[57]

Quantitative stochastic models of molecular interaction networks can be expressed as *stochastic Petri nets* (SPNs).[58] An SPN comprises a set of places P, a set of transitions T, an input function I, an output function O, a weight function W, and an initial marking M_0. To represent a system of molecular interactions as an SPN, each place represents a distinct molecular species. Each transition represents an elementary chemical reaction. SPN can be converted into instantaneous transitions to form a Petri-net and numerical analysis can be used to derive both steady-state and transient behavior. In addition to standard Petri nets, Hybrid Functional Petri Nets (HFPN) contains continuous place nodes and continuous transitions.

In *Neural Networks approaches*,[59] one of the aims is to gain robustness against stochastic noise. Random variables are introduced to model synthesis and degradation processes. Given the input/output pairs, networks weights are learned by training. The stochastic model is based on Poisson random variables in the form:

$$u_{i(n+1)} = u_{in} + P(s_i g_i(t_n) h_n) - P(d_i s u_{in} h_n).$$

Genetic Programming[60] identifies a target gene for which the network is to be modeled. The approach to the problem of automatically creating both the topology and sizing of a network of chemical reactions involves:

1. Establishing program trees.
2. Converting each tree in the population into an analog electrical circuit representing the network of chemical reactions.
3. Obtaining behavior of network of chemical reactions by simulating the electrical circuit.
4. Defining a fitness measure that measures how well the network matches the observed data.
5. Applying GP to breed a population of improving program trees using the fitness measure.

Methods for modeling the covariance structure of high dimensional distributions with a focus on sparse structure have been developed.[61] Taking x for a random normal p–vector, a natural, direct route to specifying a set of uni-variate models that cohere is via a "triangular" model that defines the joint distribution by composition:

$$p(x) = \prod_{i=1}^{p-1} p(x_i | x_{(i+1):p}) p(x_p).$$

The result of inducing a model from the linear regression is a compositional network.

In the modeling framework,[62] models of regulatory networks are represented as *Bayesian networks*, allowing the models to compactly and robustly capture probabilistic multivariate statistical dependencies between the various cellular factors in these networks. The models are discovered using a heuristic search algorithm based on simulated annealing to visit high-scoring regions of the model posterior and then using posterior model averaging to compute likely statistical dependencies between model variables.

Controlling regulatory networks

One direction of research on regulatory models is how to interact efficiently with the model to find perturbations or external controls leading to desirable states of the model. The other advantage is to help scientists in making further experiments with some recommendations, as well as in generating plans from the domain in order to achieve the target of reaching the

desired states.[63] The model used is based on Markov Decision Process (MDP). This Gene regulation network model M is a triple

$$M = < S; A; T >,$$

where S is the discrete state-space, A is the discrete action space and T is the transition function or behavior of the model.

Intervention has been considered in the context of *Probabilistic Boolean networks*. They have been considered where the transition probabilities between the various states can be altered by the choice of some *control inputs*. The control objective here would be to "optimally" apply one or more treatments so that an appropriate cost function is minimized over a finite number of steps, which we will refer to as the *treatment horizon*.[54,64]

Optimal control problems can be solved by using the technique of Dynamic Programming. Let $J^*(z(0))$ be the optimal value of the cost function equation. Then

$$J^*(z(0)) = J_0(z(0)),$$
$$J^M(z(M)) = C^M(z(M)),$$
$$J^k(z(k)) = min_{v(k)\in\{1,2,\ldots,2m\}},$$
$$E\{C_k(z(k), v(k)) + J_{k+1}[z(k+1)]\}.$$

If $\nu^*(k) = \mu_k^*(z(k))$ minimizes the right hand side of the equation for each $z(k)$ and k, the control law $\pi^* = \{\mu_0^*, \mu_1^*, \ldots, \mu_{N-1}^*\}$ is optimal. The DP solution will cause the equation to become as follows

$$J^*_k(z(k)) = \min_{Tv(k)\in 1,\ldots,2m} \left[C_k(z(k), v(k) + \sum_{j=1}^{2n} a_Z(k)\, j(v(k)) J_{k+1}(j) \right].$$

Various studies have considered the design of optimal infinite-horizon control for context-sensitive probabilistic Boolean networks (PBNs) too. The stationary policy obtained is independent of time and dependent on the current state. Discounted problems with bounded cost per stage and on average-cost-per-stage problems are considered.[65] Here, the cost per stage $g(i, u, j)$ is bounded for all i, $j\in S$, $u\in C$, and a discounting factor $\alpha\in(0, 1)$ is introduced in the cost to make sure that the limit of the finite sums converges as the horizon length goes to infinity. More specifically, the objective is to find a policy $\prod = \{\mu_0, \mu_1, \ldots\}$ where $\mu_t: S \to C$ which minimizes the cost function:

$$J_\Pi(z_0) = \lim_{M\to\infty} E\left\{ \sum_{t=0}^{M-1} \alpha^t g(z_t, \mu_t(z_t), w_t) \right\}$$

For the optimal control problem of this section, the main formula for finite horizon will have to be replaced by:

$$J_t(i) = \min_{u\in C} \left\{ g(i,u) + \alpha \sum_{j=0}^{2n-1} p_{i,j}(u) J_{t+1}(j) \right\}.$$

For larger biological models involving interactions among many genes, a stochastic control method that has polynomial time complexity is needed.[66] One approach is to use *Backward/forward sparse sampling* which approximates the value function with reduced complexity. A *reinforcement learning* (RL) method can also be used to overcome the curse of dimensionality. The proposed approximate method can yield a near-optimal stationary policy.[52] In RL, approximate value functions are learned through interaction with the model. By this way, value functions are learned for states that can actually be seen. *Parametric function approximation* is also useful when features can be generated from the state information. Supervised learning techniques like NNs can be used to learn approximate value functions.

Given the distributions, a reinforcement learning algorithm progressively computes the value function of a given policy by generating several sample trajectories of the PBN and their associated costs. The Bellman optimality equation for each state $i \in S$ is:

$$J^*(i) = \max_{u\in C} \left\{ \sum_{j=0}^{d^n-1} (p_{ij}(u) r(i,u,j) + \lambda J^*(j)) \right\}.$$

Accordingly, we can define the Q-factor for each state control pair (i,u) by

$$Q^{k+1}(i,u) \leftarrow (1-\alpha) Q^k(i,u) + \alpha \left[r(i,u,j) + \lambda \max_{u'\in C} \left[Q^k(j,u') \right] \right].$$

Because of complexity issues, in order to use PBNs in practice, a new multi-variant Markov model is used to approximate the PBN.[48] This first-order model is used to model the relation between the sequences of genes:

$$V_{i+1}^{(j)} = \sum_{j=1}^{n} \lambda_{ij} P^{ij} V_t^j,\ \lambda_{ij} \geq 0 \text{ and } \sum_j \lambda_{ij} = 1.$$

The transition frequency of each state in each sequence forms a matrix representing P and λ is estimated using the following formula:

$$\min_{\lambda} \max k \left| \left[\lambda_{ii} \widehat{P^{ii} V^i} + \sum\nolimits_{j=1, i \neq j}^{n} \lambda_{ij} \hat{V}^j - \hat{V}^i \right]_k \right|.$$

The objective here is to minimize the overall average of the distances of the state vectors:

$$\min_{v \in U(T)} \frac{1}{T} \sum_{\tau=1}^{T} \left\| v(i_\tau \ldots i_1) - z \right\|_2.$$

In order to solve the above equation, the following linear estimation is used:

$$\begin{aligned} D(v(w_{t-1}), t-1, k) = min \{ & \| v_0(w_{t-1}) - z \|_2 \\ & + D(v_0(w_{t-1}), t, k), \| v_1(w_{t-1}) - z \|_2 \\ & + D(v_1(w_{t-1}), t, k+1) \}. \end{aligned}$$

The work done in this section is still at hand and few methods have been applied yet. Using the models introduced in the previous section, the data can be used to control the network using computational methods specially learning mechanisms.

Conclusion

This review has tried to demonstrate the most important computational and mathematical methods used so far in the context of gene expression data. Gathering the information with the least possible noise from the microarray is the first step including missing value imputation, normalizing the data and preprocessing. Well-known computer algorithms like clustering and classifying are used to discrete the data. After obtaining the knowledge, the data will be further used to induce a model. This model is then used in controlling the overall network of genes.

Disclosure

This manuscript has been read and approved by all authors. This paper is unique and is not under consideration by any other publication and has not been published elsewhere. The authors and peer reviewers of this paper report no conflicts of interest. The authors confirm that they have permission to reproduce any copyrighted material.

References

1. Chalidabhongse TH, Kim K, Harwood D, Davis L. A perturbation method for evaluating background subtraction algorithms. *Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*. 2003.
2. Eisen MB, Spellman PT, Brown PO, Botstein D. Cluster analysis and display of genome-wide expression patterns. *Proc Natl Acad Sci U S A*. 1998;95:14863–8.
3. Quackenbush J. Computational Analysis of Microarray Data. *Macmillan Magazines*. 2001;2:418–27.
4. Zien A, Aigner T, Zimmer R, Lengauer T. Centralization: A new method for the normalization of gene expression data. *Bioinformatics*. 2001;17: S323–31.
5. Cleveland WS, Devlin SJ. Locally weighted regression: an approach to regression analysis by local fitting. *J Am Stat Assoc*. 1988;83:596–610.
6. BeadStudio Normalization Algorithms for Gene Expression Data. *Technical Note: Illumina® RNA Analysis*. 2007.
7. Kano M, Kashima H, Shibuya T. A Method for Normalization of Gene Expression Data. *Genome Informatics*. 2003;14:336–7.
8. Troyanskaya O, Cantor M, Sherlock G, et al. Missing value estimation methods for DNA microarrays. *Bioinformatics*. 2001;17:520–5.
9. Datta A, Choudhary A, Bittner ML, Dougherty ER. External Control in Markovian Genetic Regulatory Networks. *Machine Learning*. 2003;52:169–91.
10. Hu J, Li H, Waterman M, Zhou XJ. Integrative missing value estimation for microarray data. *BMC Bioinformatics*. 2006;7:449–62.
11. Tuikkala J, Elo L, Nevalainen OS, Aittokallio T. Improving missing value estimation in microarray data with gene ontology. *Bioinformatics*. 2006; 22:566–72.
12. Xiang Q, Dai X, Deng Y, He C, Wang J, Feng J, Dai Z. Missing value imputation for microarray gene expression data using histone acetylation information. *BMC Bioinformatics*. 2008;9:252–68.
13. Friedland S, Niknejad A, Kaveh M, Zare H. An Algorithm for Missing Value Estimation for DNA Microarray Data. *Preprint*. 2005.
14. Keller AD, Schummer M, Hood L, Ruzzo WL. Bayesian Classification of DNA Array Expression Data. *Technical Report UW*. 2000.
15. Shang C, Shen Q. Aiding Classification of Gene Expression Data with Feature Selection: A Comparative Study. *International Journal of Computational Intelligence Research*. 2005:1.
16. Ben-Dor A, Bruhn L, Friedman N, Nachman I, Schummer M, Yakhini Z. Tissue Classification with Gene Expression Profiles. *Proceedings of the 4th Annual International Conference on Computational Molecular Biology*. 2000:54–64.
17. Livingston G, Li X, Li G, Hao L, Zhou J. Analyzing Gene Expression Data Using Classification Rules. *bioML workshop*. 2003.
18. Tan AC, Gilbert D. Ensemble machine learning on gene expression data for cancer classification. *Applied Bioinformatics*. 2003;2:S75–83.
19. Breiman L. Bagging predictors. *Machine Learning*. 1996;24:123–40.
20. Freund Y, Schapire RE. Experiments with a new boosting algorithm. *Proceedings of the Thirteenth International Conference on Machine Learning*. 1996:148–56.
21. Machado LO, Vinterbo S, Weber G. Classification of gene expression data using fuzzy logic. *Journal of Intelligent and Fuzzy Systems*. 2002; 12:19–24.
22. Chen Y, Dougherty ER, Bittner ML. Ratio-based decisions and the quantitative analysis of CDNA microarray images. *Journal of Biomedical Optics*. 1997;2:364–74.
23. Newton MA, Kendziorski CM, Richmond CS, Blattner FR, Tsui KW. On differential variability of expression ratios: Improving statistical inference about gene expression changes from microarray data. *Journal of Computational Biology*. 1999;8:37–52.
24. Smyth GK, Yang YH, Speed T. Statistical issues in microarray data analysis. *Functional Genomics: Methods and Protocols*. 2003.
25. Xiao Y, Frisina R, Gordon A, Klebanov L, Yakovlev A. Multivariate search for differentially expressed gene combinations. *BMC Bioinformatics*. 2004; 5:164–73.
26. Tusher VG, Tibshirani R, Chu G. Significance analysis of microarrays applied to the ionizing radiation response. *Proceedings of the National Academy of Sciences of the United States of America*. 2001;98:5116–21.

27. Pan W. A comparative review of statistical methods for discovering differentially expressed genes in replicated microarray experiments. *Bioinformatics*. 2002;18:546–54.
28. Gusnanto A, Ploner A, Pawitan Y. Fold-change estimation of differentially expressed genes using mixture mixed model. *Statistical Applications in Genetics and Molecular Biology*. 2005:4.
29. Jia Z, Xu S. Bayesian Mixture Model Analysis for Detecting Differentially Expressed Genes. *International Journal of Plant Genomics*. 2008.
30. Jiang D, Tang C, Zhang A. Cluster analysis for gene expression data: a survey. *IEEE Trans on Knowl and Data Eng*. 2004;16:1370–86.
31. MacQueen J. Some methods for classification and analysis of multivariate observations. *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*. 1967;1:281–97.
32. Rose K, Gurewitz E, Fox G. Statistical Mechanics and Phase Transitions in Clustering. *Phys Rev Lett*. 1990;65:945–8.
33. Kohonen T. Self-Organization and Associative Memory. Spring-Verlag; 1984.
34. Xiao X, Dow ER, Eberhart R, Miled ZB, Oppelt RJ. Gene Clustering Using Self-Organizing Maps and Particle Swarm Optimization. *Proc of the 17th Int. Symposium on Parallel and Distributed Processing*. 2003;154:2.
35. Jiang D, Pei J, Zhang A. DHC: A Density-based Hierarchical Clustering Method for Time Series Gene Expression Data. *Proc of the 3rd IEEE Int. Symposium on Bioinformatics and Bioengineering*. 2003:393.
36. Kaufman L, Rousseeuw PJ. Fitting Groups in Data: An Introduction to Cluster Analysis. New York: Wiley; 2005.
37. Barash B, Friedman N. Context-specific Bayesian clustering for gene expression data. *J Comp Bio*. 2002;9:169–91.
38. Shamir R, Sharan R. Click: A clustering algorithm for gene expression analysis. *Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology*. 2000:307–16.
39. Ben-Dor A, Shamir R, Yakhini Z. Clustering gene expression patterns. *Journal of Computational Biology*. 1999;6:281–97.
40. Xu Y, Olman V, Xu D. Minimum Spanning Trees for Gene Expression Data Clustering. *Genome Informatics*. 2001;12:24–33.
41. Seno S, Teramoto R, Takenaka Y, Matsuda H. A Method for Clustering Gene Expression Data Based on Graph Structure. *Genome Informatics*. 2004;15: 151–60.
42. Diday, Simon. Clustering Analysis. Digital Pattern Recognition. Springer-Verlag; 1980.
43. Fraley C, Raftery AE. How Many Clusters? Which Clustering Method? Answers Via Model-Based Cluster Analysis. *The Computer Journal*. 1998; 41:578–88.
44. Getz G, Levine E, Domany E. Coupled two-way clustering analysis of gene microarray data. *Proc Natl Acad Sci*. 2000;97:12019–84.
45. Lazzeroni L, Owen A. Plaid models for gene expression data. *Statistical Sinica*. 2002;12:61–89.
46. Cheng Y, Church GM. Biclustering of expression data. *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*. 2000;8:93–103.
47. Zhao L, Zaki MJ. Tricluster: An Effective Algorithm for Mining Coherent Clusters in 3D Microarray Data. *Proc of the 2005 ACM SIGMOD Int. Conference on Management of Data*. 2005:694–705.
48. Akutsu T, Kuhara S, Maruyama O, Miyano S. A system for identifying genetic networks from gene expression patterns produced by gene disruptions and overexpressions. *Genome Inform ser Workshop Genome Inform*. 1998;9:151–60.
49. Akutsu T, Miyano S, Kuhara S. Identification of genetic networks from a small number of gene expression patterns under the boolean network model. *In Pacific Symposium on Biocomputing*. 1999;4:17–28.
50. Lewin B. Genes VII. Oxford: Oxford University Press; 1999.
51. Shmulevich I, Dougherty ER, Kim S, Zhang W. Probabilistic boolean networks: a rule-based uncertainty model for gene regulatory networks. *Bioinformatics*. 2002;18:261–74.
52. Faryabi B, Datta A, Dougherty ER. On Approximate Stochastic Control in Genetic Regulatory Networks. *IET SYST BIOL*. 2007;1:361–8.
53. Kim S, Li H, Dougherty ER, Cao N, Chen Y, Bittner M, Suh E. Can Markov chain models mimic biological regulation. *Journal of Biological Systems*. 2002;10:337–57.
54. Dougherty ER, Pal R, Qian XN, Bittner ML, Datta A. Stationary and structural control in gene regulatory networks: basic concepts. *International Journal of Systems Science*. 2010;41:5–16.
55. Weaver DC, Workman CT, Stormo GD. Modeling regulatory networks with weight matrices. *Pacific Symposium on Biocomputing*. 1999;4:112–23.
56. Chen L, Aihara K. Stability of Genetic Regulatory Networks With Time Delay. *IEEE Transactions on Circuits and Systems*. 2002;49:602–8.
57. El-Samad H, Khammash M. Modelling and analysis of gene regulatory network using feedback control theory. *International Journal of Systems Science*. 2010;41:17–33.
58. Goss PJE, Peccoud J. Quantitative modeling of stochastic systems in molecular biology by using stochastic Petri nets. *Proc Natl Acad Sci U S A*. 1998;95:6750–5.
59. Tian T, Burrage K. Stochastic Neural Network Models for Gene Regulatory Networks. *The 2003 Congress on Evolutionary Computation*. 2003:162–9.
60. Koza JR, Mydlowec W, Lanzo G, Yu J, Keane M. Reverse engineering of metabolic pathways from observed data using genetic programming. *Pacific Symposium on Biocomputing*. 2001;434–45.
61. Dobra A, Hans C, Jones B, Nevins J, West M. Sparse Graphical Models for Exploring Gene Expression Data. *Journal of Multivariate Analysis*. 2004;90:196–212.
62. Hartemink AJ, Gifford DJ, Jaakkola T, Young RA. Combining Location and Expression Data for Principled Discovery of Genetic Regulatory Network Models. *Pacific Symposium ON Biocomputing*. 2002:437–49.
63. Zhang W, Shmulevich I. Computational and Statistical approach to Genomics. Springer-Verlag; 2006.
64. Datta A, Choudhary A, Bittner ML, Dougherty ER. External Control in Markovian Genetic Regulatory Networks. *Machine Learning*. 2003;52:169–91.
65. Pal R, Datta A, Dougherty ER. Optimal Infinite-Horizon Control for Probabilistic Boolean Networks. *IEEE Transaction Signal Process*. 2006;54: 2375–87.
66. Abul O. Controlling Discrete Genetic Regulatory Networks. PHD THESIS. 2005.