Artificial Intelligence in Medicine (2005) **35**, 49—60

ARTIFICIAL INTELLIGENCE IN MEDICINE

http://www.intl.elsevierhealth.com/journals/aiim

SiMCAL 1 algorithm for analysis of gene expression data related to the phosphatidylserine receptor

Daniel Dvorkin [a,b,*], **Valerie Fadok** [c], **Krzysztof Cios** [a,d,e]

[a] *University of Colorado at Denver and Health Sciences Center, Denver, CO 80217, USA*
[b] *University of Minnesota School of Public Health, Minneapolis, MN 55455, USA*
[c] *National Jewish Medical and Research Center, Denver, CO 80206, USA*
[d] *University of Colarado at Boulder, CO 80309, USA*
[e] *4C Data LLC, Golden, CO 80401, USA*

Received 11 October 2004; received in revised form 9 January 2005; accepted 12 January 2005

KEYWORDS
Clustering;
Jarris-Patrick;
Inflammation;
Microarray;
Nearest neighbor;
Nonparametric;
Phosphatidylserine;
PSR;
SiMCAL;
TGF-β

Summary

Objective: SiMCAL 1 (simple multilevel clustering and linking, version 1) is a novel clustering algorithm for time-series microarray data, presented here with an application to a specific data set. The purpose of the algorithm is to present a complete feature set not found in either Jarvis-Patrick clustering, from which it is derived, or in other popular clustering methods such as hierarchical and *k*-means. The data concern the activity of the phosphatidylserine receptor (PSR) which is believed to be a crucial molecular switch in the mediation of inflammatory response in apoptosis and lysis. By analyzing the behavior of PSR-related genes in mouse macrophages, we hope to elucidate the mechanisms involved in this important biological process.
Methods and materials: SiMCAL 1 is implemented in the Python programming language using the Numerical Python extensions, and the data are stored using the MySQL database management system. The data are derived from exposures of multiple Affymetrix mouse gene microarray chips to elevated levels of PSR antibody and control conditions. Code and data are available at http://www.dvorkin.com/daniel/Simcal1.zip (accessed: 17 January 2005).
Results: The algorithm meets its objectives: it is *simple*, in that it is computationally inexpensive; it is *multilevel*, in that it provides a small number of clearly defined hierarchical levels of clusters; and it offers *linking* between clusters at the same level in each hierarchy. Clustering and linking results indicate previously unknown co-regulation for genes expressing PGH synthase (COX2) and PGE2, appear to confirm increased production of proteins for clearance of apoptotic cells in the presence of PSR antibody, and correspond to other findings regarding the temporal relationship between PGE2 production and B cell proliferation and differentiation. These results are promising but should be taken as highly preliminary.

* Corresponding author at: 1138 E. 14th Ave. Unit 8, Denver, CO 80218, USA. Tel.: +1 303 548 4568; fax: +1 303 607 9430.
E-mail address: daniel.dvorkin@gmail.com (D. Dvorkin).

0933-3657/$ — see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.artmed.2005.01.010

Conclusion: Both the algorithm and its application to this problem show great potential for future development. We plan to improve and extend the SiMCAL family of algorithms, and to obtain new data so that the algorithm(s) may be further applied to this and other problems of interest.
© 2005 Elsevier B.V. All rights reserved.

1. Introduction

The introduction is divided into biological and computational background sections. Although the computational aspect depends on the biological aspect, each may be considered in some degree of isolation: the biology is an area of ongoing "wet-lab" research as well as computational analysis, while the computation is based on data mining techniques with a wide variety of applications, and which are not specific to the problem at hand.

1.1. Biological background

Normal cell death, or *apoptosis*, requires regular housekeeping activity within all multicellular organisms. In vertebrates, this activity is performed by macrophages, which must be able to distinguish between three classes of cells: healthy cells, which the macrophages must not consume; apoptotic cells, which must be consumed without triggering an inflammatory response; and cells undergoing *lysis* (death from some external cause such as injury or disease) that also must be consumed, but which should trigger an inflammatory response [1].

Since cells are constantly undergoing apoptosis throughout the bodies of all multicellular organisms, the mechanism for preventing inflammatory response must work perfectly or disease will result. In humans, serious diseases such as cystic fibrosis, lupus, and some forms of diabetes may be at least partly the result of a breakdown of this mechanism [2].

Phosphatidylserine(PS) is a phospholipid generally found on the cytosolic surface of the cell membrane in most animal cells. Apoptotic cells appear to shift PS rapidly to the exterior of the membrane, at least partly as a signal for phagocytosis. However, it is far from the only signal for this process. What makes PS interesting is the effect PS has on the macrophage. Henson et al. state that the phosphatidylserine receptor (PSR) is "a crucial molecular switch" in the control of immune response [3]. Specifically, when PS is expressed in apoptotic cells, it appears to reduce inflammation and other aspects of the immune response by engaging the PSR in macrophages. Cells undergoing lysis do not show this behavior, allowing the full range of immune response, including inflammation, to come into play.

1.2. Computational background

The overall purpose of the project is to analyze microarray data on genes which might be relevant to PSR activity. Below we discuss both the nature of the data and background information on the algorithms used in analyzing the data. Details on the specific methods used in the analysis are found in Section 2.

1.2.1. Data

To ensure that the data were accurate and replicable, three separate and identical experiments (replicants) were performed. In each, microarrays containing 12,488 mouse genes known to be active in mouse macrophages, with each gene being represented by between 5 and 30 RNA probes, were exposed to a single concentration of monoclonal PSR antibody over time. Fluorescent hybridization measurements were taken for RNA at 30 min, 2 h, and 8 h 30 min.

Identical measurements were taken of control samples which were treated with an isotype, i.e., another IgM antibody with better-known effects. Another control sample, of which only one measurement was taken, was not treated. Therefore, seven distinct data points were generated for each probe. These steps were performed using Affymetrix ("Affy") microarray equipment and the Affy chip MG-U74Av2, a standard chip containing mouse genes.

1.2.2. SiMCAL versus existing methods

The origin of SiMCAL 1 lies in the Jarvis-Patrick (JP) algorithm [4], which has been widely used in fields ranging from astronomy to chemoinformatics, but has so far not been common in bioinformatics applications. JP begins with two user-defined parameters: k and $k_{\min}$. For each data element, a list of that element's k nearest neighbors is calculated. Then the algorithm states that two data elements are in the same cluster if either of the following conditions applies:

- They are within each other's lists of nearest neighbors.
- They have at least $k_{\min}$ nearest neighbors in common.

The advantages of JP are as follows. First, because it allows chains of relationships between data elements (e.g., a may be clustered with b, and b clustered with c, so that a and c end up in the same cluster even if they are not obviously related to each other) it allows non-convex clusters which may more accurately reflect patterns in the data. Second, it is largely non-parametric; it does not impose an arbitrary threshold on similarity between elements in order for them to be clustered together, but operates primarily by ranking, and is thus less sensitive to outliers than parametric methods. Third, and perhaps most important, it provides a method of determining the "natural" number of clusters within the data, rather than requiring a user to a priori select a number of clusters as most other clustering algorithms do.

However, its disadvantages are that the values of k and $k_{\min}$ are arbitrary and must be specified a priori. Also arbitrary is the definition of what degree of similarity constitutes "neighborness". (Determination of differential expression between genes is an area of active research; lack of differentiation as a standard of "neighborness" may be used in the future versions of the algorithm.) Also, it is computationally expensive; since it compares lists of neighbors, the algorithm is $O(kn^2)$. Finally, it provides only one level of resolution; in its basic form, it provides no way of building hierarchical levels of clusters which may contain each other.

The idea of SiMCAL is to address these shortcomings, as well as to provide additional functionality not found in any of the algorithms so far examined. We have designed "SiMCAL" to have the following attributes:

1. It must be *simple*, in that it has low computational complexity, preferably $O(n^2)$ or less, and/or relatively few computations are performed at each step of the inner loop. Subjectively, the user should feel that the algorithm is "fast" compared to other methods.
2. It must be *multilevel*, in that it provides a hierarchical relationship between clusters, but with fewer and better-defined levels than hierarchical clustering provides. Therefore, the user can, to use a laboratory metaphor, "switch the objective" to examine the results at a few well-defined levels of resolution, rather than choosing an arbitrary cut level as in standard hierarchical clustering.
3. It must provide *linking* between clusters, in that it yields information on the relationship between different clusters at the same level in the hierarchy. For time-series microarray data, this relationship is temporal. The user can then trace the relationship through multiple clusters by navigating a directed graph.

SiMCAL 1 fulfills these requirements. With regards to computational complexity, it is strictly $\Theta(n^2)$, in contrast to standard hierarchical clustering methods [5, pp. 379—382],which are $\Omega(n^2)$but approach $O(n^3)$ for some data sets, and usually perform at about $O(n^2 \lg n)$. Also, the inner loop calculations in SiMCAL 1 are much more lightweight than those in hierarchical clustering. Unlike the commonly used k-means method [5, pp. 382—384], however, SiMCAL 1 does establish a hierarchical relationship between clusters. Finally, it shares the advantage of JP in that it shows the "natural" number of clusters in the data at each level in the hierarchy, while being less expensive than JP and offering the unique advantage of providing information on the relationship between different clusters at the same level.

1.2.3. Notations and definitions

It is useful here to define some notations and terms used throughout the following sections. First, given two vectors $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ and $\mathbf{y} = (y_1, y_2, \ldots, y_n)$, and a scalar z; and for arbitrary functions $f(\mathbf{x})$ such as $\mathbf{x}^2$, $g(\mathbf{x}, \mathbf{y})$ such as $\mathbf{xy}$, or $h(\mathbf{x}, z)$ such as $\mathbf{x} + z$, let:

$$f(\mathbf{x}) = (f(x_1), f(x_2), \ldots, f(x_n)); \tag{1}$$

$$g(\mathbf{x}, \mathbf{y}) = (g(x_1, y_1), g(x_2, y_2), \ldots, g(x_n, y_n)); \tag{2}$$

$$h(\mathbf{x}) = (h(x_1, z), h(x_2, z), \ldots, h(x_n, z)). \tag{3}$$

Note that the *correlation coefficient*[6, pp. 218—220] of $\mathbf{x}$ and $\mathbf{y}$ is:

$$\rho_{\mathbf{x},\mathbf{y}} = \frac{\mathrm{Cov}(\mathbf{x}, \mathbf{y})}{\sigma_{\mathbf{x}}\sigma_{\mathbf{y}}}. \tag{4}$$

Let the *weighted mean* of $\mathbf{x}$ and $\mathbf{y}$ be the mean of $\mathbf{x}$ weighted by the values of $\mathbf{y}$. In other words, if $\mathbf{y} = (1, 2, 1)$, then in the calculation of the mean, x_2 has a weight equal to the combined weights of x_1 and x_3, which have weights equal to each other. The definition of the function depends on the dimension of its arguments. For $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ and $\mathbf{y} = (y_1, y_2, \ldots, y_n)$:

$$W(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^{n} x_i y_i}{\sum_{i=1}^{n} y_i}. \tag{5}$$

Also let $\max_0(\mathbf{x})$ be defined as:

$$\max_0(\mathbf{x}) = (\max(x_1, 0), \max(x_2, 0), \ldots, \max(x_n, 0)). \tag{6}$$

The vector and matrix definitions of ρ, W, and $\max_0$ follow naturally from these definitions and from Eqs. (1)—(3).

If matrix **X** is $m \times n$, then the *transpose* of **X**[7, p. 17] is $\mathbf{X}^T$ such that:

$$\mathbf{X}^T_{j,i} = \mathbf{X}_{i,j} \, \forall \, i \in [1, m], \, j \in [1, n] \tag{7}$$

2. Data analysis

2.1. Preprocessing

Preprocessing is of fundamental importance in any data mining operation [5, pp. 444—447]. The raw data, although interesting, do not yet contain usable information. The primary goals of preprocessing are to eliminate or reduce unreliable data, combine separate data points to take all measurements into account, and to normalize the data in such a way that clustering and linking will produce meaningful results.

2.1.1. Raw measurements and reliability

For each "raw" gene in each replicant, there exists a set of measurements, based on one measurement for the unexposed control, and time-series measurements for the isotype-exposed genes, *iso*, and the PSR-exposed genes, *psr*. In the time-series measurements, psr_t and iso_t denote the values measured at time point t. For the current data, $t = 1$, $t = 2$, and $t = 3$ indicate exposure times of 30 min, 2 h, and 8 h 30 min, respectively. Therefore, overall:

$$\text{measurements}_{\text{gene} \in \text{replicant}} = \{\text{control}\} \bigcup_{t=1}^{3} \{iso_t, \, psr_t\} \tag{8}$$

Each of these measurements has an associated p-value, and a lower p-value indicates higher reliability placed on the measurement [8]. A reliability can be calculated for any gene in a replicant based on the average p-value. First, let $\varepsilon = \log_{0.95} 0.05 \approx 58.4$, and then reliability is:

$$rel_{\text{gene} \in \text{replicant}} = \min_{t \in \{0,1,2,3\}} (1 - (p\text{-value}_{\text{gene} \in \text{replicant}}))^{\varepsilon}. \tag{9}$$

In other words, for any gene in any replicant, that gene's reliability is the *minimum* of the reliability for any measurement associated with it. One bad measurement in an otherwise good time-series could badly throw off all subsequent calculations; by this method, the algorithm is aware of such measurements, and will either discard them or assign them less weight in calculations, as described in following sections. Eq. (9) also ensures $rel \in [0, 1]$, providing a simple and consistent basis for calculation and interpretation.

The definition of ε deserves some discussion. Affy's own evaluation of reliability marks measurements having a p-value significantly above 0.05 as "absent,", those having a p-value around 0.05 as "marginal," and those having a p-value significantly less than 0.05 as "present." The definition of $\varepsilon = \log_{0.95} 0.05$ means that a "marginal" gene having a maximum p-value of 0.95 will have a reliability of 0.05, or "5%." This is the minimum reliability used in any subsequent calculation and leads to very low (but not zero) weighting for measurements which have a p-value only slightly less than or equal to 0.05. This choice is fairly arbitrary; it would be possible to let $\varepsilon = 1$, or, going to the opposite extreme, a very large value such as $\varepsilon = 1000$. In any case, $\varepsilon > 1$ serves the purpose of "concentrating" the more reliable values [5, 81], and the higher its magnitude, the more emphasis is given to those expressions with lower p-value.

2.1.2. Measurement combination and normalization

Before calculating normalized values, replicants with $rel < 0.05$ are discarded. This ensures that the analysis will proceed only with reliable data. Now, a consistent basis is required for comparing expression levels between genes for which reliable measurements exist. Unfortunately, there is no meaningful way to compare the raw expression levels to each other for different genes; it cannot be said that if one gene at one time point expresses at 5000 according to the Affy reading, and another at the same time point expresses at 10,000, that the second gene is expressing twice as much of the relevant protein as the first. However, it can be said that if a single gene expresses at 5000 at one time point and at 10,000 at the next, that its expression level has doubled [8].

Therefore, a measurement of the relative expression levels of the PSR-exposed data points relative to the isotype-exposed data points and the untreated control data point is defined in Eq. (10). Note that $\lg x \equiv \log_2 x$, that is, the base-2 logarithm of x.

$$\exp_t = \lg \frac{psr_t^2}{\text{control} \times iso_t}, \, t \in \{1, 2, 3\}. \tag{10}$$

Now let the expression level for the untreated control be $\exp_0 = 0$, and:

$$\mathbf{exp}_{\text{gene} \in \text{replicant}} = (\exp_0, \exp_1, \exp_2, \exp_3). \tag{11}$$

Each expression level has been calculated to take into account the change between the control expression level and the PSR-exposed expression level, and just as importantly, the changes between the PSR-exposed expression level and the isotype-exposed expression level. In other words, the normalized expression level should reflect the difference PSR specifically makes in expression, as well the difference PSR as an IgM antibody makes in expression. Consider two possible cases:

$$\frac{psr_t^2}{\text{control} \times iso_t} = \begin{cases} 9 & \text{if } (\text{control}, iso_t, psr_t) = (1000, 1000, 3000); \\ 3 & \text{if } (\text{control}, iso_t, psr_t) = (1000, 3000, 3000). \end{cases}$$

The use of base-2 log ratios has become standard in processing raw microarray expression values; some recent examples are to be found in [9—11]. The particular ratio used in Eq. (10) is original, and chosen, as shown in the example above, to allow us to differentiate between changes in expression induced specifically by PSR and those induced generally by IgM antibodies. In the first case, PSR specifically is clearly responsible for up-regulating the gene under examination, while in the second case it appears to be the presence of IgM antibody in general which is responsible for the up-regulation.

A logarithmic scale for normalized expression levels is used for two reasons, both relating to revealing small differences in the expression levels. First, the values of $(psr_t^2/(\text{control} \times iso_t))$ cover an enormous range, from roughly 0.00003—3500, and clearly any reporting mechanism which can cover such a range would lack the precision to distinguish between a subtle but important difference such as that between 3 and 9. Second, such a reporting mechanism would tend to overemphasize up-regulation and underemphasize down-regulation, since the magnitude of the values for up-regulated genes is so much greater.

Finally, 2 is used as the base of the logarithm, rather than some other common choice such as *e* or 10, so as not to "squeeze" the scale too much and make it difficult to report on changes at the extreme ends of the range. This choice also makes interpretation straightforward: a change of 1 in the reported value means a two-fold change in expression.

2.1.3. Gene expression calculation

At this point, reliability and expression values exist for each gene in each replicant. The reliability score is used as a weighting factor in the averaging. In addition to the expression vector exp and the reliability rel, it is also useful to calculate variability, var, which is an overall measure of how much the gene reacts to the experimental conditions. For each gene:

$$\mathbf{exp}_{\text{gene}} = W_{\text{replicants}}(\mathbf{exp}_{\text{gene} \in \text{replicant}}, rel_{\text{gene} \in \text{replicant}}); \tag{12}$$

$$rel_{\text{gene}} = \mu_{rel_{\text{gene} \in \text{replicant}}}; \tag{13}$$

$$var_{\text{gene}} = \frac{1}{3}\sum_{t=1}^{3} abs(\mathbf{exp}_{\text{gene}}). \tag{14}$$

Genes are assigned Id numbers, ordered from least variable (Id 0) to most variable (with our data, for all genes that made it through preprocessing, Id 5285). They may then be considered as an array, and individual genes denoted by their position in the array — for example, the gene with Id number 2201 is G_{2201}.

2.2. Clustering and linking

This section describes the distance measures used and how clusters are formed, associations are mapped, and genes and clusters are linked from this information.

2.2.1. Distance measures and matrices

The similarity measure, for any two genes, used throughout the clustering and linking calculations is the correlation coefficient ρ as defined in Eq. (4). This is a general measure of the degree of linear dependence between two vectors — i.e., the degree to which they tend to rise or fall together [6, pp. 218-220].

This measure, often referred to as the Pearson correlation coefficient, has become a canonical measure of similarity (or, in the inverse, difference) between gene expression patterns in a wide variety of applications; some of many recent examples include [10,12—14]. More complex time-series measures, such as Box-Jenkins, generally require a larger number of data points than those generated in this experiment, and also often have a requirement that the data be evenly spaced; in contrast, $\rho$is robust for multidimensional data of almost any dimensionality and interval.

Compared to other dependence measures, ρ has the useful quality that it is dimensionless: $\rho_{\mathbf{xy}} \in [-1, 1]$ for all vectors of equal length **x** and **y**. Interpretation is straightforward:

- $\rho_{\mathbf{xy}} = 0$ indicates that **x** and **y** are linearly independent.
- $\rho_{\mathbf{xy}} \to 1$ indicates increasing positive linear dependence; as **x** rises or falls, so does **y**. Similarly, $\rho_{\mathbf{xy}} \to -1$ indicates increasing negative linear dependence; as **x** rises, **y** falls, and vice versa.

- $|\rho_{\mathbf{xy}}| \geq 0.8$ indicates a strong linear dependence between **x** and **y**, $0.5 < |\rho_{\mathbf{xy}}| < 0.8$ indicates moderate dependence, and $0 < |\rho_{\mathbf{xy}}| \leq 0.5$ indicates weak dependence.

Why use a dependence measure as opposed to the common Minkowski metric such as the Euclidean? (Minkowski metrics are distance measures of the family $\Delta_{\mathbf{x}}^{\mathbf{y}} = \sqrt[p]{\sum_{i=1}^{n} |x_i - y_i|^p}$. Euclidean distance is the most common special case of Minkowski metrics: $\Delta_{\mathbf{x}}^{\mathbf{y}} = \sqrt{\sum_{i=1}^{n} |x_i - y_i|^2}$. [5, p. 377]) The answer is that clustering is based here on similarity in the *changes*in expression patterns under experimental conditions, and a dependence measure such as ρ provides a better measure of the relationship between different genes' up- or down-regulation over time than does a Minkowski metric.

Fig. 1, which represents possible values of $\exp_{\text{gene}}$, shows an example of such a situation. Clearly **y**and **z** are closer to each other by any Minkowski measure than either is to **x**; just as clearly, however, the responses of **x** and **y** under experimental conditions are more similar, since both are steadily upregulated while **z** is downregulated. Clustering based on ρ accurately reflects this.

The use of ρ is the reason that $\exp_0 = 0$. Recall that each gene is starting from the "natural" (untreated) state, and changes in expression level are measured from this baseline. If the expression vector were simply $\exp_{\text{gene}} = (\exp_1, \exp_2, \exp_3)$, this could lead to false correlations.

Suppose $\mathbf{x} = (-1, -3, -2)$ and $\mathbf{y} = (2, 0, 1)$. (See Fig. 2.) Without $\exp_0 = 0$, $\rho_{\mathbf{x},\mathbf{y}} = 1$, which is clearly inaccurate, since in fact **x**is downregulated under experimental conditions while **y** is upregulated. Setting $\mathbf{x} = (0, -1, -3, -2)$ and $\mathbf{y} = (0, 2, 0, 1)$ gives a much more accurate value of $\rho_{\mathbf{x},\mathbf{y}} = 0.13$.

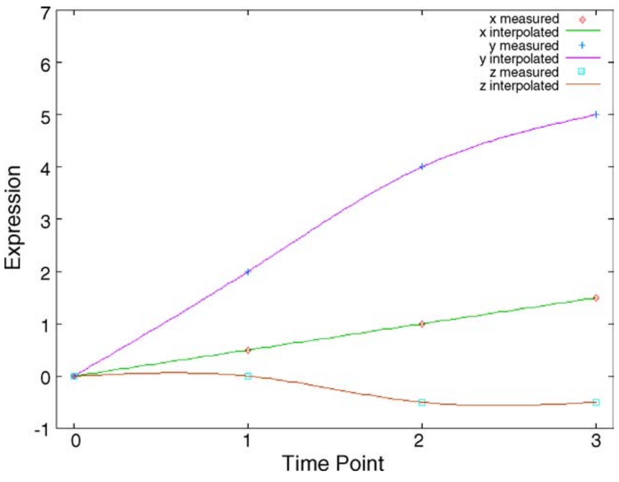

Figure 1 Expression levels of **x**, **y**, and **z**.

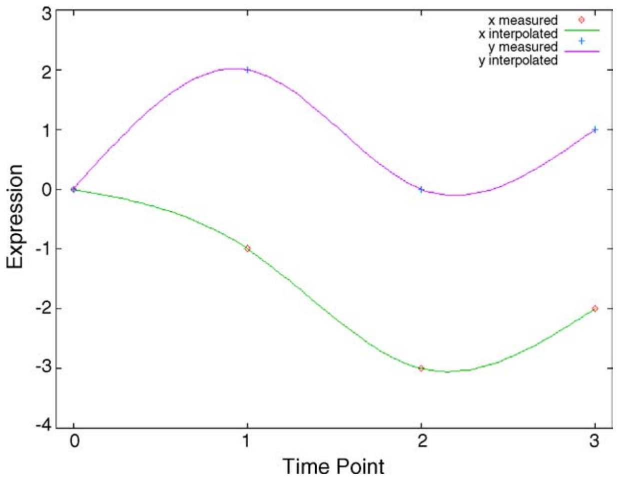

Figure 2 Expression levels of **x** and **y**.

Therefore, the initial step in clustering is to form a correlation matrix for all genes under consideration. This is the most complex step in the entire algorithm, but is strictly $O(n^2)$, and is still computationally inexpensive because calculating ρ is quite fast. Thus SiMCAL 1 as a whole is $O(n^2)$, and the first SiMCAL condition is fulfilled.

For the n genes $g \in [1, n]$ and $\mathbf{exp}_g = (\exp_{g,0}, \exp_{g,1} \exp_{g,2}, \exp_{g,3})$, let $\mathbf{exp}_g^{\text{early}} = (\exp_{g,0}, \exp_{g,1} \exp_{g,2})$ and $\mathbf{exp}_g^{\text{late}} = (\exp_{g,1}, \exp_{g,2} \exp_{g,3})$. Thus there are three expression matrices:

$$\begin{aligned} &\mathbf{Exp} = (\mathbf{exp}_1, \mathbf{exp}_2, \ldots, \mathbf{exp}_n); \\ &\mathbf{Exp}^{\text{early}} = (\mathbf{exp}_1^{\text{early}}, \mathbf{exp}_2^{\text{early}}, \ldots, \mathbf{exp}_n^{\text{early}}); \qquad (15) \\ &\mathbf{Exp}^{\text{late}} = (\mathbf{exp}_1^{\text{late}}, \mathbf{exp}_2^{\text{late}}, \ldots, \mathbf{exp}_n^{\text{late}}). \end{aligned}$$

These expression matrices are used to obtain the similarity matrices which we will use in clustering. First we obtain the "raw" matrices:

$$\begin{aligned} &\mathbf{Neighbors} = \max_0(\rho_{\mathbf{Exp},\mathbf{Exp}}); \\ &\mathbf{Leaders} = \rho_{\mathbf{Exp}^{\text{late}},\mathbf{Exp}^{\text{early}}}; \qquad (16) \\ &\mathbf{Followers} = \mathbf{Leaders}^T. \end{aligned}$$

The derivation of **Neighbors** is obvious. **Leaders** and **Followers** are somewhat deceptively named, but once they are used to obtain ranks based on the values of their elements (Section 2.2.2) they do indeed provide a measurement of the degree to which $\mathbf{Exp}^{\text{early}}$predicts $\mathbf{Exp}^{\text{late}}$. In other words, the maximum value in the i th row of **Leaders** is the index of the gene which best predicts gene i: its "Leader". Similarly, the maximum value in the i th row of **Followers** is the index of the gene which gene i best predicts: its "Follower".

However, the matrices in their current form are not adequate for clustering to proceed, and must be adjusted. First, we subtract **Neighbors**, used in clustering, from both **Leaders** and **Followers**, used in linking:

$$\begin{aligned}\mathbf{Leaders} &= \mathbf{Leaders} - \mathbf{Neighbors};\\ \mathbf{Followers} &= \mathbf{Followers} - \mathbf{Neighbors}.\end{aligned} \tag{17}$$

We perform this step to prevent false leader–follower ("succession") associations between genes whose expression levels track each other very closely under experimental conditions. Such genes should be clustered together using **Neighbors**, but they should not be linked using **Leaders** or **Followers**.

To illustrate this, suppose that for two genes $\mathbf{x}$ and $\mathbf{y}$, and some constant $c > 0$, $\mathbf{y} \approx c\mathbf{x}$. In other words, $\mathbf{x}$ and $\mathbf{y}$ approach complete positive linear dependence. Then "$\mathbf{x}$ leads $\mathbf{y}$" and "$\mathbf{y}$ leads $\mathbf{x}$" would be equally accurate statements. Since the purpose of the succession analysis is the elucidation of biological pathways, based on the assumption that a genes' leaders precede it in these pathways and its followers succeed it (Section 2.2.3) this would not be very helpful. Thus the subtraction step, which ensures that the more likely two genes are to be clustered, the less likely they are to be linked.

Finally, we wish to ensure that genes are not clustered or linked with themselves, by reducing their reported associations to below $\min(\rho)$. To accomplish this, we first define the matrix $\mathbf{Mask} = 2I_n$, where n is the number of genes:

$$\mathbf{Mask}_{i,j} = \begin{cases} 2 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \forall i, j \in [1, n]. \tag{18}$$

Then we subtract **Mask** from all three of our similarity matrices:

$$\begin{aligned}\mathbf{Neighbors} &= \mathbf{Neighbors} - \mathbf{Mask};\\ \mathbf{Leaders} &= \mathbf{Leaders} - \mathbf{Mask};\\ \mathbf{Followers} &= \mathbf{Followers} - \mathbf{Mask}.\end{aligned} \tag{19}$$

2.2.2. Ranks and cluster formation

The correlation matrices are now used to obtain rankings of association. Define the *rank* of a gene's relationship to another gene as its ranking in the appropriate correlation matrix; that is, rank 1 corresponds to the index of the largest argument in a given row of the correlation matrix, rank 2 corresponds to the index of the second largest argument, and so on.

Let $j = \text{argmax}(\mathbf{Neighbors}_i)$. Then gene $\mathbf{x}_j$ is the rank 1 neighbor of $\mathbf{x}_i$. (Note that $i \neq j$ because of the mask subtraction step.) The rank 2 neighbor of $\mathbf{x}_i$ is $\mathbf{x}_k$ where $k = \text{argmax}(\mathbf{Neighbors}_i)|k \neq j$; the rank 3 neighbor of $\mathbf{x}_i$ is $\mathbf{x}_l$ where $l = \text{argmax}(\mathbf{Neighbors}_i)$ $|l \neq j \wedge l \neq k$; etc. The calculations of leader and follower ranks proceed in similar fashion.

As implemented, this calculation is efficiently performed on all three similarity matrices (**Neighbors**, **Leaders**, and **Followers**) in their entirety, in one step, using the Numerical Python function `argsort`, which produces a sorted list of indices of a matrix or vector, as shown below in Python sample code:

```
a = array([1.3, 3.0, 2.9, 4.1])
b = argsort(a)
#b = array([0, 2, 1, 3])
```

In this example, `a[0]` is the rank 1 member of `a`, `a[2]` is the rank 2 member, `a[1]` is the rank 3 member, and `a[3]` is the rank 4 member.

The `iplr` function simply flips a matrix or vector from left to right, which makes it easier to extract rank information in the order the algorithm demands. Thus the code for finding ranked associations is:

```
Neighbors = fliplr(argsort(Neighbors))
Leaders = fliplr(argsort(Leaders))
Followers = fliplr(argsort(Followers))
```

Now clustering can proceed. Clusters are ranked just as neighbors, leaders, and followers are; a cluster of rank 1 contains all genes which have each other as neighbors of rank 1, and so on through higher ranks until only one "supercluster" exists. In other words, the condition for clustering is that for any rank $R > 0$, a gene is in the same cluster of rank R as other genes which it has as neighbors of rank $r \leq R$. Thus at rank 1, a gene is in the same cluster as its neighbors of rank 1 (and those genes' neighbors of rank 1, and so on); at rank 2, it is in the same cluster as its neighbors of rank 1 or 2, etc.

This goal is achieved by starting, at rank 1, with each gene in its own "rank 0" cluster, then finding its neighbor of rank 1, and placing that gene, along with any other genes which may be in its cluster, in the same cluster as the first gene. As the algorithm proceeds, of course, the clusters grow rapidly. Note that there is no requirement that genes be directly related to each other to be placed in the same cluster — this fulfills the requirement, inherited from JP, that the algorithm should be able to produce non-convex clusters. Instead, the algorithm navigates along chains of close association through the data. If **x** has **y** as its nearest neighbor, and **y** has **z** as its nearest neighbor, then **x** and **z** will end up in the same cluster of rank 1.

Clusters of higher rank are produced by reusing existing data structures. At rank 2, the genes are already clustered with their neighbors of rank 1, so there is no need to repeat the entire process with each gene's entire list of neighbors. Instead, the existing rank 1 clusters are agglomerated into rank 2 clusters, and so on through higher ranks. This process allows a hierarchy of clusters of different ranks, expressed as *parent* and *child* relationships. Let A and B be clusters of rank r, and C be a cluster of rank $r+1$. If $\mathbf{x} \in A$, $\mathbf{y} \in B$, and $\mathbf{x}, \mathbf{y} \in C$, then C is the parent of A and B, and A and B are children of C.

The following Python code from the clustering control file shows the core of the clustering procedure, at any rank:

set equal to 5. Determining the proper value of `MAX _RANK` is a sort of preprocessing step, based on early analysis of the data in question, and of course the value may be changed for new data.

Cluster centers are calculated, although they exist only for reporting and visualization purposes (Fig. 3) rather than any further algorithmic uses. Like a gene, a cluster has the expression vector $\mathbf{exp}_{\text{cluster}} = (\exp_0, \exp_1, \exp_2, \exp_3)$. For a given cluster C with m genes as members, let $\mathbf{G}$ be the matrix of the expression vectors of the genes in the cluster:
$\mathbf{G} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m) | \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m \in C.$

Also let **rel** be the vector of the reliabilities of those genes:
$\mathbf{rel} = (rel_1, rel_2, \ldots, rel_m).$

Then the location of the cluster center is:
$$\mathbf{exp}_C = W(\mathbf{G}, \mathbf{rel}). \quad (20)$$

```
# cycle through genes
for GeneId in xrange(NumOfGenes):

    # get ID's into easy-to-use forms
    i = GeneId
    j = GeneNeighbors[rank-1, i]

    # cluster assignment
    if GeneClusterIndexes[j] != GeneClusterIndexes[i]:
        # merge higher-ID cluster with lower-ID cluster
        localGCI = array([GeneClusterIndexes[i],
                          GeneClusterIndexes[j]])
        live = min(localGCI)
        dead = max(localGCI)
        # update GeneClusterIndexes to reflect merge
        for id in ClusterGenes[dead]:
            GeneClusterIndexes[id] = GeneClusterIndexes[live]
        # merge old cluster into new
        ClusterGenes[live] += ClusterGenes[dead]
        # empty out old cluster
        ClusterGenes[dead] = []
    # else:  already in the same cluster
```

Clustering ends when the algorithm reaches a rank such that only one cluster exists. The algorithm could, of course, continue operations up to rank $n-1$ for n genes, but that would serve no useful purpose. In fact, to speed up overall operations and reduce the amount of data storage required, the constant `MAX _RANK` is defined; no neighbor, leader, or follower associations beyond this rank are calculated or stored. With the current data set, clustering terminates at rank 4, and `MAX _RANK` is (generously)

Like genes, clusters are given Id numbers, starting at 0 within each rank and going up to the last cluster in the rank. They are not ordered by variability or any other inherent feature, although because the clustering algorithm proceeds along the gene array as ordered by Id, higher-numbered clusters will generally have higher variability. The ranks, and the clusters within them, may be considered as a matrix, and clusters may be uniquely identified by their position within the matrix —

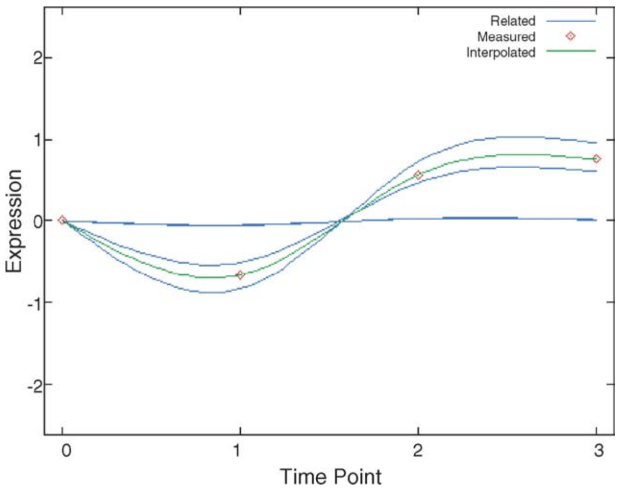

Figure 3 Expression plot for G_{4355} and its clusters.

for example, the cluster of rank 2 with Id 47 is $C_{2,47}$.

With regards to computational complexity, the creation of clusters at each rank is of linear complexity, or $O(n)$. Because $\mathtt{MAX\_RANK}$ is a constant, the entire clustering process is $O(\mathtt{MAXRANK}n) = O(n)$. If we consider $\mathtt{MAX\_RANK}$ to be a variable,

```
insert into ClusterSuccessionsIndividual
select QUERY.ClusterId as LeaderClusterId,
       ClusterGenes.ClusterId as FollowerClusterId,
       QUERY.Rank as Rank
from   GeneSuccessions, ClusterGenes
where  GeneSuccessions.LeaderGeneId = QUERY.GeneId and
       GeneSuccessions.Rank <= QUERY.Rank and
       ClusterGenes.Rank = QUERY.Rank and
       ClusterGenes.GeneId =
            GeneSuccessions.FollowerGeneId
```

which in a sense it is because it depends on the structure of the data, we may note that $\mathtt{MAXRANK} \ll n$, and so clustering still approaches $O(n)$ performance. Calculation of cluster centers is also $O(n)$. Therefore, the most complex step in the algorithm remains the calculation of the correlation matrices as described in Section 2.2.1, and the SiMCAL 1 algorithm is $O(n^2)$overall.

2.2.3. Linking

Linking proceeds in a similar fashion to clustering. Like neighbors, leaders and followers are ranked up to $\mathtt{MAX\_RANK}$. A gene **x** leads a gene **y** at rank *r* if **x** is the rank *r* leader of **y**; similarly, a gene **z** follows **y** if **z** is the rank *r* follower of **y**. This leads naturally to the concept of *successions*, the union of leaders and followers — in this example, **y** succeeds **x** and **z** succeeds **y**. Also, **y** is a successor of **x**, and so on.

Now let *A* and *B* be two clusters of rank *R*, and $\mathbf{x} \in A$ and $\mathbf{y} \in B$. Then *A* is a leader of *B*, and *B* is a follower of *A*, if *y* succeeds *x* at rank $r \leq R$. In other words, for one cluster to lead another (and for the second cluster to be a follower of the first cluster; the two cases are identical) it is sufficient either for a gene in the first cluster to be a leader (at a rank equal to or less than that of the cluster) of a gene in the second cluster, or for a gene in the second cluster to be a follower of a gene in the first cluster.

These operations, as implemented, are performed via SQL queries on the stored data for cluster membership and gene successions. First, for each gene in each cluster of each rank:

This step creates a rather overpopulated set of successions with many duplicates because a large number of genes are both leaders and followers — i.e., both of the conditions for succession above are met. Therefore, another SQL query provides the set of successions with no duplicate entries:

```
insert into ClusterSuccessions
select LeaderClusterId,
       FollowerClusterId,
       Rank,
       count(*) as Prediction
from   ClusterSuccessionsIndividual
group  by LeaderClusterId, FollowerClusterId, Rank
order  by LeaderClusterId, FollowerClusterId, Rank
```

The `Prediction` variable, which may be of interest for reporting purposes, is a count of duplicates in the `ClusterSuccessionsIndividual` table.

3. Results

3.1. Computational results

Since clusters at higher ranks may be very large, containing numerous genes, some have many leaders and followers. This is desirable for the possible elucidation of biochemical pathways, since such pathways generally have many branch points. Representing the set of clusters as a directed graph allows navigation along many leader—follower paths through the graph, and comparison of those paths with biochemical pathways of interest.

Analysis of the linking yields some unexpected results. For instance, at rank 3 (the highest rank with more than one cluster) there is a single very large cluster (Cluster Rank 3, Id 0, or $C_{3,0}$) containing thousands of genes which showed little change in expression under experimental conditions. This is not surprising, as the majority of genes on the Affy mouse chip are presumably not involved in inflammatory response or other PSR-mediated activity.

What is surprising is that the majority of the other, much smaller clusters at rank 3 have $C_{3,0}$ as their sole leader and follower. This may imply that the rank 3 clusters are too large to be useful in analysis, and that the rank 1 and 2 clusters are more useful when trying to determine the biological meaning of the clustering results. On the other hand, it may have some biological significance — perhaps implying that PSR-mediated reactions generally follow short pathways which involve few genes at a time.

3.2. Biological results

A major challenge in biological analysis of the clustering results is in the quality of the data. The data used in this phase of the project are relatively sparse, i.e. of low dimension, with only three time points, and, even with three replicants, have proven to have an unfortunately large number of measurements that were not deemed reliable enough to make it through preprocessing. As a practical matter, this means that many genes of interest were not included in the final data set.

Furthermore, the time course may have been too short to detect substantial associations between changes in gene activity relating to apoptosis and inflammatory response. Most studies in this area deal with time courses of 24 h or longer. However, even with the existing limitations, some promising congruences with evidence from other sources did emerge.

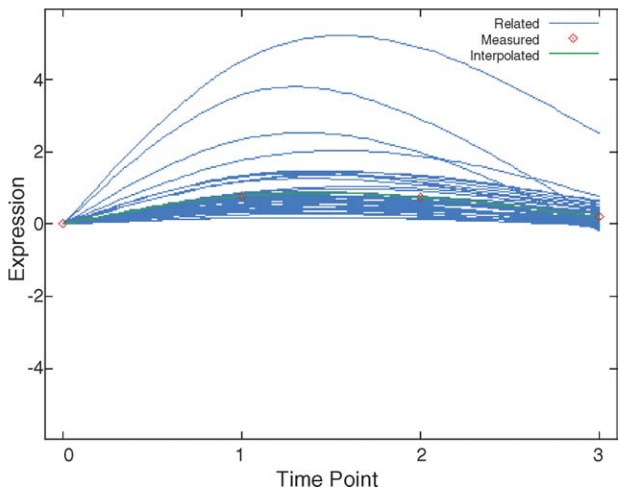

Figure 4 $C_{2,84}$ and its constituent genes.

$C_{2,84}$ contains two genes which both hold significant interest and have Affy annotations which provide useful information about their function: G_{5285} and G_{5264}. Both genes are highly upregulated under experimental conditions, and both show (as does $C_{2,84}$) a pattern of sharp up-regulation early in the experiment followed by decline. In Fig. 4, $G_{5285}$and G_{5264} are represented by the top and second-to-top blue lines, respectively. $C_{2,84}$ is surrounded by other genes toward the middle of the plot.

G_{5285}, which is known to be stimulated by PSR, is a gene for production of the protein prostaglandin (PG) G/H synthase, or PGH synthase, which is also known as COX2. This is a key enzyme in prostaglandin production. G_{5264} is a gene for the PG receptor PGE2, and was not previously known to be stimulated by PSR. The fact that these genes show the same expression pattern may be an indicator that production of the PG receptor proceeds at the same time as the production of PG precursors, rather than following such production at a later time.

Another cluster of interest is $C_{2,29}$, which shows consistently high levels of expression throughout the experiment, with steadily increasing up-regulation. Among its genes are G_{5100}, described by Affy as a "scavenger receptor," and G_{4974}, described as a "house-keeping protein." The presence of these genes in the same cluster, with consistently high expression on exposure to high concentrations of PSR, may indicate that they play a role in PSR mediation of immune response to apoptosis. Particularly interesting in this case is that $C_{2,29}$ is a follower of $C_{2,84}$.

$C_{2,29}$ and other followers of $C_{2,84}$ also contain several genes related to B cell production and activity: specifically, $G_{1004} \in C_{2,29}$, $G_{4544} \in C_{2,176}$, and $G_{3061} \in C_{2,229}$. He et al. [15] found that "in the presence of resistant T cells, PGE2-increased B cell proliferation and differentiation." Resistant T cells are defined here as cells which can increase immunoglobulin, or Ig, production; PSR antibody, of course, is an IgM antibody, and almost all B cells have IgM receptors. This is a good sign that the linking phase of the analysis correctly tracks the progression from PGE2 production to B cell proliferation.

4. Conclusions and future directions

The algorithm presented here can be extended to use the *uncentered correlation coefficient*, or ϱ, as a similarity measure rather than the current ρ. This is an extremely simple quantity to calculate, as it is equal to the cosine of the angle between two data vectors. It is particularly suitable for the type of data used here, in which there is a "zero reference state" from which all vectors are assumed to originate [16, pp. 5-6]: in this case, the untreated control expression level, $\exp_0$. Using this measure would reduce or eliminate the need for the somewhat artificial $\exp_0$data point. Other types of similarity measures may be added as time allows.

As mentioned above, determination of differential gene expression, i.e. deciding whether or not the expression patterns of any two genes in a microarray are actually "different" in a statistically meaningful sense, is an active area of mathematical and biological research, and several methods exist [17]. The lack of differentiation might meaningfully be considered a standard of "neighborness" in the JP sense and be superior to an arbitrary cutoff distance for JP variants. We may also consider entirely different approaches such as evolutionary search through the data space, biclustering, etc.

The number of time-series measurements used here is unfortunately small; although these particular time points were chosen deliberately based on the current understanding of the biological activity of PSR, a denser and more evenly spaced time-series — perhaps one measurement at 30 min, followed by hourly measurements over the following 8, 9, or 10 h — would be desirable for future study. It is worth noting that other experiments have had good results with small numbers of time-series microarray measurements; an interesting example is [10], in which Hashimoto et al. compared expression levels of genes involved in spinal cord injury healing at 1, 3, and 7 days. However, more data can always provide more information, and it is to be hoped that we will obtain higher dimensional data on PSR activity in the future.

Ideally, the analysis outlined in this paper should form a feedback cycle with the biological experimentation: analytical results are verified by experimental results, then examined to suggest new experiments, the results of which are then analyzed, and so on. As previously stated, the ultimate purpose of SiMCAL analysis of microarray data is to elucidate previously unknown biochemical pathways.

Acknowledgements

Drs. Peter Henson and Jenifer Monks, both of National Jewish Medical and Research Center, have given generously of their time and expertise at critical points in development and analysis. Both of these individuals deserve our profound thanks and gratitude.

References

[1] McDonald P, Fadok V, Bratton D, Henson P. Transcriptional and translational regulation of inflammatory mediator production by endogenous TGF- β in macrophages that have ingested apoptotic cells. J Immunol 1999;163:6164—72.
[2] Vandivier R, Fadok V, Hoffman P, Bratton D, Penvari C, Brown K, Brain J, Accurso F, Henson P. Elastase-mediated phosphatidylserine receptor cleavage impairs apoptotic cell clearance in cystic fibrosis and bronchiectasis. J Clin Invest 2002;109:661—70.
[3] Henson P, Bratton D, Fadok V. The phosphatidylserine receptor: a crucial molecular switch. Nat Rev Mol Cell Biol 2001; 2:627—33.
[4] Jarvis RA, Patrick EA. Clustering using a similarity measure based on shared near neighbors.. IEEE Trans Comput 1973;C22(11):1025—34.
[5] Cios K, Pedrycz W, Swiniarski R. Data mining methods for knowledge discovery. Norwell, MA: Kluwer Academic Publishers, 1998.
[6] Devore JL. Probability and statistics for scientists and engineers. Pacific Grove, CA: Duxbury, 2000.
[7] Friedberg SH, Insel AJ, Spence LE. Linear algebra, 3rd ed., Upper Saddle River, NJ: Prentice Hall, 1997.
[8] Affymetrix Inc., Statistical algorithms description document, http://www.affymetrix.com/support/technical/whitepapers/sadd_whitepaper.pdf (accessed: 17 January 2005), 2002.
[9] Gwinn MR, Whipkey DL, Weston A. The effect of oxythioquinox exposure on normal human mammary epithelial cell gene expression: a microarray analysis study. Environmental Health; 2004 (E-publication ahead of print).
[10] Hashimoto M, Koda M, Ino H, Yoshinaga K, Murata A, Yamazaki M, Kojima K, Chiba K, Mori C, Moriya H. Gene expression profiling of cathepsin D, metallothioneins-1 and -2, osteopontin, and tenascin-C in a mouse spinal cord injury model

by cDNA microarray analysis. Acta Neuropathol; 2004 (E-publication ahead of print).

[11] Nguyen DH, Toshida H, Schurr D, Beuerman RW. Microarray analysis of the rat lacrimal gland following the loss of parasympathetic control of secretion. Physiol Genomics 2004;18(1):108—18.

[12] Kim H, Golub GH, Park H. Missing value estimation for DNA microarray gene expression data: local least squares imputation. Bioinformatics; 2004 (E-publication ahead of print).

[13] Taguchi YH, Oono Y, Relational patterns of gene expression via nonmetric multidimensional scaling analysis. Bioinformatics; 2004 (E-publication ahead of print).

[14] Zorn KK, Jazaeri AA, Awtrey CS, Gardner GJ, Mok SC, Boyd J, Birrer MJ. Choice of normal ovarian control influences determination of differentially expressed genes in ovarian cancer expression profiling studies. Clin Cancer Res 2003;9: 4811—8.

[15] He X, Weyand CM, Goronzy JJ, Zhong W, Stuart JM. Bi-directional modulation of T cell-dependent antibody production by prostaglandin E_2. Int Immunol 2002;14(1): 69—77.

[16] deHoon M, Imoto S, Miyano S. The C Clustering Library, 2002, http://bonsai.ims.u-tokyo.ac.jp/mdehoon/software/cluster/cluster.pdf (accessed: 17 January 2005).

[17] Pan W. A comparative review of statistical methods for discovering differentially expressed genes in replicated microarray experiments. Bioinformatics 2002;18(4):546—54.