Gavin Sherlock
is Director of the Stanford Microarray Database at Stanford University, and also author of XCluster. His interests include the modelling and storage of microarray data, as well as its analysis.

Analysis of large-scale gene expression data

Gavin Sherlock

Date received (in revised form): 3rd October 2001

Abstract

DNA microarray technology has resulted in the generation of large complex data sets, such that the bottleneck in biological investigation has shifted from data generation, to data analysis. This review discusses some of the algorithms and tools for the analysis and organisation of microarray expression data, including clustering methods, partitioning methods, and methods for correlating expression data to other biological data.

Keywords: *microarray, clustering, analysis*

INTRODUCTION

Microarray studies may generate millions of data points and such volumes of data are too large to analyse by simple sorting in spreadsheets, or plotting as graphs. For sense to be made of the data, systematic methods for its organisation are thus required, with a means of measuring quantitatively if two expression profiles are similar to each other. In this regard it is useful to consider the values that make up the expression profile for a single gene/clone as a series of coordinates, which define a *gene expression vector*, with as many dimensions as there are data points within the expression profile. Using standard mathematical metrics of distance the similarity (or dissimilarity) between different vectors can be then measured. Both the Pearson correlation, which measures the similarity between the directions in which two vectors point, and the Euclidean distance, which measures the distance between two points in space, have been used.

NORMALISATION OF MICROARRAY DATA

This section on normalisation is mainly pertinent to two-channel microarray data, though many of the principles are applicable to large-scale expression data in general. There are many sources of systematic variation in microarray experiments that affect the measured gene expression levels. Normalisation is the term used to describe the process of removing such variation. These sources of systematic variation will affect different microarray experiments to different extents. Thus to compare data from different microarrays, we need to try to remove the systematic variation, to bring the data into register between the two arrays.

Such sources of systematic variation include:

- differences in labelling efficiency between the two dyes;
- differences in the power of the two lasers;
- differing amounts of RNA labelled between the two channels;
- spatial biases in ratios across the surface of the microarray.

For example, consider the following trivial example. An RNA sample is taken, and exactly equal amounts are labelled with Cy3 and Cy5 dyes, but the Cy5 dye labels the sample twice as efficiently as the Cy3 dye. If everything else is equal, analysis of the scanned image will yield a ratio of 2 for every spot, instead of a correct ratio of 1, owing to the greater efficiency of labelling with the red dye.

Gavin Sherlock,
Department of Genetics,
Stanford University Medical Center,
Stanford,
CA 94306-5120, USA

Tel: +1 (650) 498 6012
Fax: +1 (650) 723 7016
E-mail:
sherlock@genome.stanford.edu

Clearly we want to correct this. Consider further that the experiment is repeated, but now there is only a 1.5 fold increase in labelling efficiency in the red channel. To compare these two experiments, they both need normalisation, to remove the systematic effect of different labelling efficiency of the two dyes.

Normalisation makes assumptions about the data

Most methods of normalisation make the assumption that the average (geometric mean) ratio is 1, or the average (arithmetic mean) log ratio is 0. This in effect says that the average gene does not change its expression under the condition being studied. This may or may not be true, but if you have good reason to believe that the average ratio is not going to be 1, then these commonly used methods are not applicable, and data normalisation will be more difficult. For instance an experiment where RNA polymerase is turned off would yield the expectation that almost all transcription will decrease, and the average ratio will be far below 1. The methods below will not be particularly suitable for that kind of experiment.

Normalisation is required to compare data

In addition to shifting the distribution of the log ratios, by making either the mean or median log ratio equal to 0, some methods also scale the distribution of the ratios to give a uniform standard deviation. However, consideration of the following:

> A yeast culture growing at 25 °C is split in 2, and one half is shifted to 26 °C, and the other half is shifted to 37 °C. The usual expectation would be that the changes in gene expression in the half shifted to 37 °C would be greater than those in the half shifted to 26 °C. Thus the ratios, when each sample is compared to the unshifted culture, would have a greater spread, and a higher standard deviation in the half shifted to 37 °C

suggests that scaling of the distribution will not accurately represent the degree of change that has actually occurred in the experiment.

Some commonly used normalisation methods

House-keeping genes

This method simply preselects some so-called housekeeping genes, and assumes that these genes do not change under the tested condition. A normalisation factor, which is calculated to make this set of genes have a mean ratio of 1, is then applied to all the data. This method is probably unreliable, as it has been observed in yeast for instance, that all genes show some change under some conditions.

Global mean or median normalisation

This method calculates a normalisation factor based on all the data (sometimes filtering out those spots that are not considered well measured, see Yang *et al.*[1] and Tseng *et al.*[2]), which when applied will make the mean log ratio of all the data equal to zero. Alternatively it may be applied to make the median log ratio zero, to avoid the effects of outliers. Alternatively stated, a histogram of log ratios yields a distribution that looks somewhat normal. This method simply shifts that distribution along the *x*-axis, so that it is centred on zero.

Intensity-dependent normalisation

This method,[3] using dye swap experiments, has demonstrated that instead of having a single normalisation factor, applied equally to all the data, that having an equation, whereby a normalisation dependent on the spot intensity is used, results in better normalisation of the data.

Spatial bias

It has been noted that often there is a spatial bias of ratios on a chip, such that a microarray image may have regions where the ratios are high, and another where they are low. Assuming that chip position is independent of the function of the gene that an element represents, this would be unexpected, and probably a consequence of some systematic variation. Thus instead of normalising all spots on an

array simultaneously, they could be split into subgroups, for instance by sector, and each subgroup be normalised independently.

It should be noted that the issues of normalisation of microarray data strongly suggest that, where possible, experimental design, and microarray design, be used to maximise the ability to analyse the data, and identify source of systematic variation. For instance, reverse-labelled duplicate experiments will help address normalisation issues, while reproduction of some spots on different places of the microarray may help address spatial biases.

Clustering is a simple way to organise the data

MISSING DATA

There are usually missing values in a microarray data matrix, which may arise for many reasons – some genes were not represented on all chips used, some spots were unusable on some arrays due to technical problems and experimental artefacts, or some spots have a signal that is below a threshold, resulting in the spot being flagged as unreliable. Intuitively, missing data will affect our ability to analyse the data. Some analysis methods (eg SVD, see below) cannot tolerate missing data. Use of such methods requires either that genes with missing data must be discarded (hardly an appealing option), or an estimate of the missing data must be made. Other methods, such as hierarchical clustering, tolerate missing data by treating the data as having fewer dimensions for those genes with missing values, which may affect the ability to analyse the data accurately. To address this problem, Troyanskaya *et al.*[4] propose estimating the missing data, and tested two algorithms for doing so. The most robust of these is a simple *k*-nearest neighbours technique. In this method each gene expression vector that is missing data is compared with all other gene expression vectors that are not missing the data. The missing data are then estimated using a weighted average of the appropriate data points in the *k*-most similar genes. They found[4] that the algorithm was somewhat insensitive to *k*, such that a range of *k* could be used (between 10 and 20 neighbours), without much effect on the outcome, and the technique was acceptable at estimating values for up to 15 per cent of the data for a gene expression vector. While estimates of missing data should be represented as such, and should be removed after the analysis is complete, it is clear that when analysing data, the sensitivity is increased when making a reasonable estimate of missing data, rather than simply leaving missing values blank.

Missing data impacts the ability to analyse it

CLUSTERING

Clustering is a simple but proven method for analysing gene expression data. With this method, the gene expression vectors that make up the microarray data matrix are reordered, to place similar vectors closer to each other within the matrix. Clustering can also be done in the second dimension of the matrix, that of samples/experiments, such that the experiment vectors may be reordered, eg Alon *et al.*[5] and Perou *et al.*[6] Where the arrays correspond to different cell types, this two-dimensional clustering serves as a method for distinguishing cell types from one another. Clustering in both the gene and experiment dimensions may be carried out sequentially on the same matrix. Several clustering techniques exist – here only two of them are highlighted.

Eisen *et al.*[7] and Wen *et al.*[8] have both used a bottom up (agglomerative) clustering technique. In this approach, all gene expression vectors are compared with each other, such that a matrix of correlations is generated. The largest correlation in the matrix defines the two most similar vectors, and these are then joined to form a node, which has a compound vector associated with it, which is calculated as the average of the vectors that contribute to it. This compound vector is then compared to all existing unjoined gene expression and compound vectors, and the process is then repeated. Thus single expression profiles are successively joined to form

nodes, which in turn are then joined further. The process continues until all individual profiles and nodes have been joined to form a single hierarchical tree. The utility of this approach is that it is simple, and the end result can be easily visualised, from which coordinately regulated patterns can be relatively easily discerned by eye (Figure 1a). When joining two nodes together, the nodes can be rotated in one of two possible ways, leading to four possible ways of combining them (Figure 1b). In a cluster tree of n leaves there are 2^{n-1} linear orderings consistent with the structure of the tree. To find the optimal solution (or solutions) is an NP-hard problem, as the runtime needed to consider all possible solutions grows exponentially with the number of leaves in the tree. A simple algorithm, implemented in XCluster,[9] simply rotates nodes around their roots when they are being joined, to place the most similar outer leaves of the nodes adjacent to each other. A more directed algorithm[10] formalises testing the quality of the leaf ordering, and reorders it to achieve a more parsimonious solution. The runtime of the algorithm is, however, of the order n^3, and also requires an amount of memory proportional to n^2. In contrast, the simple node switching in XCluster requires a constant, small amount of memory, and its impact on the runtime is proportional to n, where n is the number of genes.

False colour displays make interpretation of gene expression data easier

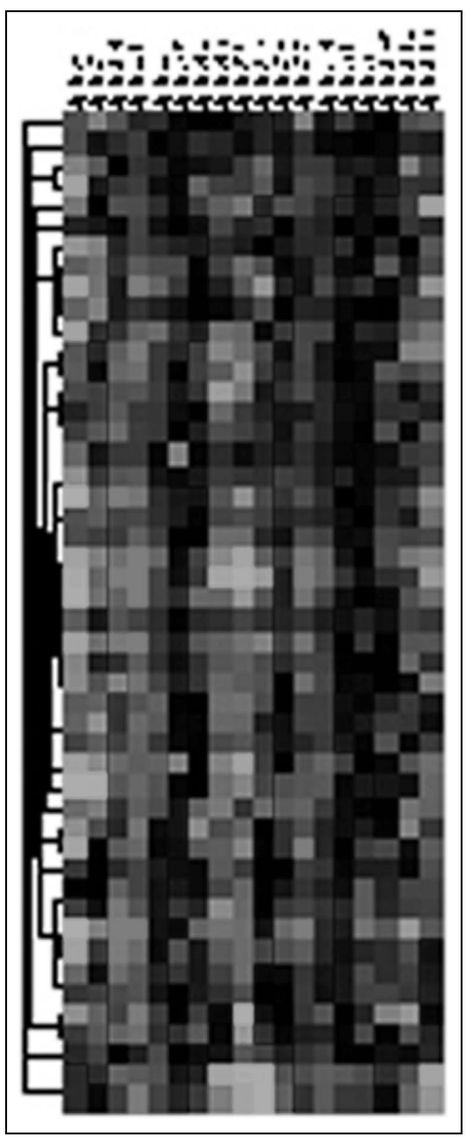

Figure 1a: An example hierarchical cluster.

The above discussion refers to average linkage clustering, whereby compound nodes are formed, and used for further vector comparisons. There are two additional methods of agglomerative hierarchical clustering, namely single linkage clustering and complete linkage clustering, both of which are supported by Mike Eisen's Cluster Software.[11] In single linkage clustering, instead of calculating the distance between two nodes as the distance between compound vectors representing those nodes, the distance is the minimum of all pairwise distances between all members of the two nodes. In complete linkage clustering the distance between two nodes is the maximum of all pairwise distances between all members of the two nodes. Single linkage clustering tends to produce long chains that form loose, straggly clusters, whereas complete linkage clustering tends to produce very tight clusters of similar profiles.

A divisive clustering method has also been applied to gene expression data,[5] whose approach is top down, rather than bottom up, in that it successively splits the

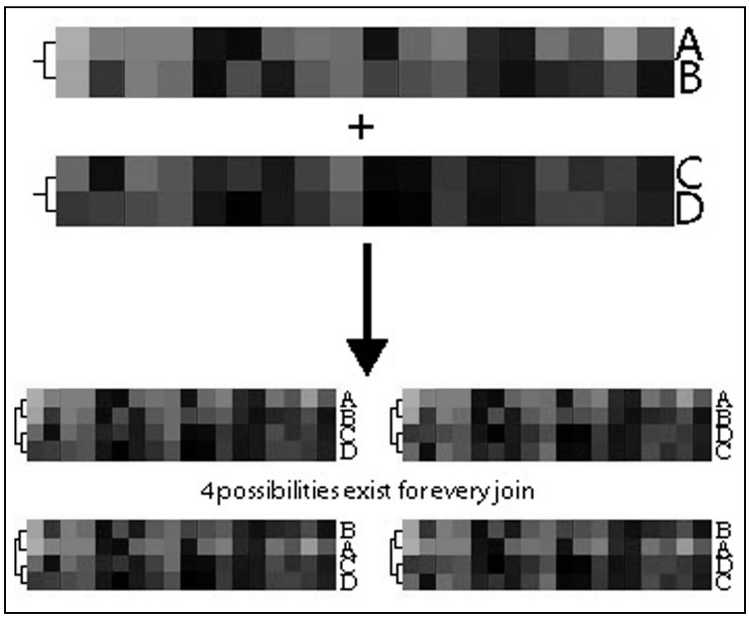

Figure 1b: An example of joining two nodes when hierarchically clustering, which indicates that there are four possible ways in which to join the nodes, by rotating each of them around their roots

A cluster tree is one of many possible trees

data into smaller and smaller clusters. Two random seed vectors are generated, and each gene expression vector is assigned to one of them, using a probability function. Iterative recalculation of the seed vectors is performed until they form the centroids of two clusters, which are then successively split in the same fashion, until each cluster consists of a single gene expression vector. A binary tree is then constructed from the history of the how the data were split. Alon *et al.*[5] also included a node-switching algorithm to order the branches in a somewhat optimal manner.

Clustering way lead to artefacts

The clustering methods described above can lead to artefacts. In the agglomerative method, the expression vector that represents a cluster, which is the average of all gene expression vectors that belong to the cluster, may not reflect accurately any of the contained vectors, especially as clusters become larger. Thus as one looks higher up in the tree, gene expression vectors within a cluster will become less similar to each other. In addition 'bad' decisions made early on during tree construction cannot be corrected later. A further drawback is that when clustering by arrays (columns), the similarity between each expression vector is typically calculated over the total number of genes within the data set, and is therefore only an overall measurement of similarity – indeed this is true also when clustering genes. Thus, if sample A is most similar to sample B overall, the fact that for a subset of genes, sample A is most similar to sample C, will be ignored, potentially losing some valuable biological information. Furthermore, it may simply be the case that a hierarchical structure does not apply to the data. There are clustering methods, which instead of organising data into a hierarchy, seek to partition the data into groups. Several clustering methods exist to partition expression data into groups; below the application of self-organising maps and *k*-means clustering is discussed.

DATA PARTITIONING USING SELF-ORGANISING MAPS

Self-organising maps (SOMs)[12] have been applied to gene expression data.[13,14] To initialise a SOM the number of partitions to use must be defined, as must their geometry with respect to each other, for

instance a 4 × 4 two-dimensional grid of 16 partitions. Each partition in the grid is more related to its neighbours than to distant partitions, and thus the geometry, as well as the number of partitions, will influence the outcome, ie a 1 × 16 grid will give a different, though similar, result from a 4 × 4 grid. Each partition is assigned a seed vector, which has the same number of data points, or coordinates, as there are experiments being considered, and is usually initialised with random data. Genes are then assigned to these partitions by an iterative method that manipulates these seed vectors. During each iteration the seed vectors, and thus the partitions, are recalculated to represent the expression data more closely, by the following sequence of events. A gene is picked at random, and its expression vector is compared to each of the seed vectors. The seed vector that is most similar to the expression vector of the picked gene is then modified, so that it more closely resembles the expression vector of that gene. In addition the seed vectors of the partitions that are physically closest (in the two-dimensional grid) to the partition whose vector was just modified are also modified, so that they too resemble the gene's expression vector a little more closely. This process is repeated thousands of times. With every iteration, the amount by which the seed vectors are altered decreases, and the definition of which partitions are close to each other also changes, eg in the first iteration all partitions may be considered close to each other, but after 50,000 iterations a partition may be considered close to another, only if it is less than half of the width of the entire grid away. At the end no partition may be considered close to another. Hence each iteration results in fewer vectors being modified by smaller amounts, so that the map eventually converges to a solution. Thus as the map is organised, the vectors of neighbouring partitions being somewhat similar to each other, and vectors of partitions that are physically distant being dissimilar to each other.

SOMs and *k*-means can be used to partition the data

Deciding how many partitions to make is a drawback of such methods

DATA PARTITIONING BY *k*-MEANS CLUSTERING

k-Means clustering[15] partitions data in a manner similar to self-organising maps, the key difference being that one partition does not directly influence another. *k*-Means may therefore be considered as one-dimensional. The seed vectors that are associated with each partition are initialised randomly, but differently from SOMs: the genes are immediately segregated to the partition with the most similar seed vector. These seed vectors are then recalculated as the centroids of the genes that mapped to them. This process is iterated until convergence is reached, which is the point where subsequent iterations do not result in genes being segregated to different partitions from one iteration to the next. *k*-Means clustering has been successfully used to analyse microarray data generated from studies of the yeast cell cycle.[16] It should also be noted that a drawback of *k*-means clustering is that the initial partitioning, which is based on random vectors, may greatly affect the final outcome, resulting in a local, rather than a global optimum.

An important consideration is how many partitions to make, and this is considered one of the main drawbacks of such methods. Several methods for determining the correct number of partitions to make have been suggested (see Milligan and Cooper[17] for discussion), including the Gap statistic,[18] which was designed with gene expression data in mind. The main goal when partitioning expression data is to reduce the within-cluster dispersion, such that each cluster is reasonably homogeneous, while at the same time the between-cluster dispersion is large (ie a partition is not inappropriately split into two or more similar partitions). Simply plotting the within-cluster dispersion (which for the purposes of the Gap statistic is defined as the sum of the squared Euclidean distances between all members of a cluster, divided by twice the number of members within the cluster, then summed over all clusters) results in a line that

decreases as the number of clusters increases. This make intuitive sense – the more clusters we have, the less variation we will have within each cluster, as they will have fewer and fewer members. However, looking at such a plot, there is often an elbow, or a point where the plot flattens markedly. The Gap statistic attempts to formalise detection of this point in the plot, by making the plot for real data, and also for a reference distribution of data, which is created by drawing random data from the same distribution as the original data. The difference between these two curves is then plotted, and where their difference is maximal (the details are somewhat simplified for discussion here), is the number of clusters, k, into which the data should be partitioned.

The Gap statistic tries to optimise the number of partitions

Even though partitioning of the data using either of the above methods helps avoid some of the problems associated with simple hierarchical clustering methods, neither of these methods is able to extract features from the data, in such a way that they find, for instance, a series of experiments over which a group of genes are co-expressed, even if that group of experiments forms only a small subset of the entire data set. The constraint of the methods discussed above is that they may cluster either genes or experiments separately. While they may cluster both, using two-way clustering, the clustering of each is independent of the other, and the methods are thus ill suited to find such features of a data set. What instead is required is a method that couples the two-way clustering, such that the two are dependent, and features that span a few genes and/or experiments can be discerned. Getz *et al.*[19] has applied a coupled two-way clustering analysis to gene expression data, such that they are able to identify subsets of genes and samples, such that when one is clustered by the other, stable and significant partitions emerge. As Getz *et al.* discuss, a naive way to achieve this would be to consider all possible submatrices of the original data, and apply standard clustering techniques to each of them, and keep track of all stable clusters. Such a method would guarantee finding all possible stable clusters, but is intractable, owing to the exponential increase in the number of submatrices that exist as the input matrix increases in size. Getz *et al.* thus define an efficient heuristic to generate such subsets that form stable clusters, using an iterative method. Performing standard two-way clustering on the entire data matrix initialises the process, and both genes and experiments that form stable clusters are stored. Each of the groups of genes that define a stable cluster are then combined with each of the groups of samples that form a stable cluster, and again standard two-way clustering is performed, to further identify groups of genes, and groups of samples that form stable clusters. This process is iterated until no new stable clusters can be found that satisfy some criteria, such as size or stability. In this fashion, only submatrices are considered whose parent matrices formed stable clusters of either genes or samples, thus significantly reducing the search space. The authors point out that any clustering method, with an attached notion of what is a stable cluster, can be used with this coupled two-way clustering (CTWC) technique, thus making it flexible. Intuitively, it helps us consider and, it is to be hoped, solve, the following – we have 500 samples/experiments, which span the range of what biological phenomena have been investigated. Genes may participate in many different processes, and thus different, overlapping groups may be co-regulated in different samples. CTWC may enable identification of such subgroups, and thus enable us to really get to the heart of the combinatorial nature of biological control.

CTWC finds significant subsets of genes and experiments

SINGULAR VALUE DECOMPOSITION

Background

Singular value decomposition (SVD) (also known as Karhunen–Loève expansion in pattern recognition[20] and principal components analysis in statistics[21]) is a

linear transformation, which relies on the following theorem from linear algebra: any $M \times N$ matrix, $\boldsymbol{A}$, whose number of rows M is greater than or equal to its number of columns N, can be written as the product of an $M \times N$ column-orthogonal matrix $\boldsymbol{U}$, an $N \times N$ diagonal matrix $\boldsymbol{W}$ with positive or zero elements (the *singular values*), and the transpose of an $N \times N$ orthogonal matrix $\boldsymbol{V}$. This can be written as:

$$\boldsymbol{A} = \boldsymbol{U} \times \boldsymbol{W} \times \boldsymbol{V}^{\mathrm{T}}$$

and is represented in Figure 2. For the case of gene expression data, our *expression data matrix* corresponds to $\boldsymbol{A}$, which has M genes and N experiments. The reason that this technique is of utility when analysing gene expression data, is that the matrix $\boldsymbol{V}^{\mathrm{T}}$ contains essentially the underlying patterns within the data (which may be referred to as Eigen vectors), while the matrix $\boldsymbol{W}$ (which contains the Eigen values) indicates how much information each Eigen vector contributes to the original data matrix. The matrix $\boldsymbol{U}$ contains coefficients for each gene, in each Eigen vector, thus indicating the amount of information contributed by each Eigen vector to each gene's expression vector. There are some important points to note: the matrix $\boldsymbol{V}^{\mathrm{T}}$ is orthonormal, ie each Eigen vector has unit length, and is at an angle of 90° from every other Eigen vector. The orthogonal nature of the Eigen vectors means that they are independent of one another. Secondly, the matrix $\boldsymbol{W}$ is a diagonal matrix, with positive values found on the diagonal line from top-left to bottom-right, with all other values being zero. When the matrices are sorted such the Eigen values in matrix $\boldsymbol{W}$ are in descending order, the order of Eigen vectors in $\boldsymbol{V}^{\mathrm{T}}$ will be from most important to least. In addition, the fraction of 'Eigen expression' for each Eigen vector can be calculated as:

$$p_l = \frac{\varepsilon_l^2}{\sum_{k=1}^{L} \varepsilon_k^2}$$

where ε_l is the lth Eigen value, in the diagonal matrix $\boldsymbol{W}$, and p_l indicates the relative significance of the lth eigenvector, in terms of the overall expression that it captures (Figure 2). So what is SVD good for?

Data normalisation

It has been found that the most significant Eigen vector for a gene expression data matrix is frequently a constant pattern, which dominates the data.[22] This may simply represent the 'steady state', and

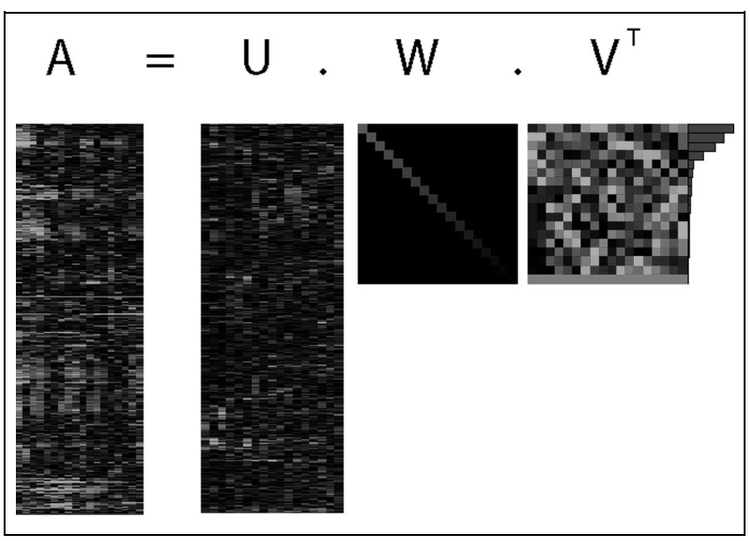

Figure 2: Singular value decomposition. Matrix *A* is the input expression data, the diagonal matrix, *W*, is the eigen values, matrix V^{T} is the eigen vectors, and matrix *U* is the coefficients for the genes in those vectors. The horizontal bars on the right indicate how much information each eigen vector captures (see text for details)

removal of this pattern allows the more biologically interesting patterns underneath to be better seen. This technique is akin to centring data, which is done by making the average expression for each gene equal to zero, such that the variation occurs centred on the *x*-axis. Removal of this pattern is simple: the Eigen value for this pattern, in matrix W, is set to zero, to yield matrix W', and a new matrix A' is then calculated as $U \cdot W' \cdot V^{T}$. This new matrix will contain the original expression data, but with the constant pattern filtered out.

Correlating expression information to sample information can add to prognostic and diagnostic information

Removal of experimental artefacts

It may be the case that an experimental artefact is known to exist in the data, for instance a patterns that correlates with the day of hybridisation of an array. SVD may be able to detect this pattern, and as described above, subsequently remove it. This means that a pattern can be removed, without any need to remove any genes or arrays from the data matrix.

Detection of biologically relevant patterns within the data

SVD finds patterns in the data

It has been found[22] that the Eigen vectors derived from a gene expression data matrix contain patterns with very real biological relevance. In the case studied by Alter *et al.*, after centring of the data by SVD, the two most prominent patterns within the data approximated a sine and cosine wave respectively. The data that were being analysed were from the yeast cell cycle study of Spellman *et al.*,[23] and so periodic patterns of expression were of biological relevance. Thus SVD is data-driven, and does not require any suppositions about what may be expected within the data. SVD has also been applied (in this case referred to as principal components) by Raychaudhuri *et al.*[24] to the sporulation data of Chu *et al.*,[25] and was successful in reducing the features of the data to their principal components, and assigning function to two of those, which captured nearly all of the contained information.

CORRELATING EXPRESSION DATA TO OTHER INFORMATION

While microarray data may form the vast majority of data associated with a microarray experiment, there is often other salient information about the experiment, which is equally important. Such information may be annotation of the samples, or of the genes themselves, and these data, in the context of the expression data, may be what is needed to give biological insight to the analysis. For instance, several different tumour subtypes may be assayed for expression, with the goal of identifying genes whose expression are predictive of the tumour subtype, or of correlating genes with survival information. Thus external sample information may be used in conjunction with gene expression data to glean either diagnostic or prognostic information. Alternatively, many of the genes within a selected group of genes with similar expression patterns (eg a SOM partition, or a subcluster from hierarchical clustering) may have associated annotations, which could potentially be used to annotate genes for which the process in which they participate is not known.

One of the first attempts to associate microarray data with clinical data was that of Golub *et al.*[26] who analysed the expression of 6,817 human genes in 27 acute lymphoblastic leukaemias and 11 acute myeloid leukaemias, in an effort to build a class predictor that could be used for tumour subtype diagnosis. They identified genes whose expression correlated to the class distinction to be predicted, and then tested whether the number of genes that appear to have predictive value was greater than would be expected by chance. They identified roughly 1,100 genes, and conservatively chose the 50 with the highest scores to be used as part of a predictor. 'Weighted votes' for each of these genes were then

used to assign unknown samples to one class or the other. Summation of the weighted votes for all the genes was then used to assign the new unknown sample to one class or the other, and the margin of victory of the vote allowed a confidence to be attributed to the assignment. Golub *et al.*[26] were able to correctly assign 36 of 38 of their samples, with the other two having uncertain assignment. On an independent set of 34 leukaemias, the technique made strong and correct predictions about 29 of them. Further they showed that these predictions were robust using predictors derived from between 10 and 200 genes.

SAM finds genes with significantly differential expression

A more generic method for associating microarray data with external data, such as clinical data, has been developed by Tusher *et al.*,[27] called Significance Analysis of Microarrays (SAM). SAM seeks to solve the following problem: if a gene is identified as having differential expression between two different classes of experimental sample, using a conventional *t*-test, then even if the probability is 0.01, if 5,000 genes are considered, there will be 50 false positives. To do this, SAM assigns a score to each gene on the basis of change in gene expression relative to the standard deviation of repeated measurements. Then, for genes with scores greater than an adjustable threshold, SAM uses permutations of the repeated measurements to estimate the percentage of genes that would be identified by chance, the false discovery rate. Of course the key component to SAM is thus setting this adjustable threshold. As the threshold is decreased, both the number of significant genes, as well as the number of false positives will rise. In practice, the threshold is likely to be set empirically, using a plot of the expected relative differences *v.* the observed relative differences. SAM[27] can be applied not only to two-class data, but by redefining the way in which the score is calculated, may be extended to multi-class data, survival times, paired data and quantitative parameters such as tumour stage, making it a versatile tool in the arsenal for microarray analysis.

Use of GO can help in interpreting microarray data

Using clusters of genes made by *k*-means clustering of the cell cycle data of Cho *et al.*,[28] Tavazoie *et al.*[16] devised a technique to determine whether the functional categories into which the genes within each cluster fell was significant. They used the MIPS functional categories,[29] and determined whether the overlap between the genes in a functional class and the genes in a particular expression cluster was greater than would be expected by chance. While some categories showed significant overlap with some of the clusters, it should be noted that the number of clusters that they used for the *k*-means clustering was a subjective number, and that the significance scores will change with different numbers of clusters. The concept is however of obvious interest. Further discussion of this idea is presented by Kell and King,[30] who discuss the need for methods to assign functional classes to gene expression data. Of significant promise towards this end is the gene ontology (GO), which is a controlled vocabulary to which genes may be annotated.[31,32] The most significant features of GO are (1) it is structured as a directed acyclic graph, such that a node may have several parents, rather than simply being a binary tree, (2) there are three separate ontologies, which represent the roughly orthogonal concepts of molecular function, biological process, and cellular component, and (3) genes may be annotated to as many nodes within any ontology as reflects their roles in the cell, and at any levels, depending on the current state of knowledge of that gene. Use of GO to do a similar study to that published by Tavazoie *et al.*,[16] but which takes advantage of the structure of GO, will require large numbers of high-quality annotations to GO, which are currently in progress by several model organism databases.

FUTURE PROSPECTS

While there are currently huge amounts of microarray data being generated, it is

clear that we have not exhausted either the experimental potential of microarrays, nor have we more than scratched the surface of the methods and algorithms that can be used to characterise and classify the data, with the goal of elucidating biology. An important lesson learned by the sequencing community was that access to large amounts of data drove the development of tools to analyse the data. For this to happen in the microarray community, not only do the data need to be made freely available, potentially in public repositories, but also those data need to be recorded in a format that allows automated queries and analyses to be run against the data. In addition, sufficient detail about the experiments themselves needs to be recorded such that sense can be made of interesting results that may arise from such analyses. There is movement in the microarray community to adopt both a standard set of minimal information that should be recorded about an experiment, as well as a standard format for recording and exchanging the data.[33] The fledgling public repositories have pledged support for both of these standards, as have two of the open source databases, though both standards are far from being finalised and adopted. Without such standards and repositories, we will be left with a morass of data, for which interpretation will be difficult. A second lesson that can be learnt from the sequencing community is that once sequence data had quality metrics attached, the ability to use the data, and assemble sequence reads with defined confidence, greatly increased. While there may be many different metrics and statistics associated with each spot on a microarray, there is no standard way of using that information to assess the reliability of a measurement, nor is there a standard method for combining replicate measurements, both from the same microarray, and different microarrays. Development of such methods will greatly enhance our ability to reach robust conclusions about the biology we are trying to observe and dissect.

Standards for data recording and exchange are needed

It is also important to remember that microarrays do not merely let us look at gene expression – there are in fact a myriad of applications for microarrays, such that we can investigate the differences between any two populations of nucleic acids. Thus in addition to looking at expression, we can investigate DNA copy number, to look at genomic differences,[34] we can look at DNA binding proteins going on and off their chromatin binding sites[35–37] and we can even look at the parental origin of DNA, using genomic mismatch scanning,[38] or using single nucleotide polymorphisms using Affymetrix arrays.[39,40] Investigating biological systems using microarrays to probe more than just the gene expression within those systems, and then combining those data, will allow us to build a more complete picture of the system, thus allowing us to get ever closer to our ultimate goal, which is simply to understand biology, and to use that understanding to our advantage, such as positively impacting human health.

Acknowledgement

The author would like to thank Catherine Ball for careful and critical reading of the manuscript, and many useful suggestions.

References

1. Yang, M. C., Ruan, Q. G., Yang, J. J. *et al.* (2001), 'A statistical procedure for flagging weak spots greatly improves normalization and ratio estimates in microarray experiments', *Physiol. Genomics*, Vol. 8, p. 8.
2. Tseng, G. C., Oh, M. K., Rohlin, L. *et al.* (2001), 'Issues in cDNA microarray analysis: Quality filtering, channel normalization, models of variations and asssessment of gene effects', *Nucleic Acids Res.*, Vol. 29(12), pp. 2549–2557.
3. Yang, Y. H., Dudoit, S., Luu, P. and Speed, T. P. (2001), 'Normalization of cDNA Microarray Data', technical report. URL: http://www.stat.berkeley.edu/users/terry/zarray/Html/normspie.html.
4. Troyanskaya, O., Cantor, M., Sherlock, G. *et al.* (2001), 'Missing value estimation methods for DNA microarrays', *Bioinformatics*, Vol. 17(6), pp. 520–525.
5. Alon, U., Barkai, N. Notterman, D. A. *et al.* (1999), 'Broad patterns of gene expression

revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays', *Proc. Natl Acad. Sci. USA*, Vol. 96(12), pp. 6745–6750.

6. Perou, C. M., Jeffrey, S. S., van de Rijn, M. *et al.* (1999), 'Distinctive gene expression patterns in human mammary epithelial cells and breast cancers', *Proc. Natl Acad. Sci. USA*, Vol. 96(16), pp. 9212–9217.
7. Eisen, M. B., Spellman, P. T., Brown, P. O. and Botstein, D. (1998), 'Cluster analysis and display of genome-wide expression patterns', *Proc. Natl Acad. Sci. USA*, Vol. 95(25), pp. 14863–14868.
8. Wen, X., Fuhrman, S., Michaels, G. S. *et al.* (1998), 'Large-scale temporal gene expression mapping of central nervous system development', *Proc. Natl Acad. Sci USA*, Vol. 95(1), pp. 334–339.
9. URL: http://genome-www.stanford.edu/~sherlock/cluster.html
10. Bar-Joseph, Z., Gifford, D. K. and Jaakkola, T. S. (2001), 'Fast optimal leaf ordering for hierarchical clustering', *Bioinformatics*, Vol. 17, Suppl. 1, pp. S22–S29.
11. URL: http://rana.lbl.gov/EisenSoftware.htm
12. Kohonen, T. (1995), 'Self Organizing Maps', Springer, Berlin.
13. Tamayo, P., Slonim, D., Mesirov, J. *et al.* (1999), 'Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation', *Proc. Natl Acad. Sci. USA*, Vol. 96(6), pp. 2907–2912.
14. Toronen, P., Kolehmainen, M., Wong, G. and Castren, E. (1999), 'Analysis of gene expression data using self-organizing maps', *FEBS Lett.*, Vol. 451(2), pp. 142–146.
15. Everitt, B. (1974), 'Cluster Analysis', Heinemann, London.
16. Tavazoie, S., Hughes, J. D., Campbell, M. J. *et al.* (1999), 'Systematic determination of genetic network architecture', *Nat. Genet.*, Vol. 22(3), pp. 281–285.
17. Milligan, G. W. and Cooper, M. C. (1985), 'An examination of procedures for determining the number of clusters in a dataset', *Psychometrika*, Vol. 50, pp. 159–179.
18. Tibshirani, R., Walther, G. and Hastie, T. (2000), 'Estimating the number of clusters in a dataset via the Gap statistic'. URL: http://www-stat.stanford.edu/~tibs/ftp/gap.pdf
19. Getz, G., Levine, E. and Domany, E. (2000), 'Couple two-way clustering analysis of gene microarray data', *Proc. Natl Acad. Sci. USA*, Vol. 97(22), pp. 12079–12084.
20. Mallat, S. G. (1999), 'A Wavelet Tour of Signal Processing', 2nd edn, Academic, San Diego.
21. Anderson, T. W. (1984), 'Introduction to Multivariate Statistical Analysis', 2nd edn, Wiley, New York.
22. Alter, O., Brown, P. O. and Botstein, D. (2000), 'Singular value decomposition for genome-wide expression data processing and modeling', *Proc. Natl Acad. Sci. USA*, Vol. 97(18), pp. 10101–10106.
23. Spellman, P. T., Sherlock, G., Zhang, M. Q. *et al.* (1998), 'Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization', *Mol. Biol. Cell*, Vol. 9(12), pp. 3273–3297.
24. Raychaudhuri, S., Stuart, J. M. and Altman, R. B. (2000), 'Principal components analysis to summarize microarray experiments: Application to sporulation time series', *Pacific Symp. Biocomput.*, pp. 455–466.
25. Chu, S., DeRisi, J., Eisen, M. *et al.* (1998), 'The transcriptional program of sporulation in budding yeast', *Science*, Vol. 282(5389), pp. 699–705.
26. Golub, T. R., Slonin, D. K., Tamayo, P. *et al.* (1999), 'Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring', *Science*, Vol. 286(5439), pp. 531–537.
27. Tusher, V. G., Tibshirani, R. and Chu, G. (2001), 'Significance analysis of microarrays applied to the ionizing radiation response', *Proc. Natl Acad. Sci. USA*, Vol. 98(9), 5116–5121.
28. Cho, R. J., Campbell, M. J., Winzeler, E. A. *et al.* (1998), 'A genome-wide transcriptional analysis of the mitotic cell cycle', *Mol. Cell*, Vol. 2(1), pp. 65–73.
29. Mewes, H. W., Frishman, D., Gruber, C. *et al.* (2000), 'MIPS: A database for genomes and protein sequences', *Nucleic Acids Res.*, Vol. 28(1), pp. 37–40.
30. Kell, D. B. and King, R. D. (2000), 'On the optimization of classes for the assignment of unidentified reading frames in function genomics programmes: The need for machine learning', *Trends Biotechnol.*, Vol. 18(3), pp. 93–98.
31. The Gene Ontology Consortium (2001), 'Creating the gene ontology resource: Design and implementation', *Genome Res.*, Vol. 11(8), pp. 1425–1433.
32. Ashburner, M., Ball, C. A., Blake, J. A. *et al.* (2000), 'Gene ontology: Tool for the unification of biology', *Nat. Genet.*, Vol. 25(1), pp. 25–29.
33. URL: http://www.mged.org
34. Pollack, J. R., Perou, C. M., Alizadeh, A. A. *et al.* (1999), 'Genome-wide analysis of DNA

copy-number changes using cDNA microarrays', *Nat. Genet.*, Vol. 23(1), pp. 41–46.

35. Ren, B., Rober, F. Wyrick, J. J. *et al.* (2000), 'Genome-wide location and function of DNA binding proteins', *Science*, Vol. 290(5500), pp. 2306–2309.
36. Lieb, J. D., Liu, X., Botstein, D. and Brown, P. O. (2001), 'Promoter-specific binding of Rap1 revealed by genome-wide maps of protein–DNA association', *Nat. Genet.*, Vol. 28(4), pp. 327–334.
37. Iyer, V. R., Horak, C. E., Scafe, C. S. *et al.* (2001), 'Genomic binding sites of the yeast cell-cycle transcription factors SBF and MBF', *Nature*, Vol. 409(6819), pp. 533–538.
38. Nelson, S. F. , McCusker, J. H., Sander, M. A. *et al.* (1993), 'Genomic mismatch scanning: A new approach to genetic linkage mapping', *Nat. Genet.*, Vol. 4(1), pp. 11–18.
39. Fan, J. B., Chan, X., Halushka, M. K. *et al.* (2000), 'Parallel genotyping of human SNPs using generic high-density oligonucleotide tag arrays', *Genome Res.*, Vol. 10(6), pp. 853–860.
40. Gentalen, E. and Chee, M. (1999), 'A novel method for determining linkage between DNA sequences: Hybridization to paired probe arrays', *Nucleic Acids Res.*, Vol. 27(6), pp. 1485–1491.