General nonlinear framework for the analysis of gene interaction via multivariate expression arrays

Seungchan Kim
Edward R. Dougherty
Texas A&M University
Department of Electrical Engineering
College Station, Texas 77843-3128

Michael L. Bittner
Yidong Chen
National Institutes for Health
National Human Genome Research Institute
Laboratory for Cancer Genetics

Krishnamoorthy Sivakumar
Washington State University
Department of Electrical Engineering
Pullman, Washington 99163

Paul Meltzer
Jeffrey M. Trent
National Institutes for Health
National Human Genome Research Institute
Laboratory for Cancer Genetics

Abstract. A *c*DNA microarray is a complex biochemical-optical system whose purpose is the simultaneous measurement of gene expression for thousands of genes. In this paper we propose a general statistical approach to finding associations between the expression patterns of genes via the coefficient of determination. This coefficient measures the degree to which the transcriptional levels of an observed gene set can be used to improve the prediction of the transcriptional state of a target gene relative to the best possible prediction in the absence of observations. The method allows incorporation of knowledge of other conditions relevant to the prediction, such as the application of particular stimuli or the presence of inactivating gene mutations, as predictive elements affecting the expression level of a given gene. Various aspects of the method are discussed: prediction quantification, unconstrained prediction, constrained prediction using ternary perceptrons, and design of predictors given small numbers of replicated microarrays. The method is applied to a set of genes undergoing genotoxic stress for validation according to the manner in which it points toward previously known and unknown relationships. The entire procedure is supported by software that can be applied to large gene sets, has a number of facilities to simplify data analysis, and provides graphics for visualizing experimental data, multiple gene interaction, and prediction logic.

Keywords: *c*DNA microarray; gene expression; genomic signals; nonlinear prediction.

Paper JBO-90045 received Aug. 25, 1999; revised manuscript received Apr. 4, 2000; accepted for publication May 4, 2000.

1 Introduction

Sequences and clones for over a million expressed sequenced tagged sites (ESTs) are currently widely available. Characterization of these genes lies behind the ability to collect them. Only 14% of identified clusters contain genes (even tenuously) associated with a known functionality. One way of gaining insight into a gene's role in cellular activity is to study its expression pattern in a variety of circumstances and contexts, as it responds to its environment and to the action of other genes. Recent methods facilitate large scale surveys of gene expression in which transcript levels can be determined for thousands of genes simultaneously. In particular, *c*DNA microarrays result from a complex biochemical-optical system incorporating robotic spotting and computer image formation and analysis.[1–5] Since transcription control is accomplished by a method which interprets a variety of inputs,[6–8] we require analytical tools for expression profile data that can detect the types of multivariate influences on decision making produced by complex genetic networks. In this paper we discuss a statistical-operational framework for finding associations between expression patterns of genes by determining whether knowledge of the transcriptional levels of a small gene set can be used to predict the transcriptional state of another gene. A feature of the method is that it allows one to incorporate knowledge of other conditions, such as the application of particular stimuli or the presence of inactivating gene mutations, as predictive elements, thereby broadening the classes of information that can be simultaneously evaluated in modeling biological decision making. Our focus is on a general framework: the determination-prediction paradigm for analysis of gene interaction, comparison of constrained and unconstrained prediction in the face of limited microarray replications, estimation of the degree of determination given limited replications, interpretation of the results, and software to assist interpretation. Experimental results will be given for the purposes of explanation and verification. A particular instance of the general methodology has been applied in a separate biological paper (see Sec. 4).[9] A methodological perspective is important for appreciating the range of applicability of the proposed framework, which is not limited to *c*DNA microarrays, but can be used for studying interaction in the context of other kinds of arrays.

The mechanism of intergene association is not a factor in statistical prediction. The only factor is the ability to predict the target level from the predictor levels. The predictor genes may be upstream or downstream from the target gene in the

Address all correspondence to Dr. Edward R. Dougherty, Electrical Engineering Dept., Texas A&M University, 3128 TAMUS, College Station, TX 77843-3128; E-mail: edward@ee.tamu.edu

1083-3668/2000/$15.00 © 2000 SPIE

actual genetic network, some may be upstream and some downstream, or they may be distributed about the network in such a way that their relation to the target gene is based on chains of interaction among various intermediate genes. Whatever the relationship of the predicting genes to the predicted, if knowledge of their states allows us to better predict the expression level of the target gene, then we infer there is some relationship—the better the prediction, the stronger the relation.

As the first step in carrying out nonlinear genomic prediction on gene expression profiles, data complexity is reduced by thresholding the changes in transcript level into ternary expression data: [-1 (down regulated), $+1$ (up regulated), or 0 (invariant)]. This simplification is motivated by the way in which analysis is carried out on *c*DNA microarrays and by the need to collect many samples where gene expression levels vary due to altered cellular states. To find connections between genes, enough conditions must be sampled to detect the independent functioning of different genetic networks. This amount of sampling requires data from numerous arrays. When viewed across many arrays, the absolute intensity of signal detected by each element of the detector in this hybridization based assay can be seen to vary based both on the process of preparing and printing the EST elements, and the processes of preparing and labeling the *c*DNA representations of the RNA pools. This problem is solved via internal standardization. An algorithm that first calibrates the data internally to each microarray and statistically determines whether the data justify the conclusion that expression is up regulated or down regulated with 99% confidence is used to detect significant changes in the transcript level.[10] Requiring a high confidence level insures that the logical values -1 and 1 represent significant down and up regulation, and do not result from experimental variability.

2 Nonlinear Multivariate Prediction

The purpose of nonlinear multivariate prediction (filtering) is to predict (estimate) the output of a nonlinear system. Consider a system **S** having inputs $X_1, X_2, \ldots, X_m$ to be observed and measured, along with other inputs, which we may have no way of measuring, and may not even be able to identify (Figure 1). We do not assume a known mechanism by which the output is determined, nor is there an assumption of causality. The prediction problem is to estimate the output of **S** given only the inputs $X_1, X_2, \ldots, X_m$. As indicated in Figure 1, we view $X_1, X_2, \ldots, X_m$ as input variables to a logical system **L** that yields a logical value Y_{pred} that best predicts the value Y that **S** would provide, given the knowledge of the inputs $X_1, X_2, \ldots, X_m$. Statistical training uses only the fact that $X_1, X_2, \ldots, X_m$ are among the inputs to **S**, the output Y of **S** can be measured, and a logical system **L** can be constructed whose output Y_{pred} statistically approximates Y. The underlying scientific assumption is that the full system **S** is beyond the reach of current technology and our knowledge of **S** is derived from its effect on observable input variables. The logic of **L** represents an operational model of our understanding. It is crucial to recognize that this operational model is contingent on existing technology, which determines the inputs that can be observed, the manner in which the inputs are measured, the accuracy of those measurements, and the amount of data that can be obtained to design **L** by statistical sampling.

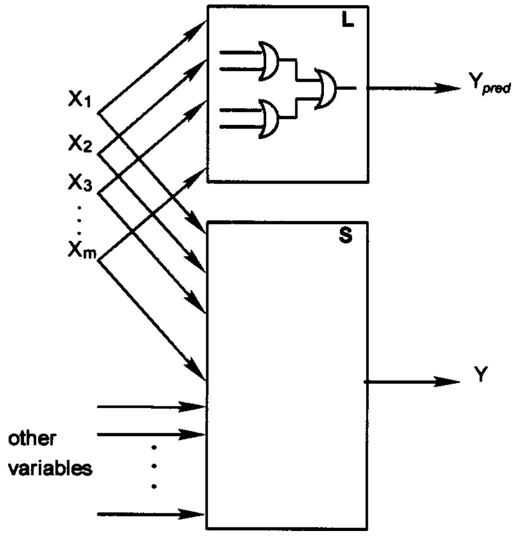


Fig. 1 Nonlinear multivariate system.

The most general nonlinear predictor (filter) for discrete data can be represented by a logic table. For ternary data, $\{-1, 0, +1\}$, the table will have a row for each input variable and one for the output variable Y_{pred}. If there are m input variables, then the table has 3^m columns and $m+1$ rows, and there are 3^{3^m} possible predictors (logic tables). For instance, if there are two predictor variables, X_1 and X_2, then a predictor is defined by the table

X_1	-1	-1	-1	0	0	0	1	1	1
X_2	-1	0	1	-1	0	1	-1	0	1
Y_{pred}	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9

where $y_1, y_2, \ldots, y_9 \in \{-1, 0, +1\}$. There are various ways to mathematically represent nonlinear predictors, including via logic circuitry.

The basic task is to design nonlinear predictors from the data. Qualitatively, the problem is straightforward: the theoretically optimal predictor of the target Y based on the predictor variables $X_1, X_2, \ldots, X_m$ is unknown and must be statistically estimated. The theoretically optimal predictor has minimum error across the population and must be designed (estimated) from a sample by some training (estimation) method. The degree to which a designed predictor approximates the optimal predictor depends on the training procedure and the sample size n. Even for a relatively small number of predictor genes, precise design of the optimal predictor requires a large number of experimental replications. The error, ϵ_n, of a designed estimate of the optimal predictor must exceed the error, ϵ_{opt}, of the optimal predictor. For a large number of microarrays, ϵ_n approximates ϵ_{opt}, but for the small numbers typically used in practice, ϵ_n may substantially exceed ϵ_{opt}. The irony of microarray analysis is that, although microarrays provide a large amount of data, these data are observed across a large set of gene expressions. As the num-

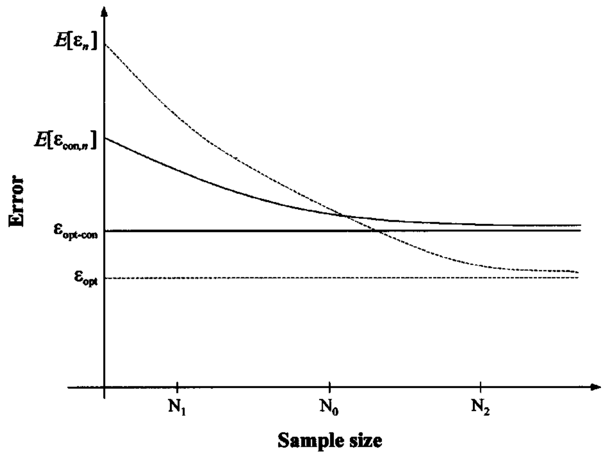


Fig. 2 Errors of unconstrained and constrained predictors.

ber of system inputs grows, the amount of replicated observations necessary for precise statistical design of an optimal predictor grows rapidly. Since a designed predictor depends on a randomly drawn sample data set, we use expectations for statistical analysis. Hence, we are concerned with the difference, $E[\epsilon_n]-\epsilon_{\text{opt}}$, between the expected error of the designed predictor and the error of the optimal predictor. A small difference means that $E[\epsilon_n]$ provides a good approximation to ϵ_{opt}.

The data problem can be mitigated if, instead of estimating the best predictor, we estimate the best predictor from a constrained set of predictors. Since the optimal constrained predictor is chosen from a subset of all possible predictors, its theoretical error exceeds that of the best predictor; however, the best constrained predictor can be designed more precisely from the data. The error, $\epsilon_{\text{con},n}$, of an estimate of the optimal constrained predictor exceeds the error, $\epsilon_{\text{opt-con}}$, of the optimal constrained predictor. We are concerned with the difference, $E[\epsilon_{\text{con},n}]-\epsilon_{\text{opt-con}}$.

If we let $\delta_n=E[\epsilon_n]-\epsilon_{\text{opt}}$ and $\delta_{\text{con},n}=E[\epsilon_{\text{con},n}]-\epsilon_{\text{opt-con}}$, then the dilemma of finding good predictors of gene expression levels can be seen to be threefold:

- $\epsilon_{\text{opt}}\leqslant\epsilon_{\text{opt-con}}$ and $\delta_n\geqslant\delta_{\text{con},n}$;
- ϵ_{opt} is decreased by using more predictor genes but δ_n is thereby increased;
- the stronger the constraint, the more $\delta_{\text{con},n}$ is reduced, but at the cost of increasing $\epsilon_{\text{opt-con}}$.

δ_n and $\delta_{\text{con},n}$ are the costs of design in the unconstrained and constrained settings, respectively. If we have access to an unlimited number of microarrays (and the design procedures do not themselves introduce error), then we could make both δ_n and $\delta_{\text{con},n}$ arbitrarily small and have

$$\epsilon_n\approx\epsilon_{\text{opt}}\leqslant\epsilon_{\text{opt-con}}\approx\epsilon_{\text{con},n}. \tag{1}$$

In our low-replication environment, δ_n can significantly exceed $\delta_{\text{con},n}$. Thus, the error of the designed constrained predictor can be smaller than that of the designed unconstrained predictor.

Figure 2 illustrates the design problem. The axes correspond to sample size n and error. The horizontal dashed and solid lines represent ϵ_{opt} and $\epsilon_{\text{opt-con}}$, respectively; the decreasing dashed and solid lines represent $E[\epsilon_n]$ and $E[\epsilon_{\text{con},n}]$, respectively. If n is sufficiently large, say, N_2, then $E[\epsilon_n]<E[\epsilon_{\text{con},n}]$; however, if n is sufficiently small, say, N_1, then $E[\epsilon_{\text{con},n}]<E[\epsilon_n]$. The point N_0 at which the decreasing lines cross is the cutoff: for $n>N_0$, the constraint is detrimental; for $n<N_0$, the constraint is beneficial. When $n<N_0$, the advantage of the constraint is measured by the difference between the decreasing solid and dashed lines.

Even if a designed constrained predictor does not perform well, the truly optimal constrained predictor may still perform well. Moreover, a less constrained predictor might provide good prediction had we a sufficient number of microarrays to design it. A critical point is that a constraint will err by missing a relationship, not erroneously indicating a strong relationship, thereby avoiding falsely attributing a predictive relation where none exists. Missed relationships depend on the constraint. Sometimes a system can be modeled in such a way that a constraint can be derived that does not yield increased error; however, this is not typical in nonlinear settings.

Perceptrons form a constrained class of nonlinear predictors that have some attractive properties: simplicity, a linear-like structure, and contributions of individual predictor variables that can be easily appreciated. The savings in replicates accelerates rapidly as the number of variables increases. Binary perceptrons are well studied and have long been used in pattern recognition.[11,12] They are also used extensively in digital signal processing, where they are called *linearly separable* operators.[13] For predicting a target expression value Y from predictor variables $X_1,X_2,\ldots,X_m$, a perceptron takes the form of

$$Y_{\text{pred}}=g(a_1X_1+a_2X_2+\ldots a_mX_m+b), \tag{2}$$

where g is a threshold function. A *binary perceptron* is defined by a binary threshold function, $g(z)=0$ if $z\leqslant 0$, and $g(z)=1$ if $z>0$. A *ternary perceptron* is defined by a ternary threshold function, $g(z)=-1$ if $z<-0.5$, $g(z)=0$ if $-0.5\leqslant z\leqslant 0.5$, and $g(z)=+1$ if $z>0.5$. For two predictors X_1 and X_2, the perceptron takes the form $Y_{\text{pred}}=g(a_1X_1+a_2X_2+b)$ and Y_{pred} is given by a table of the same form as that for a general nonlinear predictor. Now, however, $y_1,y_2,\ldots,y_9$ are determined by a_1, a_2, and b. There are 3^9 possible two-variable ternary nonlinear predictors, but the number of ternary perceptrons is only 417, which is only about 6% of the total. Design of a perceptron requires estimating the coefficients $a_1,a_2,\ldots,a_m$, and b. In the Appendix is a description of the stochastic training algorithm we have employed.

One way of constructing predictors that are less constrained than perceptrons is to use neural networks.[11,14] These comprise a form of operator representation that facilitates an easy increase and decrease in constraint. In neural-network terminology, the perceptron of Eq. (2) is called a *single-layer neural network* with *activation function g*. A *two-layer neural network* is of the form

$$Y_{\text{pred}}=g_2\left[\sum_{j=0}^{r}a_jg_1\left(\sum_{k=0}^{m}a_{jk}X_k\right)\right], \tag{3}$$

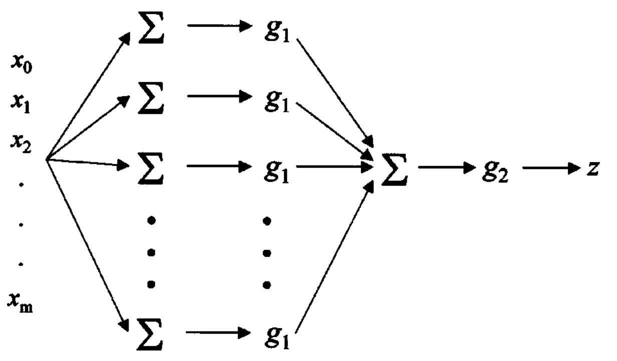


Fig. 3 Two-layer neural network.

where $X_0 \equiv 1$, and g_1 and g_2 are the activation functions for the first and second layers, respectively. The graph structure of the network is illustrated in Figure 3. Except for X_0, each node at the extreme left of the graph is an input variable; all other nodes are called *processing units*. The structure of the network determines the level of constraint. Sometimes it can be beneficial to extend a neural network beyond two layers; however, the unconstrained optimal predictor can be obtained by making the two-layer network sufficiently complex. There are a number of training algorithms available for neural networks. A key problem with using neural networks for microarray prediction is that multiple-layer stochastic training can be very imprecise when data sets are limited. In this paper we restrict ourselves to unconstrained and perceptron predictors.

3 Coefficient of Determination

If Y is real valued and error is the mean-square error (MSE), $E[|Y_{\text{pred}}-Y|^2]$, then the best predictor of Y in the absence of observations is its mean, μ_Y, and the error is σ_Y^2, the variance of Y. The best unconstrained predictor of Y based on the observed real-valued variables $X_1, X_2, \ldots, X_m$ is the conditional expectation of Y given $X_1, X_2, \ldots, X_m$. There is no general moment expression for the conditional expectation; however, there is one for the best linear predictor. Its form is that of Eq. (2), absent the threshold. The optimal coefficients are determined by the vector equation $\mathbf{A}=\mathbf{R}^{+}\mathbf{C}$, where $\mathbf{R}$ is the autocorrelation matrix for the random vector $\mathbf{V}=(1, X_1, X_2, \ldots, X_m)$, $\mathbf{C}$ is the cross-correlation vector for Y and $\mathbf{V}$, and $\mathbf{R}^{+}$ is the pseudoinverse of $\mathbf{R}$.[15] For digital processing, all predictors, including the conditional expectation and the optimal linear predictor, must be quantized.

The *coefficient of determination* of the optimal predictor is the relative decrease in error due to the presence of the observed variables.

$$\theta_{\text{opt}}=\frac{\epsilon_{\bullet}-\epsilon_{\text{opt}}}{\epsilon_{\bullet}}, \tag{4}$$

where $\epsilon_{\bullet}$ is the error for the best predictor in the absence of observations. Since $\epsilon_{\text{opt}} \leqslant \epsilon_{\bullet}$, $0 \leqslant \theta_{\text{opt}} \leqslant 1$. A similar definition applies for constrained predictors. So long as the constraint allows all constant predictors, $0 \leqslant \theta_{\text{opt-con}} \leqslant 1$. In statistics, the coefficient determination has been used to measure the significance of multiple linear regression.[16] It has recently been employed in nonlinear digital signal processing.[17]

For linear prediction using MSE, the coefficient of determination can be analytically expressed in terms of second-order moments of the observations and the target:

$$\theta_{\text{lin}}=\frac{\sigma_{X_k}}{\sigma_Y}\sum_{k=1}^{n} a_k \rho_{Y,X_k}+\frac{b\mu_Y-\mu_Y^2}{\sigma_Y^2}, \tag{5}$$

where ρ denotes the correlation coefficient. If we employ only the correlation coefficient $\rho_{Y,X}$ between two random variables X and Y, then our understanding concerns the prediction of Y from X via a linear formula of the form $Y_{\text{pred}}=aX+b$. If it happens that X and Y are jointly Gaussian, then the best predictor is linear, its error is $\sigma_Y^2(1-\rho_{Y,X}^2)$, and $\theta_{\text{opt}}=\theta_{\text{lin}}=\rho_{Y,X}^2$.

For most nonlinear predictors, including perceptrons, there is no moment expression such as Eq. (5) for the coefficient of determination. For the unconstrained ternary predictor,

$$\theta_{\text{opt}}=\frac{\epsilon_{\mu}-\epsilon_{\text{opt}}}{\epsilon_{\mu}}, \tag{6}$$

where ϵ_{μ} is the MSE from predicting Y by applying $T(\mu_Y)$, the ternary threshold of the mean of Y. For constrained predictors, ϵ_{opt} is replaced by $\epsilon_{\text{opt-con}}$ to obtain $\theta_{\text{opt-con}}$, and $\theta_{\text{opt-con}} \leqslant \theta_{\text{opt}}$.

For designed predictors, in Eq. (6), ϵ_{μ} is replaced by $\epsilon_{\mu,n}$, the error of the ternary threshold of the estimated mean resulting from the sample data, and ϵ_{opt} is replaced by ϵ_n to give

$$\theta_n=\frac{\epsilon_{\mu,n}-\epsilon_n}{\epsilon_{\mu,n}}. \tag{7}$$

$E[\theta_n]$, the expected sample coefficient of determination, is found by taking expected values on both sides of Eq. (7). $E[\epsilon_n] \geqslant \epsilon_{\text{opt}}$ always, and typically $E[\epsilon_n] > \epsilon_{\text{opt}}$, where the inequality can be substantial for small samples. Unless n is quite small, it is not unreasonable to assume that $\epsilon_{\mu,n}$ precisely estimates of ϵ_{μ}, since estimation of μ_Y does not require a large sample. Under this assumption, if we set $\epsilon_{\mu,n}=\epsilon_{\mu}$ in Eq. (7) and take expectations, we obtain

$$E[\theta_n] \approx \frac{\epsilon_{\mu}-E[\epsilon_n]}{\epsilon_{\mu}}. \tag{8}$$

Since $E[\epsilon_n] > \epsilon_{\text{opt}}$, Eqs. (6) and (8) yield $E[\theta_n] < \theta_{\text{opt}}$, and θ_n is a low-biased estimator of θ_{opt}.

For a constrained optimization, ϵ_n is replaced by $\epsilon_{\text{con},n}$ to obtain $\theta_{\text{con},n}$. In analogy to Eq. (1), if there is a sufficient number of microarrays, then

$$\theta_{\text{con},n} \approx \theta_{\text{opt-con}} \leqslant \theta_{\text{opt}} \approx \theta_n. \tag{9}$$

As the number of replicates increases, the approximations get better. In our low-replication environment, it is not uncommon to have $E[\theta_{\text{con},n}] > E[\theta_n]$.

We need data to estimate θ_n, as well as design predictors. This, too, is problematic due to limited replicates. For unconstrained predictors (and analogously for constrained predictors), one can use the *resubstitution* estimate, $\ddot{\theta}_n$. For resubstitution, all of the sample data are used to train (design) the best predictor, estimates of $\epsilon_{\mu,n}$ and ϵ_n are obtained by applying the thresholded estimated mean and the designed predictor to all of the training data, and $\ddot{\theta}_n$ is then computed by putting these estimates into Eq. (7). $\ddot{\theta}_n$ estimates θ_n, and thereby serves as an estimator of θ_{opt}. The resubstitution estimate can be expected to be optimistic, meaning it is biased high (Sec. 6).

A different approach is to split the data into training and test data, thereby producing *cross validation*. A predictor is designed from the training data, estimates of $\epsilon_{\mu,n}$ and ϵ_n are obtained from the test data, and an estimate of θ_n is computed by putting the error estimates into Eq. (7). Since this error depends on the split, the procedure is repeated a number of times and an estimate, $\hat{\theta}_n$, is obtained by averaging. For random sampling, the estimates of $\epsilon_{\mu,n}$ and ϵ_n are unbiased, and therefore the quotient of Eq. (7) will be essentially (close to being) unbiased as an estimator of θ_n (Sec. 6). Since θ_n is a pessimistic (low-biased) estimator of θ_{opt}, $\hat{\theta}_n$ is a pessimistic estimator of θ_{opt}.

Another issue is the number of predictor variables. For m and r predictor variables, $m<r$, if $\epsilon_{\text{opt}}(m)$ denotes the error for the m-variable predictor, then $\epsilon_{\text{opt}}(r)\leqslant\epsilon_{\text{opt}}(m)$. The prediction error decreases with an increasing number of variables. Hence, $\theta_{\text{opt}}(r)\geqslant\theta_{\text{opt}}(m)$. However, with an increasing number of variables comes an increase in the cost of estimation (the difference between the errors of the designed and optimal predictors). Intuitively, the information added by expanding the number of predictor variables becomes ever more redundant, thereby lessening the incremental predictive capability being added, whereas the inherent statistical variability in the new variables increases the cost (error) of design. Letting $\delta_n(m)=E[\epsilon_n(m)]-\epsilon_{\text{opt}}(m)$, we have $\delta_n(m)\leqslant\delta_n(r)$, and it may happen that $\epsilon_n(r)>\epsilon_n(m)$ and $\theta_n(r)<\theta_n(m)$. Since $\theta_{\text{opt}}(r)\geqslant\theta_{\text{opt}}(m)$, we choose the maximum between $\hat{\theta}_n(r)$ and $\hat{\theta}_n(m)$ as our estimator of $\theta_{\text{opt}}(r)$.

Cross validation is beneficial because $\hat{\theta}_n$ gives a conservative estimate of θ_{opt}. Thus, we do not obtain an overly optimistic view of the determination. On the other hand, training and testing on the same data provide large computational savings. This is important when searching over large combinations of predictor and target genes. Our current biological goal is comparative: we are interested in comparing coefficients of determination to find sets that appear significantly determinative of a particular target gene. In one case we are comparing high-biased values; in another, we are comparing low-biased values. If it happens that resubstitution and cross validation give similar comparative determinations, then using the resubstitution estimator can be practically beneficial. We will experimentally consider this question in Sec. 4.

4 Experimental Results

Tests of the ability of both the *full-logic* (unconstrained) predictor and the perceptron to detect associations based on changes in transcription level have been performed in the context of responsiveness to genotoxic stresses. As a result of a microarray study surveying transcription of 1238 genes during the response of a myeloid line to ionizing radiation,[18] 30 genes not previously known to participate in response to IR were found to be responsive. To further characterize the responsiveness of these genes to genotoxic stresses, the responsiveness of a subset of 9 of them was examined by blot assays in 12 cell lines stimulated with ionizing radiation, a chemical mutagen [methyl methane sulfonate (MMS)], or ultraviolet (UV) radiation. The cell lines were chosen so that a sampling of both *p*53 proficient and *p*53 deficient cells would be assayed.

As a blind control, expression patterns for two fictitious genes were created. Rules were made for the fictitious genes depending on other gene states in the set, and the degrees of concordance of the observations to the rules were varied. AHA has the rule: up regulated if *p*53 up regulated, down regulated if RCH1 and *p*53 down regulated. Full concordance with the rule would produce 15 instances of up regulation and 5 instances of down regulation. The data set generated for AHA includes 13 of the 15 up regulations and all 5 down regulations. OHO has the rule: up regulated if MDM2 up regulated and RCH1 down regulated, and down regulated if *p*53 down regulated and REL-B up regulated. Full concordance with this rule would produce four instances of up regulation and five instances of down regulation. The data set generated for OHO has the four expected up regulations plus seven unpredicted up regulations, and only two of the five predicted down regulations. The ternary data of the survey are given in Table 1, where the conditions IR, MMS, and UV have the values 1 or 0, depending on whether the condition is or is not in effect.

The genes in the survey are not uniformly regulated in the various cell types. All genes showed up or down regulation in at least one cell type, however the numbers of changes registered across the lines are quite variant. Such a varied response reflects the different ways in which different cells respond to the same external stimuli based on their own internal states, and is therefore a useful test set for the predictors. Since predictors operate by rules relating changes in one gene with changes in others, it is necessary that a target gene change a significant number of times in the observation set to get a meaningful prediction. We limit the target genes to those exhibiting at least 4 changes in the set of 30 observations, thereby eliminating MBP1 and SSAT as targets.

For the test genes (AHA and OHO), the designed predictors identified both the *p*53 and RCH1 components of the transcription rule set for AHA. For instance, using the perceptron and data splitting gives 0.785 as the (low-biased) estimate of the coefficient of determination. Since many rule violations were introduced into the data set for the OHO gene, it was expected that the coefficient of determination would not be high when using the predictors MDM2, RCH1, *p*53, and REL-B. This expectation was met.

For subsequent illustrations, determination results will be presented as arrow plots, with the target gene at the right and the chained predictors plotted to the left. The determination achieved by adjoining a predictor gene is placed on the arrow preceding it. For instance, in Figure 4, predictor 1 achieves determination θ_1 for predicting the target gene, using predictors 1 and 2 together achieves determination θ_2, and using predictors 1, 2, and 3 together achieves determination θ_3.

In cases where existing biological information provides expectations, the predictions conform to these expectations. The

Table 1 Ternary expression data.

		Genes														Condition		
Cell line	Condition	*RCH1*	*BCL3*	*FRA1*	*REL-B*	*ATF3*	*IAP-1*	*PC-1*	*MBP-1*	*SSAT*	*MDM2*	*p21*	*p53*	*AHA*	*OHO*	*IR*	*MMS*	*UV*
ML-1	IR	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
ML-1	MMS	0	0	0	0	1	0	0	0	0	1	1	1	1	0	0	1	0
Molt4	IR	-1	0	0	1	1	0	1	0	0	1	1	1	1	1	1	0	0
Molt4	MMS	0	0	1	0	1	0	0	0	0	0	1	1	1	0	0	1	0
SR	IR	-1	0	0	1	1	1	1	1	0	1	1	1	1	1	1	0	0
SR	MMS	0	0	0	0	1	0	0	0	0	1	1	1	1	0	0	1	0
A549	IR	0	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	0
A549	MMS	0	0	0	0	1	0	0	0	0	0	1	1	1	0	0	1	0
A549	UV	0	0	0	0	1	0	0	0	0	0	1	1	1	0	0	0	1
MCF7	IR	-1	0	1	1	0	0	0	0	0	1	1	1	0	1	1	0	0
MCF7	MMS	0	0	1	0	1	0	0	0	0	1	1	1	1	0	0	1	0
MCF7	UV	0	0	1	1	1	0	0	0	0	1	1	1	1	0	0	0	1
RKO	IR	0	1	0	1	1	1	1	0	0	1	1	1	1	0	1	0	0
RKO	MMS	0	0	0	0	1	0	0	0	0	0	1	1	1	0	0	1	0
RKO	UV	0	0	0	0	1	0	0	0	0	0	1	1	1	0	0	0	1
CCRF-CEM	IR	-1	1	1	1	1	0	1	0	0	0	0	-1	-1	0	1	0	0
CCRF-CEM	MMS	0	0	0	0	1	0	0	0	0	0	0	-1	0	0	0	1	0
HL60	IR	-1	1	0	1	1	0	1	0	1	0	1	-1	-1	-1	1	0	0
HL60	MMS	0	0	1	0	1	0	0	0	0	1	1	-1	0	1	0	1	0
K562	IR	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	1	0	0
K562	MMS	0	0	0	0	1	0	0	0	0	0	0	-1	0	0	0	1	0
H1299	IR	0	0	0	1	0	0	1	0	0	0	0	-1	0	0	1	0	0
H1299	MMS	0	0	0	0	1	0	0	0	0	0	1	-1	0	1	0	1	0
H1299	UV	0	0	0	0	1	0	1	0	0	0	1	-1	0	1	0	0	1
RKO/E6	IR	-1	1	0	1	0	1	1	0	0	0	0	-1	-1	0	1	0	0
RKO/E6	MMS	-1	0	0	0	1	0	0	0	0	0	1	-1	-1	1	0	1	0
RKO/E6	UV	-1	0	0	0	1	0	0	0	0	0	1	-1	-1	1	0	0	1
T47D	IR	0	0	0	1	0	0	0	0	0	0	1	-1	0	-1	1	0	0
T47D	MMS	0	0	0	0	1	0	0	0	0	0	1	-1	0	1	0	1	0
T47D	UV	0	0	0	0	1	0	0	0	0	0	1	-1	0	1	0	0	1

Rows are cell lines subjected to different experimental conditions.
Comparisons are to the same cell line not exposed to the experimental treatment.
-1 means expression goes down relative to untreated
0 means expression is unchanged relative to untreated
+1 means expression goes up relative to untreated

biological expectation is that MDM2 is incompletely, but strongly, predicted by *p*53. As shown in Figure 5(a), using the perceptron and data splitting, this expectation is met. Additions of further genes to *p*53 do not increase the accuracy of the prediction. Similarly, as it is known that *p*53 is influential, but not determinative of the up regulation of both *p*21 and MDM2, some level of prediction of *p*53 should be possible by a combination of these two genes. This expectation is also met [Figure 5(b)]. Moreover, as *p*21 shows both *p*53 dependent and *p*53 independent regulation in response to genomic damage,[19] it was expected that the *p*53 component would not be recognized by the algorithm. *p*53 was not selected for the predictor. The algorithm chose the somewhat similar pattern of expression exhibited by ATF3, with some supplementary information from the MDM2 pattern as the best predictor of *p*21 [Figure 5(c)]. The prediction carries little significance.

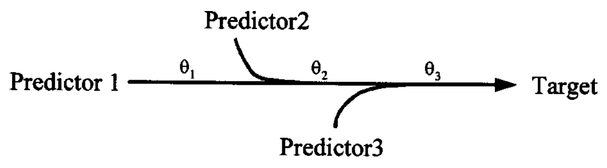


Fig. 4 Determination diagram (template).

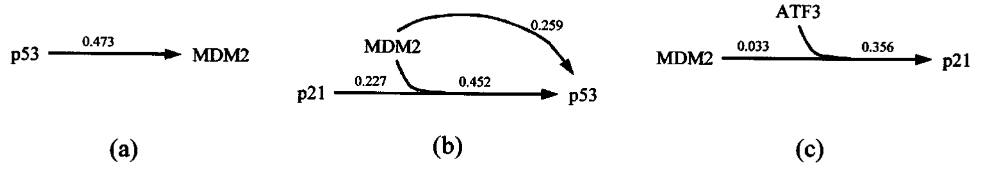


Fig. 5 Determination diagrams where there is consistency with biological information.

Among the newly found, IR responsive genes, FRA1, ATF3, REL-B, RCH1, PC1, IAP-1, and MBP-1, a set of relationships is seen that appears to link the behaviors of REL-B, RCH1, PC-1, MBP-1, BCL3, IAP-1, and SSAT. A set of perceptron and full-logic prediction trees for REL-B, PC-1, RCH1, and IAP-1 are shown in Figure 6. Both full-logic and perceptron predictors find a variety of apparently significant similarities of expression behavior within this set. Some of these genes also show a high degree of predictability based solely on exposure to ionizing radiation. When these genes are viewed with an eye to IR responsiveness, it becomes apparent that they share an overall trend to show expression level changes in response to IR rather than to UV or MMS. Even though MBP1 and SSAT only responded to IR at the very low rate of 17% of the possible chances, they responded only to this stimulus, and not to the other genotoxic stimuli, and were thus associated with other genes showing similarly high preference. Here we have only briefly outlined the manner in which the prediction methodology is consistent with existing biological knowledge and has the power to find new, potentially significant relations. In another paper, we have gone into much greater biological detail for the same genotoxic stress data using the perceptron and resubstitution estimation of the coefficient of determination.[9]

Using data splitting, we illustrate the error relation between the full-logic and perceptron predictors. The estimated determinations of the best full-logic and perceptron predictors of BCL3 in terms of IAP-1, PC-1, and SSAT are $\hat{\theta}_n = 0.334$ and $\hat{\theta}_{\mathrm{con},n} = 0.461$, respectively. Since the coefficient of the full-logic predictor exceeds the coefficient of the perceptron, the optimal full-logic predictor has determination greater than 0.461. We cannot derive this directly from the data, only from the fact that $\epsilon_{\mathrm{opt}} \leq \epsilon_{\mathrm{opt\text{-}con}}$. In fact, it could be that the optimal full-logic predictor is a perceptron, but this cannot be shown from the limited data we have.

Now consider prediction of BCL3 in terms of RCH1, SSAT, and $p21$. The best full-logic and perceptron predictors have estimated determinations $\hat{\theta}_n = 0.652$ and $\hat{\theta}_{\mathrm{con},n} = 0.431$, respectively. Constraining prediction to a perceptron underestimates the degree to which BCL3 can be predicted by RCH1, SSAT, and $p21$. In fact, the true coefficient of determination for the optimal full-logic predictor is likely to be significantly greater than 0.652. Moreover, performance of the optimal full-logic predictor surely exceeds that of the optimal perceptron by more than the differential 0.221, but this cannot be quantified from the data. We can, however, conclude with confidence that, to the degree that a perceptron approximates linear prediction, the substantial superior performance of the full-logic predictor shows that the relation among RCH1, SSAT, and $p21$ (as predictors) and BCL3 (as target) is strongly nonlinear.

The different ways in which full-logic and perceptron predictors operate to find relationships can be illustrated by examining the prediction each makes for the target gene BCL3. In Figure 7, the upper pair of determination trees represents

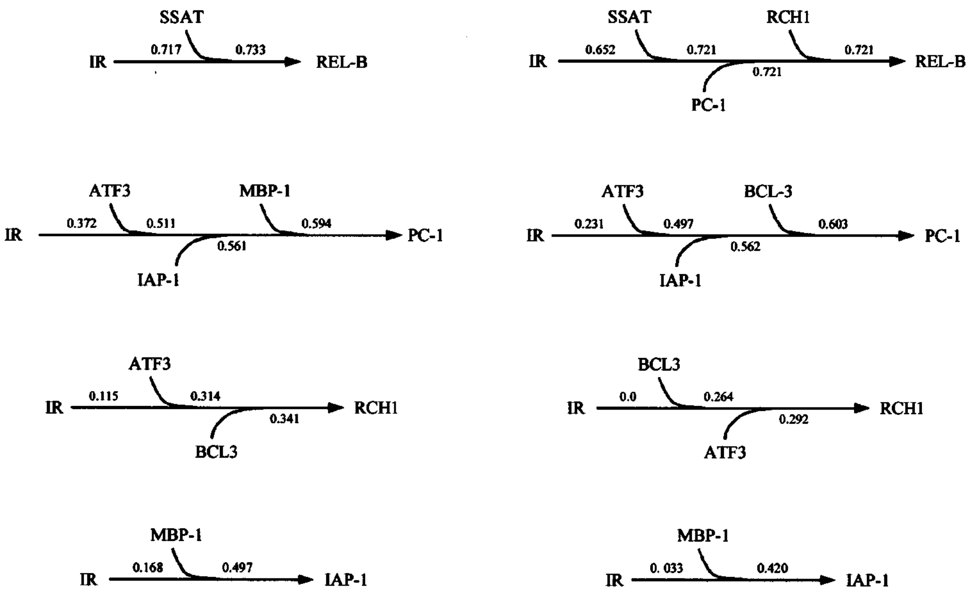


Fig. 6 Perceptronv (left) and full-logic (right) prediction trees.

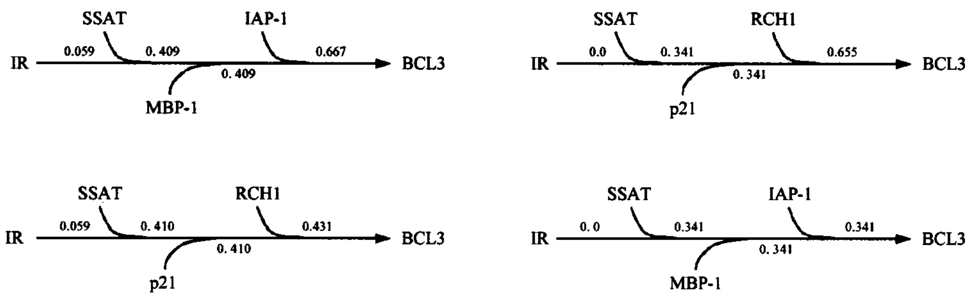


Fig. 7 Interchanging the best full-logic (right) and perceptron (left) predictors.

the optimal gene sets chosen by the perceptron (left) and full-logic (right) predictors. The lower pair shows the reduced determination achieved when the genes optimum for the full-logic predictor are used by the perceptron, and vice versa. Differences in choices made by predictors result from the different computation constraints they impose on input measurements.

We return to the issue of comparative determination using the resubstitution estimator $\ddot{\theta}_n$ as opposed to the cross-validation estimator $\hat{\theta}_n$. Due to the high and low biases of $\ddot{\theta}_n$ and $\hat{\theta}_n$, respectively, as estimators of θ_{opt}, the cutoffs for determination values considered meaningful will differ; nevertheless, the orders of the coefficients' magnitudes may still be consistent. Such consistency has useful practical consequences. Under present technological and cost constraints, experiments typically have less than 100 replicates (microarrays). Hence, the statistics being computed from the sample data are likely to be imprecise estimators of the corresponding population parameters. The coefficient of determination is being used mainly for comparative purposes to sift out potentially strong control relations among genes and external conditions. Consequently, overly optimistic estimates are not necessarily detrimental, nor are overly pessimistic estimates. If resubstitution and cross-validation error estimation yield coefficients that rank predictor-target sets in much the same order, then either can be used for comparative purposes without detriment, and the significantly faster computation of resubstitution estimates becomes a deciding factor. Only consistency among large coefficients is important, because these indicate potentially significant biological control relations.

Using two predictors per target, we have ranked the coefficients by using both methods and compare the orderings. For each target gene, a graph has been constructed with x values being the determination ranks (in order) for $\ddot{\theta}_n$ and the y value for x being the determination rank for $\hat{\theta}_n$ corresponding to the predictor pair having rank x. We have also plotted the graph of the coefficients versus the training-estimation ranks. The graphs for PC-1 are shown in Figure 8. If the orderings are consistent, then the (x,y) values should lie relatively close to the 45° line through the origin. Quantification of this condition is problematic due to the large number of rank ties for $\hat{\theta}_n$ that result from the maximization done during estimation. Nonetheless, we have noticed that for ranks corresponding to $\ddot{\theta}_n \geqslant 0.5$ there is strong consistency. This consistency can be experimentally quantified by counting the number of points lying in the box determined by the $\ddot{\theta}_n$ rank, $x_{0.5}$, corresponding to 0.5. In order not to break strings of ties for $\hat{\theta}_n$, if necessary, the box is expanded a minimal amount so that it contains a contiguous string beginning at or before $x_{0.5}$. We consider the points in the box to be consistently estimated by the two methods. Those above and to the right of the box correspond to too low or too high a $\hat{\theta}_n$ ranking corresponding to $\ddot{\theta}_n$, respectively. We let α, β, and γ be the counts in, above, and to the right of the box, respectively. Four genes have a sufficient number of predictor pairs with $\ddot{\theta}_n \geqslant 0.5$ to make the numbers meaningful. For REL-B, $x_{0.5}=38$, $\alpha=33$, $\beta=5$, and $\gamma=5$; for IAP-1, $x_{0.5}=16$, $\alpha=15$, $\beta=1$, and $\gamma=1$; for PC-1, $x_{0.5}=33$, $\alpha=32$, $\beta=1$, and $\gamma=5$; and for AHA, $x_{0.5}=14$, $\alpha=14$, $\beta=0$, and $\gamma=0$. Even when there are points outside the box, they tend to be close to the box. Moreover, were we to let $x=25$ be the box cutoff for REL-B, then $\beta=0$ and $\gamma=1$. While the last values support the notion that training-data estimates can be used for comparison purposes, the values for $x_{0.5}$ are more significant because they represent the manner in which one might look for significant determination by selecting a determination cutoff. Choosing a determination value greater than 0.5 enhances the consistency between the two orderings. For instance, for REL-B and $x_{0.6}=26$, $\alpha=26$, $\beta=0$, and $\gamma=0$.

5 Software

We briefly describe the determination software that has been developed (and continues to be developed) to process microarray data and provide tabular and graphical tools to assist analysis. While the data set in this paper consists of relatively few genes, the interactive software can be used in a workstation environment for gene sets approaching 1000 genes. More genes can be used as predictors in a supercomputing environment. In an experiment involving a collection of microarrays,

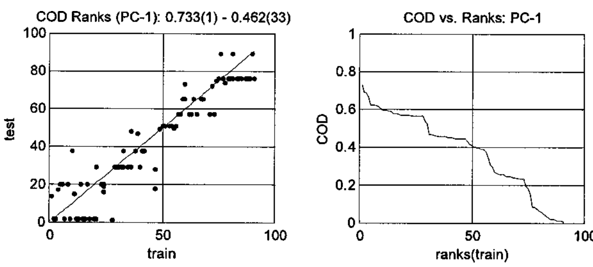


Fig. 8 Comparison of determination ranks.

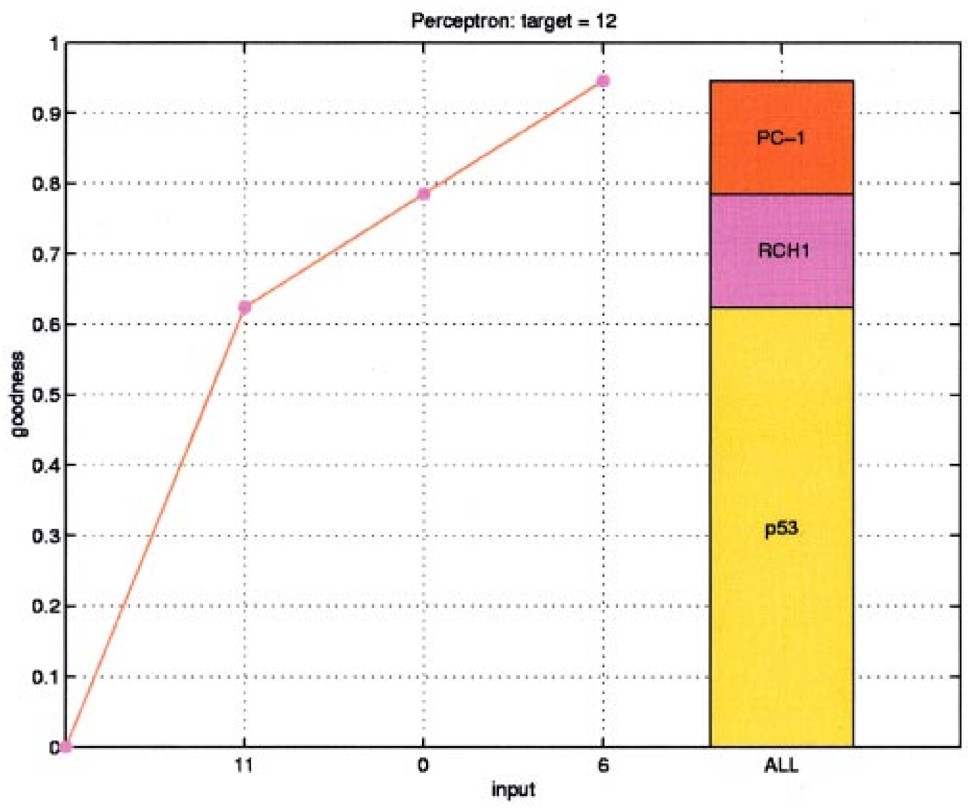


Fig. 9 Determination graph for increasing number of predictor variables.

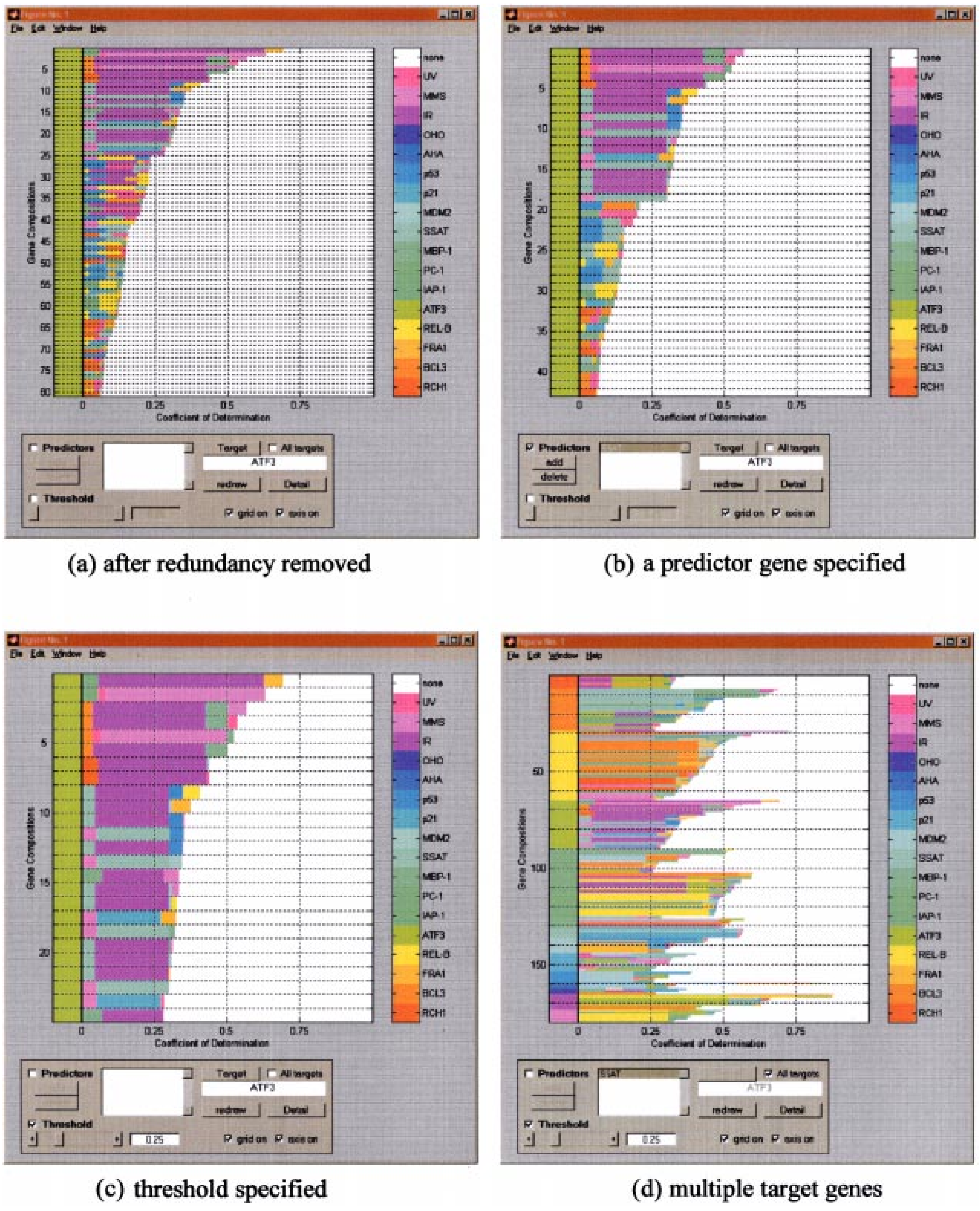
each might contain up to 8000 genes, but only those genes whose expression levels change some minimal number of times across the experiment are considered. Those that change only a few times provide very little information and do not yield meaningful determination coefficients. A consequence of this information requirement is that far fewer than the full 8000 genes are likely to be considered.

For a given target gene, the software computes the determination coefficients for all possible combinations of one, two, and three predictors. With a further restriction on the number of genes, it can also handle four predictors. Since we are computing a biased sample coefficient, given the modest number of microarrays we typically have for an experiment, it is prudent not to use too many predictor variables. The determination coefficients are computed offline and stored in tables for a researcher to access. For more than a single variable, these tables are quite long. Therefore a number of tools have been included in the software to facilitate online analysis.

- The user can specify a range in each table, from the predictor set with the *n*th highest determination to the set with the *m*th highest, $n<m$.
- Once a range is specified, the table can be scrolled downward.
- If one is interested in one or more genes being required for determination, these can be specified and a new table displayed with this requirement.
- If a gene, or gene set, has high determination, then adjoining further genes to the set can only increase the determination; therefore one can limit the number of times any gene or gene set appears in the table, including the possibility of omitting any combination containing a particular gene.
- Of special interest are situations in which adjoining a gene to a gene set yields a significant increase in determination; and therefore one can choose a threshold δ and display only those predictor sets for which there is at least a δ increase in determination over the determination for any subset for which the gene has been omitted.
- One can delete from the table any gene whose expression levels did not change some minimal number of times across the experiments.
- At any time, the software can be queried for a specific target and predictor set.

Various graphical tools are included. For any predictor set and target, a graph can be plotted online to show the increase in determination as the predictor set grows. The order of inclusion of the predictor genes can be specified, or the graph can be automatically displayed so that the best single predictor in the gene set is shown first and, given that gene, the best two are shown next, and so on. Figure 9 shows the graph for the training-data determination estimate of AHA by predictor genes PC-1, RCH1, and *p*53 where the numbering refers to the gene listing in Table 1.

Rather than simply show the performance of a single predictor set for a given target gene, the software allows visualization for various predictor sets for a given target gene, or for more than a single target gene. Figure 10(a) displays perfor-



ATF3

SSAT

0.25

Fig. 10 Multiple predictor set visualization. (a) After redundancy removed, (b) threshold specified, (c) a predictor gene specified, (d) multiple target genes.

mance bars for all possible predictor sets for the target gene ATF3 after redundancy has been removed. Redundancy occurs because adjoining a gene to a predictor set may not improve the coefficient of determination. This redundancy need not be visualized. One may also require that a particular gene, or combination of genes, be included in the predictor set. This situation is shown in Figure 10(b), where SSAT is required to be among the predictors. One may also simplify the visualization by specifying a threshold coefficient of determination that must be exceeded, as in Figure 10(c) where the threshold is 0.25. If a user finds an interesting predictor set, then more detailed information, such as gene clone-id, can be pulled from the database. Visualization can be done for more than a single target gene. Figure 10(d) shows a display for multiple target genes in which the determination threshold 0.25 must be exceeded.

For three predictors, a cubical graph can be generated on-line that displays the data for each predictor set. Figure 11 shows the data graph for the gene set of Figure 9. For three genes, g_1, g_2, and g_3, each microarray gives three values that appear as a ternary-valued vector (x_1,x_2,x_3). At each spatial point of the graph (having coordinates -1, 0, or 1) are plotted spheres whose sizes indicate the relative number of times that a particular coordinate triple was associated with the target value -1 (green sphere), 0 (yellow sphere), and 1 (red sphere) in the data. This graph shows the degree to which the target values tend to separate the predictor vectors. The actual counts are also displayed on the graph, along with the fraction of time a predictor vector appears among the experimental data. The graph can be rotated and viewed from different angles.

Graphically, a ternary perceptron separates the spatial points in the graph by two planes so that, once designed, all of the points in the same region (among the three regions) have the same perceptron value. For any three-gene predictor set, the software plots the optimal perceptron planes and displays the points within the appropriate regions as green, yellow, or red, according to the perceptron outputs -1, 0, and 1, respectively. Figure 12 shows a graphical display of the perceptron corresponding to Figure 9 and the data graph of Figure 11. The display can be rotated.

Another facility shows a logical circuit implementation of the optimal predictor for a given gene set. We use a comparator-based logic architecture based on the signal representation theory of mathematical morphology.[20] This architecture is straightforward and reflects the decision procedures inherent in a logical table. The expression levels of the predictor genes are input in parallel into two banks of comparators, each of which is an integrated circuit of logic gates and each being denoted by a triangle having two terminals. One terminal receives the input (x_1,x_2,x_3) and the other has a fixed vector input (t_1,t_2,t_3). If (t_1,t_2,t_3) is at the upper terminal, then the comparator outputs 1 if $x_1 \leq t_1$, $x_2 \leq t_2$, and $x_3 \leq t_3$; otherwise, it outputs 0. If (t_1,t_2,t_3) is at the lower terminal, then the comparator outputs 1 if $x_1 \geq t_1$, $x_2 \geq t_2$, and $x_3 \geq t_3$; otherwise, it outputs 0. The outputs of each comparator pair enter an AND gate that outputs 1 if and only if the inputs are between the upper and lower bounds of the comparator pair. The AND outputs in the upper bank enter an OR gate, as do the AND outputs of the lower bank. The outputs of the OR gates enter a multiplexer. There are three possibilities: (1) both OR gates output 0, in which case the multiplexer (and hence the circuit) outputs 0; (2) the upper OR gate outputs 1 and the lower OR gate outputs 0, in which case the multiplexer outputs -1; (3) the upper OR gate outputs 0 and the lower OR gate outputs 1, in which case the multiplexer outputs 1. Figure 13 shows the logic implementation for the perceptron of Figure 12.

6 Statistical Considerations

There are many statistical issues related to prediction in the context of microarrays. We have previously emphasized the need for employing a small number of predictor variables and the comparative manner in which the estimated coefficients are being used. Here in Sec. 6 we point out some general relations between the expected error of a trained predictor and the error of the optimal predictor, and between the error estimators and the errors they are estimating. We have postponed discussion of these issues until this point so as not to break the flow of the paper for those interested mainly in methodology and application software.

Figure 2 is generic but can be quantified for various prediction settings. One case is directly applicable here. When the predictors are gene expression levels and the target is a condition, IR, MMS, or UV, the output of the predictor is binary and the input is ternary. In this case,

$$E[\epsilon_n] \leq \epsilon_{\text{opt}} + \frac{3^m}{en} + \sqrt{\frac{3^m}{2(n+1)}}, \tag{10}$$

where e is the base of the natural logarithm.[14] The expectation of the error for the designed filter is relative to all possible samples from the joint probability distribution of the predictors and target. The bound requires no distributional assumptions and therefore may be loose for specific cases; nonetheless, it indicates the exponential need for more replicates as m increases. The need can be cut if the input data is only binary, with 3^m being replaced by 2^m in the bound.

The resubstitution estimator is a low-biased estimator of ϵ_n, $E[\ddot{\epsilon}_n] \leq E[\epsilon_n]$,[21] and a low-biased estimator of the error of the optimal predictor, $E[\ddot{\epsilon}_n] \leq \epsilon_{\text{opt}}$.[22] The problem is exacerbated if we employ a large number of variables. This can be seen by considering the MSE of the resubstitution estimator of ϵ_n. An upper bound for this error is given by

$$E[|\ddot{\epsilon}_n - \epsilon_n|^2] \leq \frac{6}{n} 3^m, \tag{11}$$

where 3^m is replaced by 2^m for binary input data.[14] Once again we see the exponential relationship between the number of replicates and the number of predictor variables.

Assuming random sampling, data splitting provides an unbiased estimator ϵ_n: $E[\hat{\epsilon}_n] = E[\epsilon_n]$. On average, $\hat{\epsilon}_n$ estimates the mean of ϵ_n, and on average it provides a conservative (pessimistic) estimate of ϵ_{opt}. For large n the MSE for $\hat{\epsilon}_n$ as an estimator of ϵ_n is small, but not for small n. The following distribution-free bound indicates the dependence on n,[14]

$$E[|\hat{\epsilon}_n - \epsilon_n|^2] \leq \frac{1+6e^{-1}}{n} + \frac{6}{\sqrt{\pi(n-1)}}. \tag{12}$$

For small n, $\hat{\epsilon}_n$ provides imprecise estimation of ϵ_n. The randomized data-splitting method we use provides a precise estimator of the error of predictors based on 20-sample designed predictors within the overall 30 samples. However, the precision of that estimator relative to the population of all possible similar stress-related microarrays depends on the degree to which the full sample reflects the entire population.

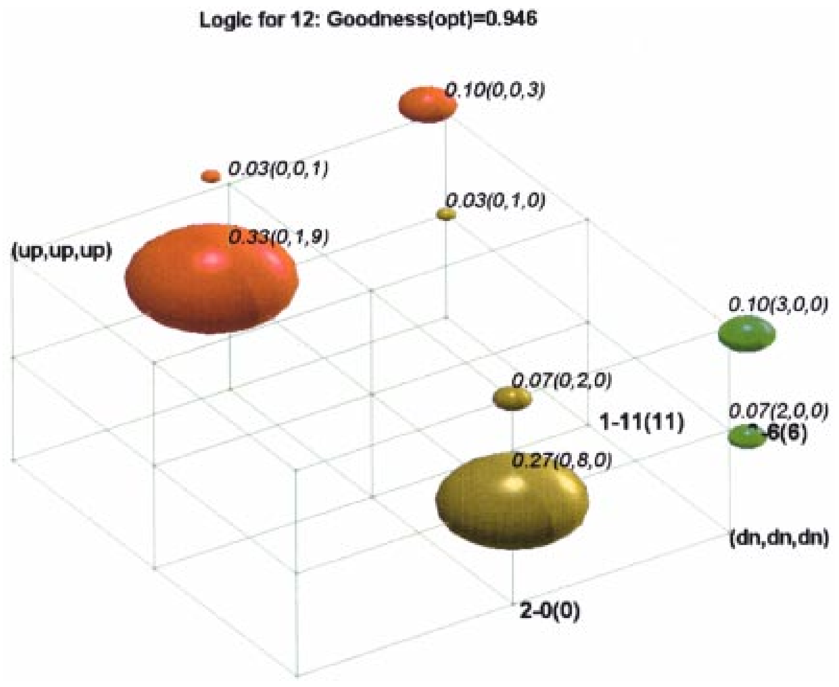


Fig. 11 Data graph.

The preceding statistical considerations indirectly apply to the coefficient of determination. The difficulty is that denominator in Eq. (7) is a sample-dependent random variable. Nonetheless, assuming that μ_Y can be precisely estimated from a relatively small sample, approximation is obtained by setting $\epsilon_{\mu,n}=\epsilon_\mu$. Assuming the validity of the approximation, we obtain the following inequalities: $E[\theta_n]\leqslant\theta_{\text{opt}}$, $E[\ddot{\theta}_n]\geqslant E[\theta_n]$, $E[\ddot{\theta}_n]\geqslant\theta_{\text{opt}}$, and $E[\hat{\theta}_n]=E[\theta_n]$.[17] Equations (10)–(12) can be rewritten to obtain bounds for the coefficient of determination.

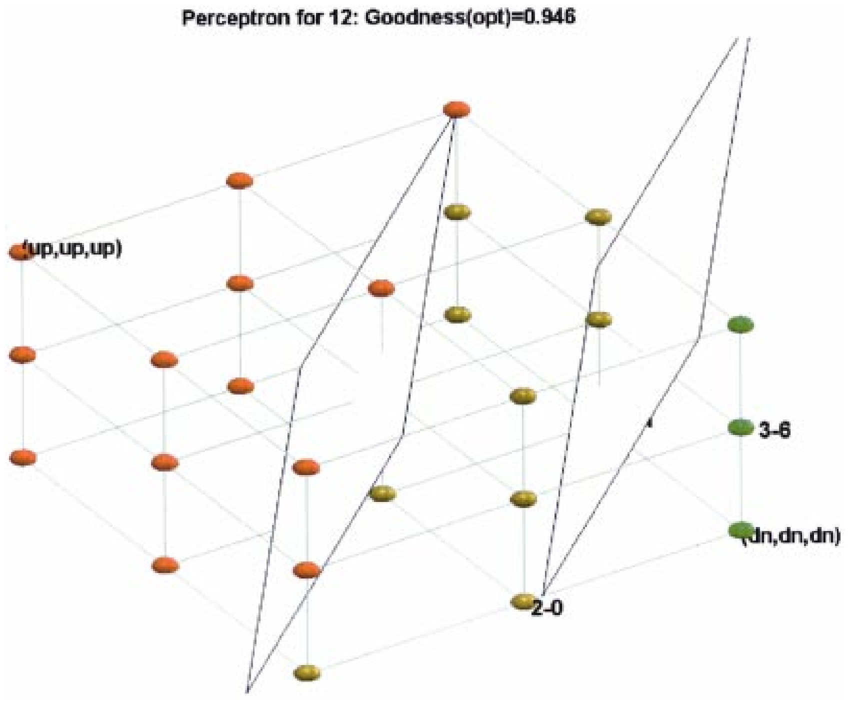


Fig. 12 Graphical display of perceptron.

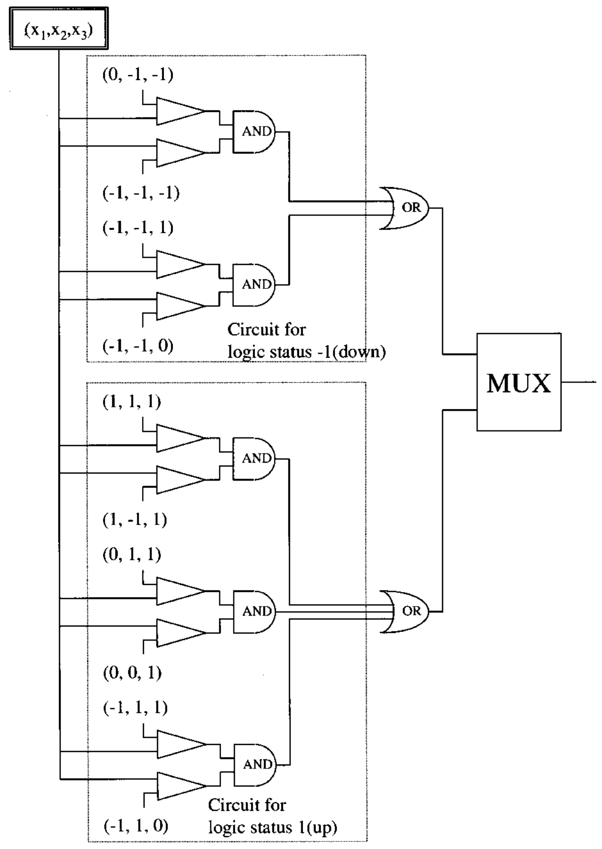


Fig. 13 Logic circuitry for perceptron.

7 Conclusion

New methods for simultaneously measuring expression levels of large numbers of genes motivate application of nonlinear multivariate expression analysis. This paper has demonstrated that multivariate nonlinear predictors (both unconstrained and constrained) can ferret out known and constructed relationships, and disclose common modes of transcriptional activity. Significant nonlinear prediction can be used to indicate potential pathway connections and such purely statistical relationships can be used to guide further investigations. The method has the flexibility to allow predictions to be formulated based not only on expression data, but also on the conditional functionality of genes and on applied external stimuli. Current technological and cost restrictions limit the number of microarray replicates; nonetheless, the degree of determination between predictor gene sets and target genes can be used in a comparative manner to discover potentially interesting control relations among the vast collection of all possible relationships.

To employ the proposed determination methodology for large classes of genes requires massively parallel computational capability. We are in the process of developing both the requisite hardware and software. Currently we can process all possible predictor sets containing three or less genes from among 600 genes, and can do this for 60 targets in less than a week. Work is continuing on computation, database management, software, and visualization tools. For the day when hundreds or thousands of replicates will be available, intelligent algorithms are being developed to allow consideration of more than three predictor genes without searching among all possible relationships. In addition, further research is necessary to develop ways to determine suitably constrained predictors so as not to miss multivariate gene relationships due to either lack of data (insufficient constraint) or lack of flexibility (too much constraint).

Appendix: Perceptron Training

Here in the Appendix we describe the design procedure for the ternary perceptron for the predictor expressions $X_1, X_2, \ldots, X_m$ and target expression Y, and we explain error estimation. Letting $\mathbf{X}=(X_1, X_2, \ldots, X_m)$ be the vector formed by the predictor variables, for the purpose of training we take a randomly selected sequence $(\mathbf{X}^1, Y^1), (\mathbf{X}^2, Y^2), \ldots,$ of training data (predictor vectors and target values) from the data. Due to a small data set, the training data are recycled using random reordering on each cycle. Letting $\mathbf{A}=(a_1, a_2, \ldots, a_n, b)$ be the perceptron coefficient vector, training involves positing an initial vector $\mathbf{A}_0$, applying $\mathbf{A}_0$ to the predictor data $\mathbf{X}^1$ to obtain an estimate $Y_{\text{pred},1}$ of Y, and using the error of this estimation to form a transition of $\mathbf{A}_0$ to $\mathbf{A}_1$. Iterating this estimation and transitioning based on the training data yields a sequence of coefficient vectors $\mathbf{A}_0 \rightarrow \mathbf{A}_1 \rightarrow \mathbf{A}_2 \rightarrow \ldots.$ Training is complete when the prediction error on the training data is zero or the change in $\mathbf{A}$ is insignificant. Predictor performance is then tested on a separate set of data pairs, $(\mathbf{X}, Y)$. For the purpose of training we use the absolute error $|Y_{\text{pred}} - Y|$.

Initialization of $\mathbf{A}$ can be either random or based on the training data. Using the methodology of linear prediction, we can initialize $\mathbf{A}$ based on empirical estimates of the cross-covariance vector $\mathbf{C}$ between $\mathbf{X}$ and Y and the autocorrelation matrix $\mathbf{R}$ of $\mathbf{V}=(1, X_1, X_2, \ldots, X_m)$. In this way, $\mathbf{A}_0 = \mathbf{R}^+\mathbf{C}$. From our experiments, prediction performance is not very sensitive to initialization; however, the initialization $\mathbf{R}^+\mathbf{C}$ sometimes yields better prediction and requires fewer iterations to converge. Transitioning of $\mathbf{A}$ can be done in two ways (as we now describe).

$\mathbf{A}$ can be transitioned following each sample, meaning that $\mathbf{A}$ is transitioned sequentially based on the prediction error for each training sample. After all samples have been used once, they are recycled in random order. Randomizing the training data before each cycle helps to mitigate any systematic bias.

The training algorithm is characterized by the update

$$\mathbf{A}^{(\text{new})} = \mathbf{A}^{(\text{old})} + \Delta\mathbf{A}, \tag{A1}$$

where the increment $\Delta\mathbf{A}$ is the transition size. Using the dot $(\bullet)$ product to represent the sum in Eq. (2) and inputting the next training sample $(\mathbf{X}, Y)$, we obtain the estimate $Y_{\text{pred}} = T[\mathbf{A}^{(\text{old})} \bullet \mathbf{V}]$ based on the current coefficient vector $\mathbf{A}^{(\text{old})}$. The training error is defined by

$$e = (\mathbf{A}^{(\text{old})} \bullet \mathbf{V} - Y)|Y_{\text{pred}} - Y|. \tag{A2}$$

$\mathbf{A}^{(\text{old})} \bullet \mathbf{V} - Y$ determines the magnitude (and sign) of the transition $\Delta\mathbf{A}$ and $|Y_{\text{pred}} - Y|$ gives the prediction error (relative to the training algorithm). Note that

1. $e > 0$ implies that we need to decrease $\mathbf{A}^{(\text{old})} \bullet \mathbf{V} - Y$ by transitioning $\mathbf{A}$;

2. $e<0$ implies that we need to increase $\mathbf{A}^{(\text{old})} \bullet \mathbf{V} - Y$ by transitioning **A**;
3. $e=0$ implies that there is no error and we do not transition **A**.

We use the perceptron transition $\Delta a_i = \alpha_i (Y - Y_{\text{pred}}) X_i$, where $0 < \alpha_i \leq 1$ are gain factors. The gain factors can be useful for proper training, although in our experiments prediction performance was not very sensitive to this choice.

The training algorithm for transitioning after each sample takes the following form: (1) initialize **A**; (2) feed in $\mathbf{X}^1, \mathbf{X}^2, \ldots,$ is some sequence, transitioning **A** at each step; (3) repeat step (2) using randomized recycling of inputs until a stopping criterion is reached. If **A** does not change during a cycle, this implies convergence of the training procedure (although it does not mean that the prediction error is 0 for each training sample). We can use any of the following stopping criteria: $\mathbf{A}_0 \rightarrow \mathbf{A}_1 \rightarrow \mathbf{A}_2 \rightarrow \ldots$ converges; a fixed number of iterations is chosen prior to training; or some minimum prediction error tolerance is reached.

Rather than transition after each sample, we may update only after the completion of a cycle of the training data. Under this protocol, e and Δa_i are computed following each sample as before; however, **A** is not transitioned after each sample. Instead, it is updated at the end of each cycle, with $\Delta\mathbf{A}$ being the sum of the stepwise increments during the cycle. Similar comments regarding the gain factors and stopping criteria apply.

Prior to applying the algorithm, it was extensively tested on simulated data generated by a mathematical model based on a thresholded linear operator corrupted by noise. Based on our experimental results, we implemented the algorithm on the microarray data using the initialization $\mathbf{A}_0 = \mathbf{R}^+ \mathbf{C}$ and transitioning **A** at the end of each cycle.

For data splitting, the perceptron is trained on the 20 training data sets and the errors of both the trained perceptron and the initialized perceptron are computed for the training data. The one that performs best on the training data is taken as the designed perceptron and is then applied to the 10 test sets to obtain a test error for the designed perceptron. This is repeated 256 times and the estimated error is taken as the average of these errors. Because of the small amount of test data and the stochastic nature of the training algorithm, the MSE of the designed predictor may be greater than the MSE of the thresholded mean. Since this would yield a negative coefficient of determination, which is impossible, the estimate is set to 0. Moreover, we apply the maximization discussed in the paper: if $m<r$ and the estimate gives a greater coefficient for m variables, then we take the maximum between $\hat{\theta}_n(r)$ and $\hat{\theta}_n(m)$ as the estimate of $\hat{\theta}_n(r)$.

References

1. J. DeRisi, L. Penland, P. O. Brown, M. L. Bittner, P. S. Meltzer, M. Ray, Y. Chen, Y. A. Su, and J. M. Trent, ''Use of a *c*DNA microarray to analyse gene expression patterns in human cancer [see comments],'' *Nature Genetics* **14**, 457–460 (1996).
2. J. L. DeRisi, V. R. Iyer, and P. O. Brown, ''Exploring the metabolic and genetic control of gene expression on a genomic scale,'' *Science* **278**, 680–686 (1997).
3. M. Schena, D. Shalon, R. W. Davis, and P. O. Brown, ''Quantitative monitoring of gene expression patterns with a complementary DNA microarray,'' *Science* **270**, 467–470 (1995).
4. M. Schena, D. Shalon, R. Heller, A. Chai, P. O. Brown, and R. W. Davis, ''Parallel human genome analysis: Microarray-based expression monitoring of 1000 genes,'' *Proc. Natl. Acad. Sci. U.S.A.* **93**, 10614–10619 (1996).
5. L. Wodicka, H. Dong, M. Mittmann, M. H. Ho, and D. J. Lockhart, ''Genome-wide expression monitoring in Saccharomyces cerevisiae,'' *Nat. Biotechnol.* **15**, 1359–1367 (1997).
6. G. Evan and T. Littlewood, ''A matter of life and cell death,'' *Science* **281**, 1317–1322 (1998).
7. H. H. McAdams and L. Shapiro, ''Circuit simulation of genetic networks,'' *Science* **269**, 650–656 (1995).
8. C. H. Yuh, H. Bolouri, and E. H. Davidson, ''Genomic *cis*-regulatory logic: Experimental and computational analysis of a sea urchin gene,'' *Science* **279**, 1896–1902 (1998).
9. S. Kim, E. R. Dougherty, M. L. Bittner, Y. Chen, K. Sivakumar, P. Meltzer, and J. M. Trent, ''Multivariate measurement of gene-expression relationships,'' *Genomics* **67**, 201–209 (2000).
10. Y. Chen, E. R. Dougherty, and M. L. Bittner, ''Ratio-based decisions and the quantitative analysis of *c*DNA microarray images,'' *J. Biomed. Opt.* **2**, 364–374 (1997).
11. C. M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, New York (1995).
12. F. Rosenblatt, *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*, Spartan, Washington (1962).
13. J. T. Astola and P. Kuosmanen, ''Representation and optimization of stack filters,'' in *Nonlinear Filters for Image Processing*, E. R. Dougherty and J. T. Astola, Eds., pp. 237–280, SPIE and IEEE, Bellingham, WA (1999).
14. L. Devroye, L. Gyorfi, and B. Lugosi, *A Probabilistic Theory of Pattern Recognition*, Springer, New York (1996).
15. E. R. Dougherty, *Random Processes for Image and Signal Processing*, SPIE and IEEE, Bellingham, WA (1999).
16. R. E. Walpole and R. H. Meyers, ''Probability and Statistics for Engineers and Scientists,'' 3rd ed., Macmillan, New York (1985).
17. E. R. Dougherty, S. Kim, and Y. Chen, ''Coefficient of determination in nonlinear signal processing,'' *Signal Processing* (in press).
18. S. A. Amundson, M. L. Bittner, Y. Chen, J. Trent, P. Meltzer, and A. J. Fornace, ''Fluorescent *c*DNA microarray hybridization reveals complexity and heterogeneity of cellular genotoxic stress responses,'' *Oncogene* **18**, 3666–3672 (1999).
19. M. Gorospe, S. Shack, K. Z. Guyton, D. Samid, and N. J. Holbrook, ''Up-regulation and functional role of *p*21 Waf1/Cip1 during growth arrest of human breast carcinoma MCF-7 cells by phenylacetate,'' *Cell Growth Differ* **7**, 1609–1615 (1996).
20. E. R. Dougherty and J. Barrera, ''Computational gray-scale operators,'' *Nonlinear Filters for Image Processing*, E. R. Dougherty and J. T. Astola, Eds., pp. 61–98, SPIE and IEEE, Bellingham, WA (1999).
21. M. Hills, ''Allocation rules and their error rates,'' *J. R. Statis. Soc.* **B28**, 1–31 (1966).
22. N. Glick, ''Sample-based multinomial classification,'' *Biometrics* **29**, 241–256 (1973).