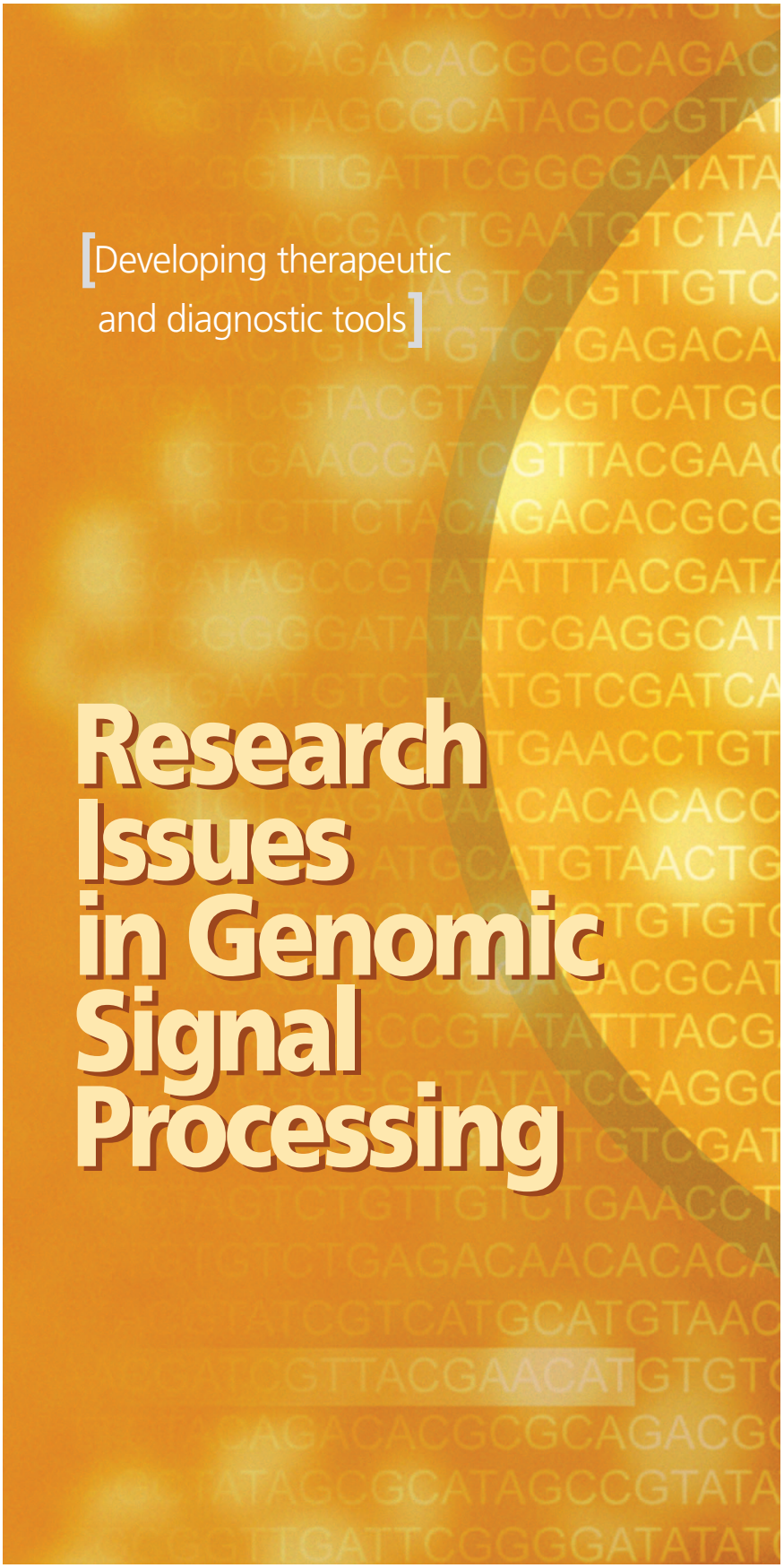
Edward R. Dougherty,
Aniruddha Datta, and Chao Sima



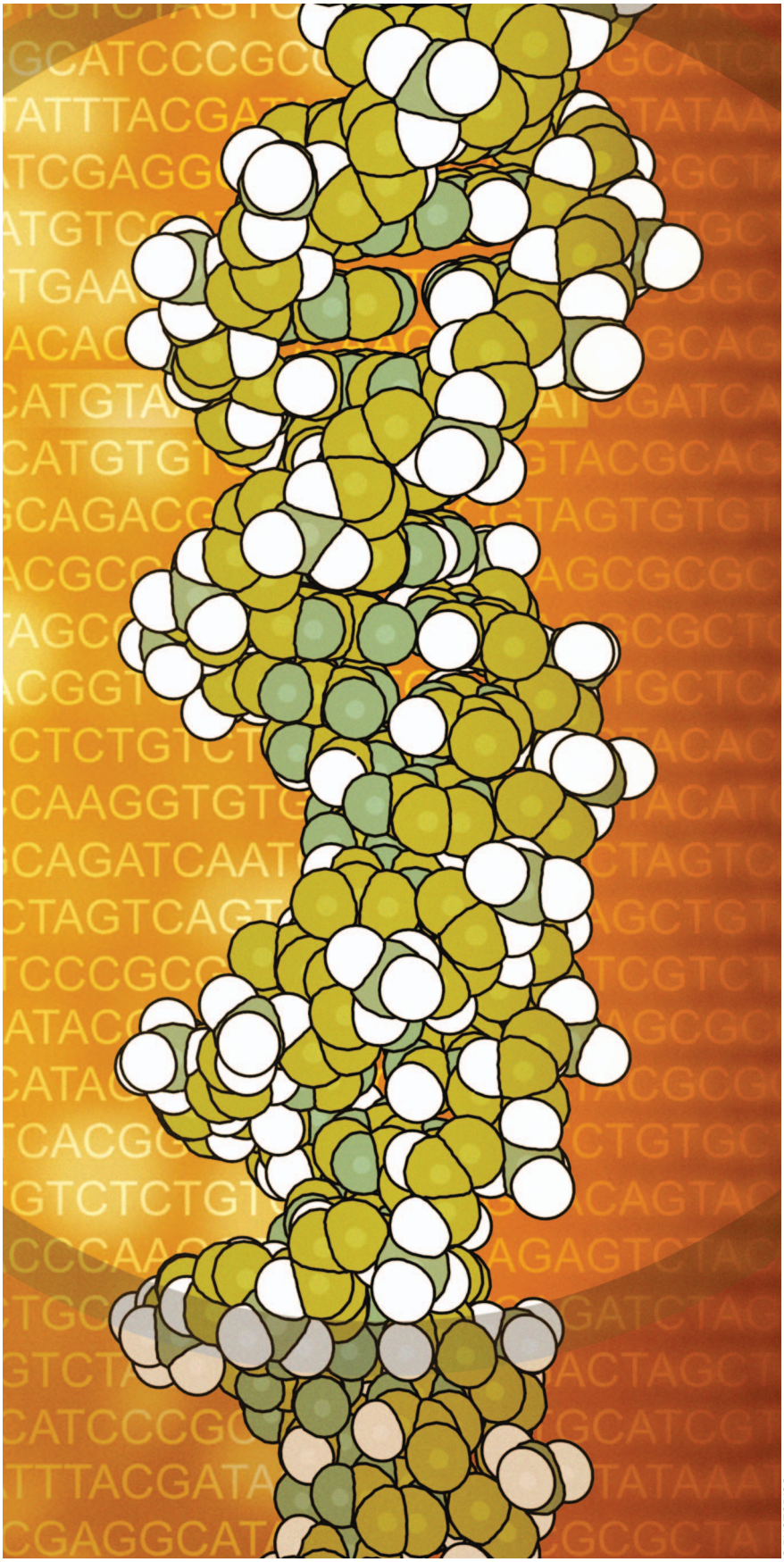
In an earlier article in *IEEE Signal Processing Magazine*, we discussed general issues regarding genomic signal processing (GSP) as it pertains to diagnosis and therapy [1]. In this article, we discuss the key research issues for GSP. It is important to recognize that "genomic signal processing" is not a name for genomic bioinformatics nor for the application of signal processing methods in genomics. GSP concerns the processing of genomic signals; it may be defined as the *analysis, processing, and use of genomic signals to gain biological knowledge and the translation of that knowledge into systems-based applications*. We note that research issues pertaining to GSP fit within the overall challenges confronting research in the area of multimodal biomedical systems [2].

We shall not review the basic biological concepts covered in our previous article except to recall two points. First, since cellular control results from multivariate activity among cohorts of genes and their products, it is not possible to separate the analyses of DNA, RNA, and protein in the DNA-to-RNA-to-protein information flow. Nevertheless, the immense interaction between levels ensures that a significant amount of the system information is available in each of the levels, with the current focus on RNA owing to measurement considerations, in particular, gene-expression microarrays. Second, two major goals of functional genomics are: 1) to use genomic signals to classify disease on a molecular level and 2) to screen for genes that determine specific cellular phenotypes and model their activity in such a way that normal and abnormal behavior can be differentiated. These goals correspond to diagnosing the presence or type of disease and developing therapies based on the disruption or mitigation of aberrant gene function contributing to the pathology of a disease. Developing diagnostic tools at the RNA level involves designing expression-based classifiers based on genes whose product abundances indicate key differences in cell state. Developing therapeutic tools involves synthesizing nonlinear dynamical networks, analyzing these networks to characterize gene regulation, and designing intervention strategies to modify dynamical behavior.

CLASSIFICATION

An expression-based classifier provides a list of genes whose product abundances are indicative of

1053-5888/05/$20.00©2005IEEE



© PHOTODISC

cellular differences and can, therefore, be used to discriminate among different phenotypes. There is a host of microarray-based classification papers in the literature, and we refer to the Web site http://gsp.tamu. edu/web2/cv_paper/pdf/refs.pdf for a large set of references. In medicine, classification relates to diagnostic issues such as the type of cancer [3] or the prognosis for patient survival [4]. The task is to design a classifier that takes a vector of gene expression levels as input and outputs a class label that predicts the class containing the input vector. A classifier is designed from a sample of expression vectors. This requires assessing expression levels from RNA obtained from different tissues, determining genes whose expression levels can be used as classifier features, and then applying a rule to design the classifier from the sample data. Design, performance evaluation, and application of classifiers must take into account randomness due to biological and experimental variability.

Figure 1 shows a graphical display in which each row corresponds to an expression profile for a particular gene across a sample of breast cancer patients, red and green indicating up and down regulation, respectively. The sample is partitioned into three subsets: a group of patients suffering from BRCA1 hereditary breast cancer, a group of patients suffering from BRCA2 hereditary breast cancer, and a third group of patients suffering from sporadic types of breast cancer. The objective is to design a classifier based on a set of genes that discriminates between BRCA1 and BRCA2 patients. The colors of the graphical display seem to indicate the possibility of successful classification. For instance, the first line corresponds to the gene KRT8, and the BRCA 1 expressions appear to be down-regulated in comparison to the BRCA2 expressions. Figure 2 shows the BRCA1 tumors linearly separated from the BRCA2 and sporadic tumors in the sample using genes KRT8 and DRPLA (atrophin-1). Pattern-recognition theory can be used to infer information from this separation regarding the general discrimination between the tumor classes. Classification using various methods has been used to exploit the class-separating power of expression data in many cancers.

The key research issues regarding expression-based classification are centered on the large number of features (genes) and the small number of sample points (microarrays). This disparity results in three critical issues:

- designing a classifier from sample data that provides good classification in general
- estimating the error of a designed classifier when data are limited

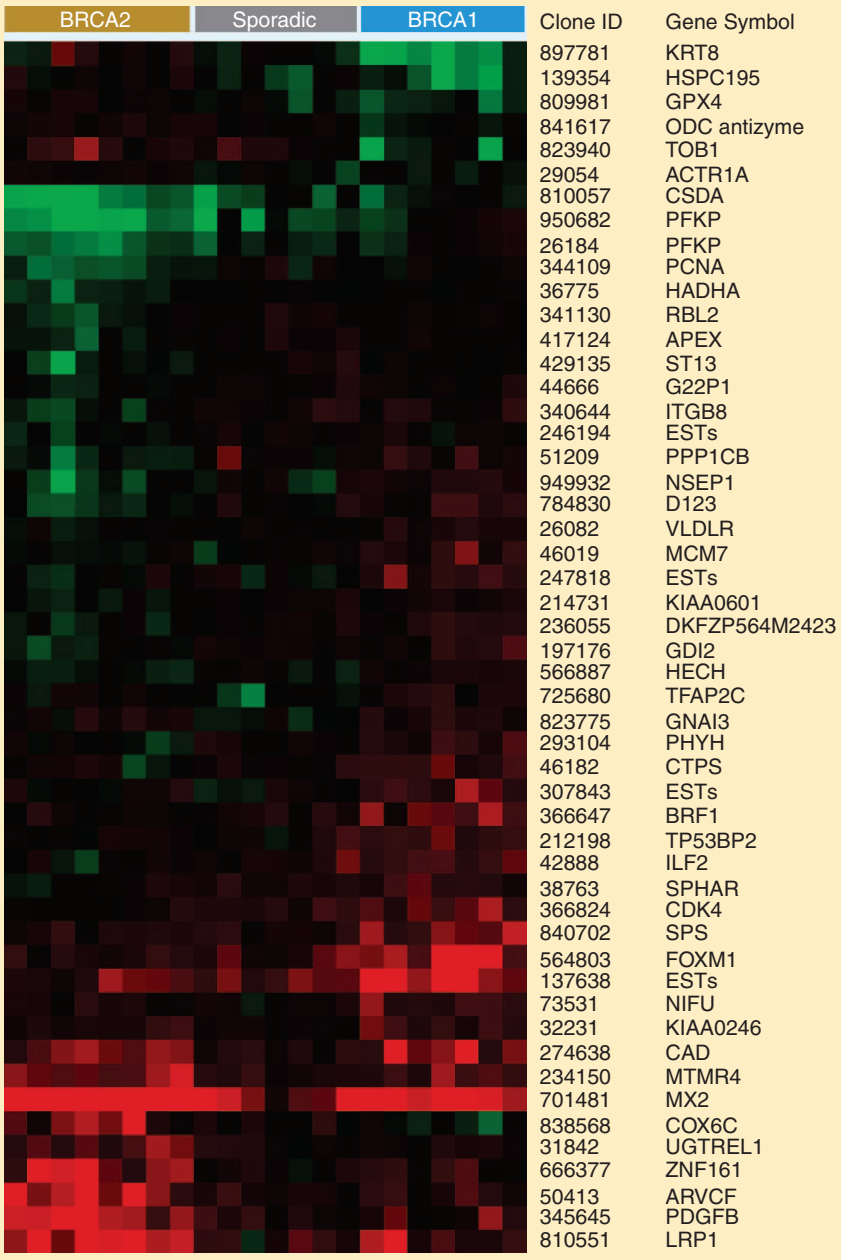
■ selecting features from a large set of potential features.

Small samples tend to result in imprecise classifier design, poor error estimation, and poor feature selection [5]. Substantial research is necessary in each of these areas.

CLASSIFIER DESIGN

The probabilistic theory of pattern classification is well developed [6]; here we sketch a bit of the theory so that we can discuss GSP research issues.

Classification involves a *feature vector* $\mathbf{X} = (X_1, X_2, \ldots, X_d)$ on d-dimensional Euclidean space $\mathcal{R}^d$ composed of random variables (features), a binary random variable Y, and a function (classifier) $\psi : \mathcal{R}^d \to \{0, 1\}$ to serve as a predictor of Y, which means that Y is to be predicted by $\psi(\mathbf{X})$. The values, zero or one, of Y are treated as class *labels*. The error, $\varepsilon[\psi]$, of ψ is the probability that the classification is erroneous, namely, $\varepsilon[\psi] = P(\psi(\mathbf{X}) \neq Y)$. $X_1, X_2, \ldots, X_d$ can be discrete or real valued. An optimal classifier ψ_d is one having minimal error ε_d among all binary functions on $\mathcal{R}^d$. ψ_d and ε_d are called the *Bayes classifier* and *Bayes error*, respectively. Classifier error depends on the probability distribution $f_{\mathbf{x},Y}(\mathbf{x}, y)$, called the *feature-label distribution*, of the feature-label pair $(\mathbf{X}, Y)$. Classification accuracy depends on the degree to which the *class conditional distributions*, $f_{\mathbf{x}|0}(\mathbf{x})$ and $f_{\mathbf{x}|1}(\mathbf{x})$, are separated. The Bayes classifier is given in terms of the label conditional probabilities by $\psi_d(\mathbf{x}) = 0$ if $P(Y = 1|\mathbf{x}) \leq 0.5$, and $\psi_d(x) = 1$ if $P(Y = 1|\mathbf{x}) > 0.5$. The Bayes error is expressed in a straightforward fashion as an integral. Since in practice we do not know the class conditional distributions, we must design a classifier from sample data. An obvious approach would be to estimate the feature-label distribution from the data, but there is rarely sufficient data for a good estimate. Nonetheless, good classifiers can be obtained even when we lack sufficient data for satisfactory distribution estimation.



[FIG1] Expression profiles for hereditary breast cancers: BRCA1, BRCA2, and sporadic.

Designing a classifier ψ_n from a random sample $S_n = \{(\mathbf{X}_1, Y_1), (\mathbf{X}_2, Y_2), \ldots, (\mathbf{X}_n, Y_n)\}$ of vector-label pairs drawn from the feature-label distribution requires a classification rule that operates on random samples to yield a classifier. A *classification rule* is a mapping of the form $\Psi_n : [\mathcal{R}^d \times \{0, 1\}]^n \to \mathcal{F}$, where $\mathcal{F}$ is the family of $\{0, 1\}$-valued functions on $\mathcal{R}^d$. Given a sample S_n, we obtain a designed classifier $\psi_n = \Psi_n(S_n)$ according to the classification rule. For a designed classifier ψ_n, there is a design cost $\Delta_n = \varepsilon_n - \varepsilon_d$, where ε_n and Δ_n are sample-dependent random variables. The expected design cost is $E[\Delta_n]$, the expectation being relative to all possible samples. The expected error of ψ_n is decomposed according to $E[\varepsilon_n] = \varepsilon_d + E[\Delta_n]$.

A key difficulty with small-sample design is that $E[\Delta_n]$ tends to be unacceptably large. A classification rule may yield a classifier that performs well on the sample data;

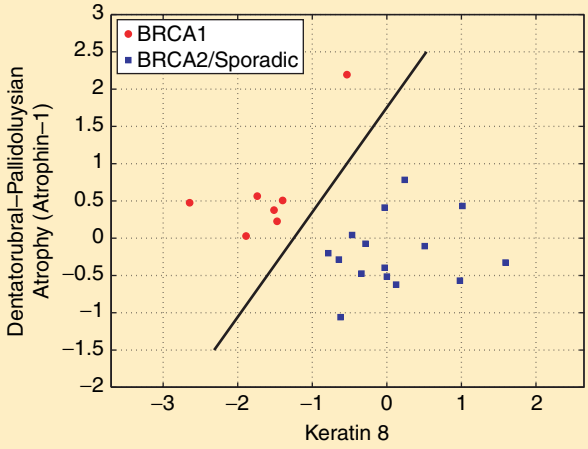
however, small samples do not generally represent the distribution sufficiently well to, on average, produce good classifiers. The result is *overfitting* of the sample data. Relative to the sample, the classifier possesses small error; but relative to the feature-label distribution, the error may be large. The problem is illustrated in Figure 3, where the *3-nearest-neighbor* (3*NN*) rule is applied to samples from two equal-variance circular Gaussian class conditional distributions. At each point $\mathbf{x}$, the 3NN classifier gives the majority value of the three nearest neighbors to $\mathbf{x}$. Figure 3(a) and (b) shows the 3NN classifier for two 30-point samples, and Figure 3(c) and (d) shows the 3NN classifier for two 90-point samples. Note the greater overfitting of the data for the 30-point samples. In particular, note the greater difference in the two 30-point designed classifiers as compared to the difference between the two 90-point classifiers and how close the latter are to the Bayes classifier given by the vertical line. The overfitting problem is not necessarily mitigated by applying an error-estimation rule to the designed classifier to see if it "actually" performs well; this is due to the fact that when only a small amount of data is available, error-estimation rules are very imprecise and this imprecision tends to be worse for complex classification rules. Hence, a low error estimate is not sufficient to overcome the large expected design error generated by using a complex classifier with a small data set. We need to consider classification rules that are constrained so as to reduce overfitting.

Constraining classifier design means restricting the functions from which a classifier can be chosen to a class C. This means trying to find an optimal *constrained* classifier, $\psi_C \in C$, having error ε_C. Constraining the classifier can reduce the expected design error, but at the cost of increasing the error of the best possible classifier. Since optimization in C is over a subclass of classifiers, the error ε_C of ψ_C will typically exceed the Bayes error, unless the Bayes classifier happens to be in C. This *cost of constraint* (*approximation*) is $\Delta_C = \varepsilon_C - \varepsilon_d$. A classification rule yields a classifier $\psi_{n,C} \in C$ with error $\varepsilon_{n,C}$, and $\varepsilon_{n,C} \geq \varepsilon_C \geq \varepsilon_d$. Design error for constrained classification is $\Delta_{n,C} = \varepsilon_{n,C} - \varepsilon_C$. For small samples, this can be substantially less than Δ_n, depending on the constraint and the rule. The error of the designed constrained classifier is decomposed as $\varepsilon_{n,C} = \varepsilon_d + \Delta_C + \Delta_{n,C}$. The expected error of the designed constrained classifier, which is our main concern, can be decomposed as

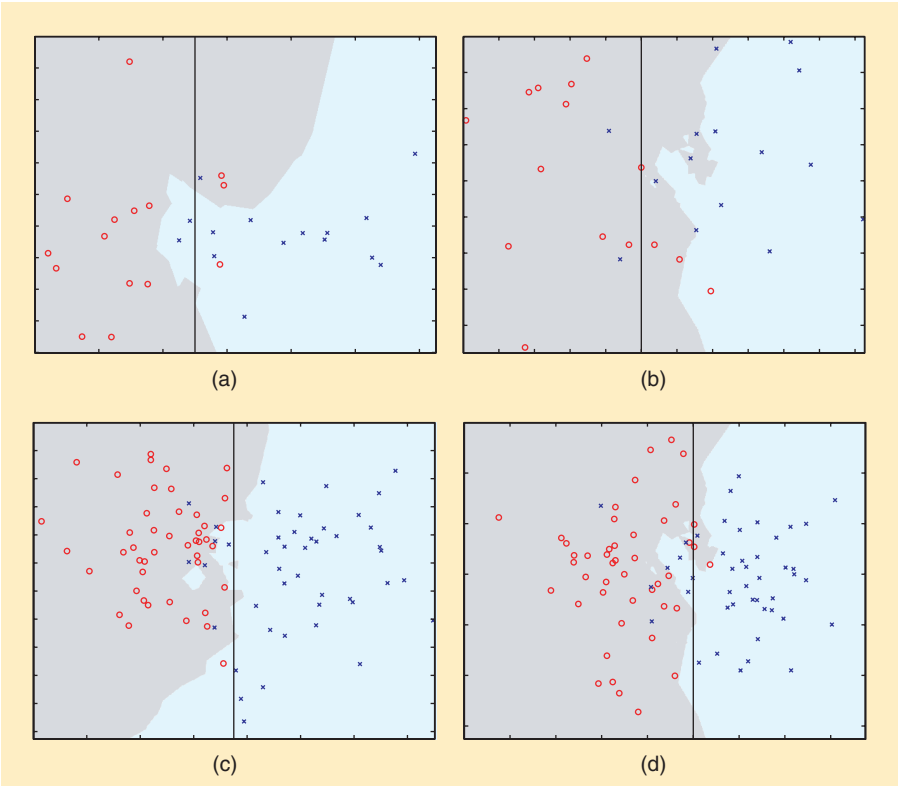
$$E[\varepsilon_{n,C}] = \varepsilon_d + \Delta_C + E[\Delta_{n,C}].$$

The constraint is beneficial if and only if the cost of constraint is less than the decrease in expected design cost. The dilemma: strong constraint reduces $E[\Delta_{n,C}]$ at the cost of increasing ε_C.

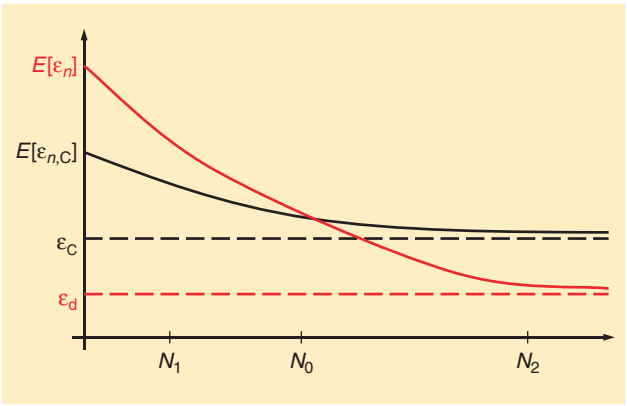
For a graphical illustration, consider a classification rule for which the expected design error never increases as sample



[FIG2] **Linear classification of BRCA1 from BRCA2 and sporadic cancers using two genes.**



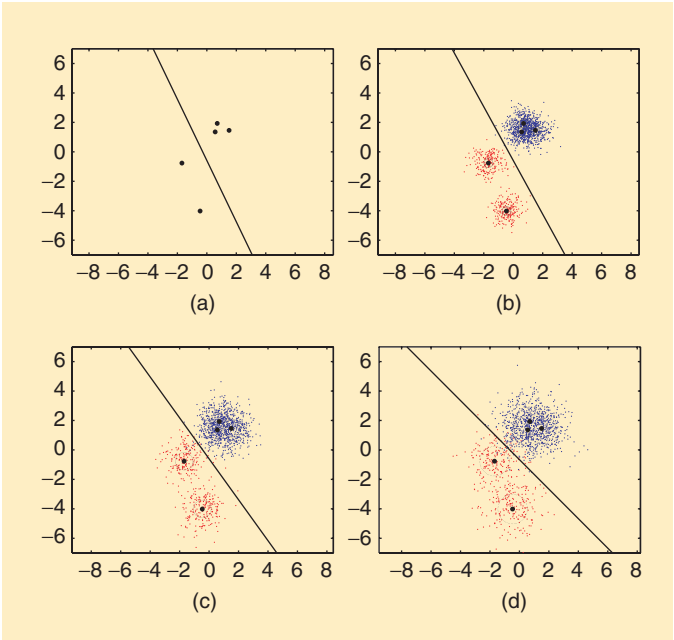
[FIG3] **3NN classification applied to two equal-variance circular Gaussian class conditional distributions: (a) for a 30-point sample, (b) for a second 30-point sample, (c) for a 90-point sample, and (d) for a second 90-point sample.**



[FIG4] Expected error for unconstrained and constrained classification as a function of sample size.

sizes increase: $E[\Delta_{n+1}] \leq E[\Delta_n]$ and $E[\Delta_{n+1,C}] \leq E[\Delta_{n,C}]$. In Figure 4, the axes correspond to sample size and error. The horizontal dashed lines represent ε_C and ε_d, and the decreasing solid lines represent $E[\varepsilon_{n,C}]$ and $E[\varepsilon_n]$. If n is sufficiently large, then $E[\varepsilon_n] < E[\varepsilon_{n,C}]$; however, if n is small, then $E[\varepsilon_n] > E[\varepsilon_{n,C}]$. The point N_0 at which the solid lines cross is the cut-off: for $n > N_0$, the constraint is detrimental; for $n < N_0$, it is beneficial.

Much effort has gone into bounding $E[\Delta_{n,C}]$. A celebrated theorem of pattern recognition provides bounds for $E[\Delta_{n,C}]$ for the *empirical-error rule*, which chooses the classifier in C that makes the least number of errors on the sample data [7], [8]. Classical bounds are of the form $E[\Delta_{n,C}] \leq \lambda_C \sqrt{\log n / n}$, where λ_C is related to the complexity of the classifiers in C. Greater classifier complexity corresponds to a greater ability to cut up the feature space, and λ_C is larger for greater complexity. Unfortunately, such bounds tend to lack practical import for GSP because they require large samples to make them useful. For instance, in the bound just cited, for a neural network with ten inputs and ten neurons, the bound exceeds 1 for $n = 5{,}000$.



[FIG5] Effect of noise injection on classifier design. (a) No spreading. (b) Spreading with $\sigma = 0.3$. (c) Spreading with $\sigma = 0.6$. (d) Spreading with $\sigma = 1.0$.

REGULARIZATION

Rather than design a classifier precisely according to some classification rule when the sample is small, it can be beneficial to *regularize* the sample data or the parameters estimated from the data; by regularization we mean some alteration of the data or modification of the estimation rule for the parameters.

Linear classification rules yield a decision boundary that is a hyperplane in the feature space. Owing to their low complexity, they tend to require less data for design and error estimation. Numerous classification rules yield linear decisions, and they can have very different properties. We confine ourselves to linear discriminant analysis. If the class conditional densities are Gaussian and equally likely, then the Bayes classifier is determined by a discriminant $Q(\mathbf{x})$ involving the covariance matrices and means for the classes, with $\psi(\mathbf{x}) = 1$ if and only if $Q(\mathbf{x}) > 0$. The decision boundary is quadratic, thereby leading to the name *quadratic discriminant analysis (QDA)*. If the classes possess a common covariance matrix, then the decision boundary is a hyperplane, and the method is called *linear discriminant analysis* (*LDA*). In practice, QDA and LDA are applied by estimating the covariance matrices and mean vectors with the sample covariance matrices and sample means. Application does not require Gaussian class conditional densities; however, better performance is expected when they are close to Gaussian. Relative to QDA, a simple regularization is to apply LDA when the covariance matrices are not equal. This means estimating a single covariance matrix by pooling the data. This reduces the number of parameters to be estimated and increases the sample size relative to the smaller set of parameters.

A softer approach than strictly going from QDA to LDA is to shrink the individual covariance estimates in the direction of the pooled estimate. This can be accomplished by introducing a parameter α between zero and one to arrive at weighted covariance matrices for which QDA results from $\alpha = 0$ and LDA from $\alpha = 1$, with different amounts of shrinkage occurring for $0 < \alpha < 1$ [9]. To get more regularization while producing little bias, one can shrink the regularized sample covariance matrix towards the identity multiplied by its average eigenvalue [10]. This has the effect of decreasing large eigenvalues and increasing small eigenvalues, which offsets the phenomenon that large eigenvalues of the sample covariance matrix are biased high and small eigenvalues are biased low, a situation that is accentuated for small samples.

A general procedure is to regularize the data itself through *noise injection*. This can be done by "spreading" the sample data by generating synthetic data about each sample point. This creates a large synthetic sample from which to design the classifier while at the same time making the designed classifier less dependent on the specific points in

the small data set. For instance, one may place a circular Gaussian distribution at each sample point, randomly generate points from each such distribution, and then apply a classification rule. This approach has been extensively examined relative to LDA [11]. An immediate advantage is that noise injection can be used in strictly data-driven iterative classifier designs. A spherical distribution need not be employed; indeed, it has been demonstrated that it can be advantageous to base noise injection at a sample point based on the nearest neighbors of the point [11]. Figure 5 illustrates noise injection with different spreads for LDA. The procedure can be posed analytically in terms of matrix operations for linear classification, and this is critical for situations in which a large number of feature sets must be examined, as is often the case for microarray-based classification [12]. Noise injection can take a different form in which the sample data points themselves are perturbed by additive noise rather than new synthetic points being generated. This approach has been used in designing neural networks, in particular, where owing to a small sample, the same data points are used repeatedly [13], [14]. Research is required to better understand the effects of noise injection and gain an appreciation of how it should be applied in various circumstances. An important question to answer is: What should be the distribution of the noise, in particular, its variance and shape?

Since increasing classifier complexity tends to result in increased design cost, rather than simply applying estimated errors to rank classifiers, one can penalize classifiers by adding a complexity term $\rho(n)$ to arrive at a new *penalized error* that is a sum of the error and a complexity penalty $\hat{\varepsilon}_n[\psi] + \rho(n)$. The idea is to achieve *complexity regularization* relative to the classifier class by having a measure that incorporates both error and classifier complexity. Put another way, from a collection of classifier models, we want to choose a model most suited for the amount of data. One approach is *structural risk minimization*, in which, relative to a sequence of constrained classes, a classifier is chosen from each class by minimizing the estimated error on the sample data and then choosing among these the one possessing minimal penalized error, where in each case the penalty is relative to the class containing the classifier [8], [15]. There are practical difficulties involved because some empirical measure must be employed [16], [17]. Another approach is to use the *minimum-description-length* (MDL) principle, which says that, given the data and a class of models, select the model that achieves the shortest code length for the data and model [18]. Using the MDL principle, complexity regularization can be achieved by replacing error minimization with minimization of a sum of code lengths—one relative to encoding the error and the other relative to encoding the classifier description—in an effort to balance increased error and increased model complexity. It has been used in the context of gene prediction, which is important to regulatory networks [19].

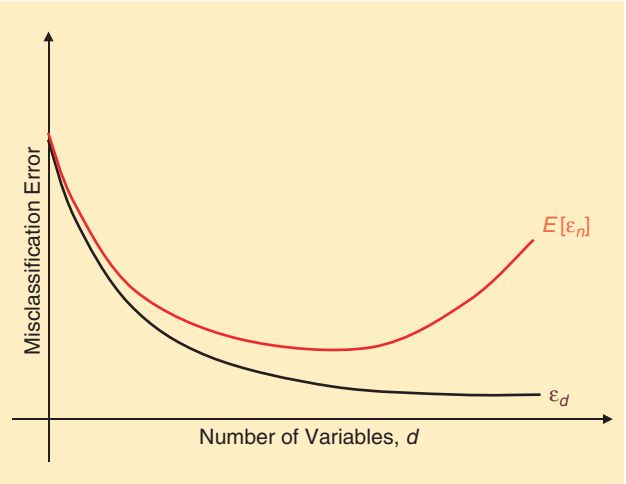
There is active research in complexity regularization, including application to genomic data. Since complexity regularization can be used for network design (relative to network complexity), it holds promise for genomic modeling, especially since error estimation is so problematic.

FEATURE SELECTION

Owing to the thousands of probes on a microarray, each yielding an expression measurement that is a potential feature for classification, massive feature reduction is an indispensable aspect of expression-based classification. Due to the lack of monotonicity for estimated errors, one cannot simply apply some naïve approach and select a large number of features, assuming the more the better. The Bayes error is monotone since, if A and B are feature sets with Bayes errors ε_A and ε_B, respectively, and $A \subset B$, then $\varepsilon_B \leq \varepsilon_A$. However, if $\varepsilon_{A,n}$ and $\varepsilon_{B,n}$ are the corresponding errors resulting from designed classifiers on a sample of size n, then $E[\varepsilon_{B,n}]$ may exceed $E[\varepsilon_{A,n}]$. Indeed, it is commonplace for the expected design error to decrease and then increase for increasingly large feature sets. This is called the *peaking phenomenon* [20], [21]. It is illustrated in Figure 6, where the horizontal axis corresponds to a sequence of features, $x_1, x_2, \ldots, x_d, \ldots$, and the vertical axis gives the error of the optimal classifier for the given features. As indicated, for d features, the Bayes error ε_d continues to decline, but the expected error, $E[\varepsilon_{d,n}]$, of the designed classifier goes down and then begins to rise.

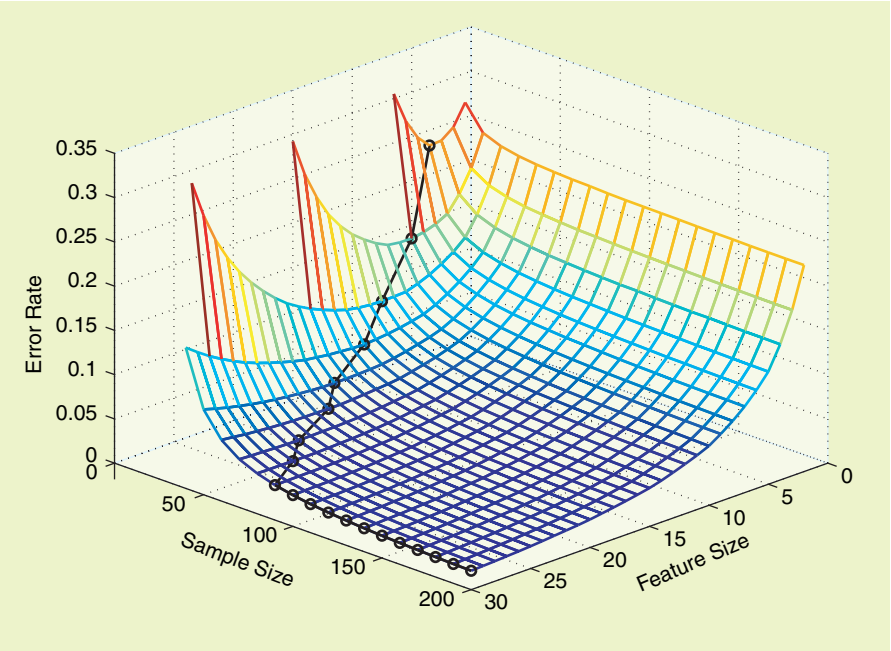
A major stumbling block is the combinatorial nature of feature selection. To select a subset of k features from a set of n potential features and be assured that it provides an optimal classifier with minimum error among all optimal classifiers for subsets of size k, all k-element subsets must be checked unless there is distributional knowledge that mitigates the search requirement, a condition rarely satisfied in practice [22].

In light of the peaking phenomenon, given a set of features, what is the optimal number of features? The question is complicated because it depends on the classification rule, feature-label distribution, and sample size. Figure 7 illustrates peaking in terms of sample size n and the number d of features. The surface gives the average error of designed LDA classifiers in terms of d and n based on two Gaussian class conditional distributions possessing the same covariance matrix. The features are slightly correlated, and we see that peaking occurs with very few features for

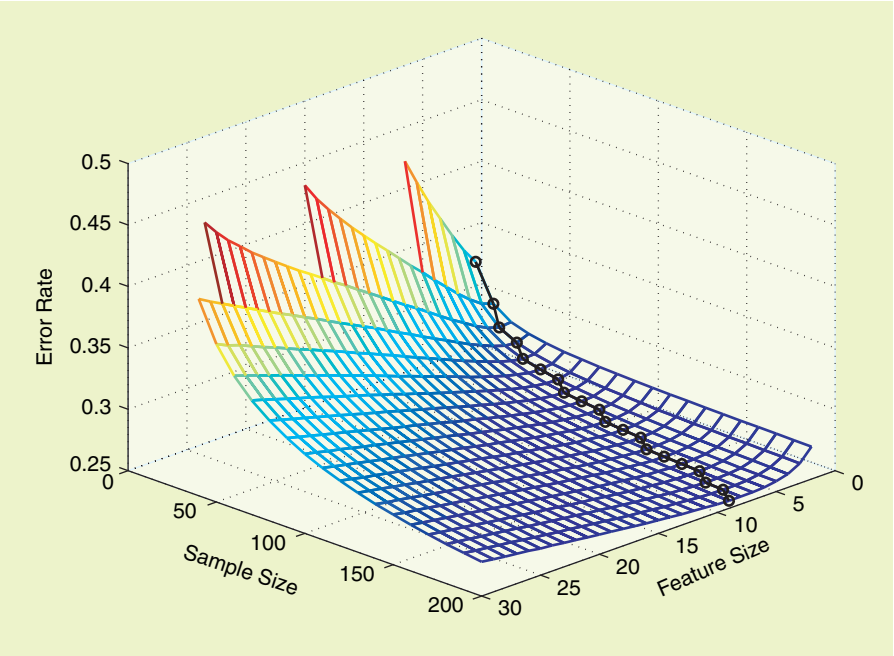


[FIG6] Peaking phenomenon: expected error and Bayes error as functions of the number of features.

sample sizes 30 and below, but then exceeds 30 features for sample sizes above 90. Matters are different in Figure 8, where the situation is the same except that the features are highly correlated. Here, even with a sample size of 200, the optimal number of features is only eight. The behavior in Figures 7 and 8 corresponds to the usual understanding of the peaking phenomenon. Figure 9 should make one wary of hasty generalizations. It shows the same error averages for designed 3NN classifiers for two Gaussian class conditional distributions possessing different covariance matrices, in which case the optimal classifier is quadratic. Here, the optimal feature size first decreases for small samples and then remains essentially constant (ignoring the small wiggles owing to simulation). The optimal number does not go beyond where it is at sample size 100 until the sample size is in the thousands. A large simulation study provides optimal feature surfaces for a number of classification rules and feature-label distributions [23]. Although we will introduce some analytic work concerning the optimal number of features for QDA, generally there is little understanding regarding peaking and very little mathematical analysis that addresses the topic. Given its importance for small-sample classification, much investigation is warranted.



[FIG7] **Peaking phenomenon for LDA with slightly correlated features.**



[FIG8] **Peaking phenomenon for LDA with highly correlated features.**

In our preceding discussion on the optimal number of features, we knew the distributions and were able to order the features so as not to have to consider all possible feature sets. In practice, there is typically no way around the combinatorial problem, and one has to resort to a feature-selection algorithm. In principle, a full exhaustive search can be mitigated by using a branch and bound feature-selection algorithm that takes advantage of the monotonicity property to obtain an optimal solution [24]. However, worst-case performance can be exponentially complex and error estimation must be used, thereby losing monotonicity, a problem exacerbated by small samples. Suboptimal approaches need to be considered. The most obvious approach is to consider each feature by itself and choose the k features that perform individually the best or perhaps choosing those most correlated with the classes. While easy, this method is subject to choosing a feature set with a large number of redundant random variables. Also, it suffers because features that perform poorly individually may do well in combination.

Feature selection is often split into two categories: the filter and wrapper methods. In the *filter method*, features are selected without regard for classifier design, for instance, by choosing features most correlated with the labels or via mutual information. In the *wrapper method*, features are selected in conjunction with classifier design. When there is a very

large number of features, such as in the case of gene expressions on a microarray, the methods are used in conjunction. First, a filtering method is used and then some selection method involving classification is employed on the preliminarily reduced set. When using a filter method, there is the danger of selecting many redundant features and also missing features that perform poorly in isolation but work well in combination.

A common approach to suboptimal feature selection is sequential selection, either forward or backward, and their variants. *Sequential forward selection* (*SFS*) begins with a small set of features (perhaps one), and iteratively builds the feature set. Where there are k features, $x_1, x_2, \ldots, x_k$, in the growing feature set, all feature sets of the form $\{x_1, x_2, \ldots, x_k, w, \}$ are compared and the best one is chosen to form the feature set of size $k+1$. A problem with SFS is that there is no way to delete a feature adjoined early in the iteration that may not perform as well in combination as other features. The SFS look-back algorithm aims to mitigate this problem by enabling deletion. When there are k features, $x_1, x_2, \ldots, x_k$, in the growing feature set, all feature sets of the form $\{x_1, x_2, \ldots, x_k, w, z\}$ are compared and the best one is chosen. Then all $(k+1)$-element subsets are checked to allow the possibility of deleting one of the earlier chosen features, the result being the $k+1$ features that will form the basis for the next stage of the algorithm. Flexibility can be added by considering *sequential forward floating selection* (*SFFS*), where the number of features to be adjoined and deleted is not fixed but is allowed to "float" [25].

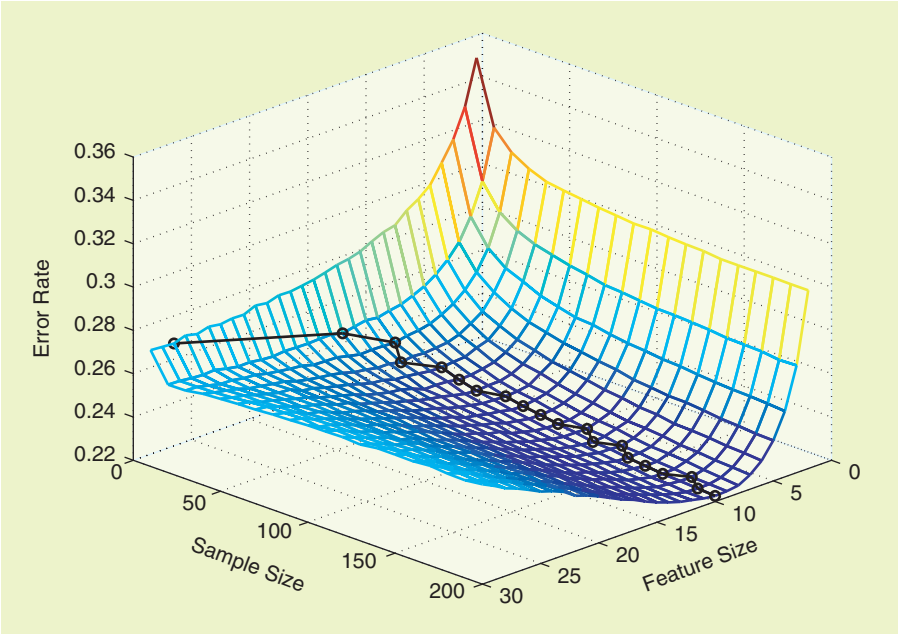
For a large number of potential features, feature selection is problematic, and the best method depends on the circumstances. Evaluation of methods is generally comparative and based on simulations; SFFS appears to be the best in practice [26], [27]. Among the key research issues for feature-selection algorithms are mathematical analyses of their performances, ground-truth comparisons, their behavior for different classifiers and feature-label distributions, the impact of error estimation on error-based decisions within the algorithm (which we will discuss shortly and is very important in small-sample settings), and validation (Does an algorithm outperform SFFS?). Owing to the importance of feature selection in selecting genes for classification and for prediction in the design of regulatory networks, this is a topic that deserves serious attention. By this we do not mean the proposal of more algorithms based on unsubstantiated heuristics. What is required is careful work on both the construction and analysis of feature-selection procedures, taking into account the problems caused by extremely large numbers of features and very small samples.

When considering feature selection, it must be recognized that feature selection is part of the classification rule and, as such, it is a form of complexity regularization. For instance, in the case of gene selection, a normalized maximum likelihood method has been proposed for feature selection in Boolean models which involves restating the classification problem as a modeling problem in terms of a class of parametric models [28]. The method is related to the MDL principle and has immediate application in model selection for gene regulatory networks. Since feature selection is part of the classification rule, if one has D features and chooses d of them, then the classification rule is being applied to D-dimensional space and this must be taken into account when treating the complexity of classifier design. Moreover, if an error estimation method relates to the classification rule, then one must incorporate feature selection into the error-estimation procedure (see cross-validation error estimation).

ERROR ESTIMATION

Error estimation is critical for classification because the error of a classifier determines its worth. Thus, the precision of error estimation is extremely important. The issue is complicated because the precision of an error estimator depends on the classification rule, feature-label distribution, number of features, and sample size. Error estimation is problematic for GSP owing to performance degradation with small samples. Achieving better small-sample error estimation is a major GSP research issue.

If a classifier ψ_n is designed from a particular sample of size n, then the error of the classifier relative to the sample is given by $\varepsilon_n = E[|Y - \psi_n(\mathbf{X})|]$, where the expectation is taken relative to the feature-label distribution. In practice, the feature-label distribution is unknown and the error must be estimated. If there is an abundance of sample data, then it can be split



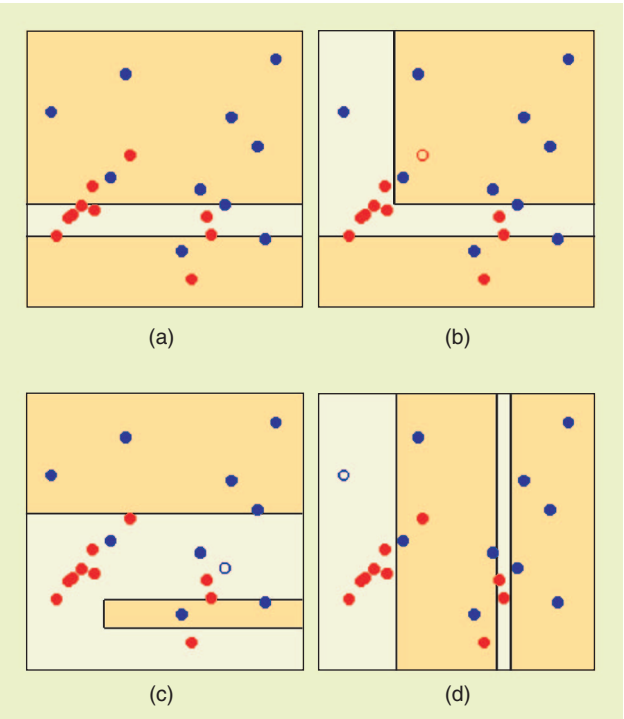
[FIG9] **Peaking phenomenon for 3NN.**

into *training* and *test* data. A classifier is designed on the training data, and its estimated error is the proportion of errors it makes on the test data. We do not consider this approach because holding out data with a small sample leaves less data available for design, thereby resulting in a less effective design. We only consider error estimators based on the sample from which the classifier is designed.

One approach is to design a classifier ψ_n from the sample and estimate ε_n by applying ψ_n to the sample. The *resubstitution estimate*, ε_n^{res}, is the fraction of errors made by ψ_n on the sample. The resubstitution estimator is typically low biased, meaning $E[\varepsilon_n^{res}] < E[\varepsilon_n]$, and this bias can be severe for small samples depending on the complexity of the classification rule.

Cross-validation is a resampling strategy in which classifiers are designed from parts of the sample, each is tested on the remaining data, and ε_n is estimated by averaging the errors. In *k-fold cross-validation*, the sample S_n is partitioned into k folds $S_{(i)}$, for $i = 1, 2, \ldots, k$. Each fold is left out of the design process and used as a test set, and the estimate $\varepsilon_n^{cv(k)}$ is the average error committed on all folds. A k-fold cross-validation estimator is unbiased as an estimator of $E[\varepsilon_{n(n/k)}]$, meaning $E[\varepsilon_n^{cv(k)}] = E[\varepsilon_{n-n/k}]$, where $\varepsilon_{n-n/k}$ is the error arising from design on a sample of size $n - n/k$. The special case of n-fold cross-validation yields the *leave-one-out estimator*, $\hat{\varepsilon}_n^{loo}$, which is an unbiased estimator of $E[\varepsilon_{n-1}]$.

Cross validation is likely the most popular error estimator in the microarray literature. Unfortunately, it is often used with neither justification nor mention of its serious shortcomings with small samples. While not suffering from severe bias, cross-validation has large variance in small-sample settings; therefore, its use is problematic [6]. Focusing on the unbiasedness of leave-one-out estimation, $E[\hat{\varepsilon}_n^{loo}] = E[\varepsilon_{n-1}]$, so that $E[\hat{\varepsilon}_n^{loo} - \varepsilon_n] \approx 0$. Thus, the expected difference between the error estimator and the error is approximately zero. But we are not interested in the expected difference between the error estimator and the error; rather, we are interested in the precision of the error estimator in estimating the error. Our concern is the expected deviation, $E[|\hat{\varepsilon}_n^{loo} - \varepsilon_n|]$, and unless the cross-validation variance is small, (which it is not for small samples), this expected deviation will not be small. By considering the deviation distributions for several error estimators, it is seen that cross-validation generally performs poorly for small samples [29].



[FIG10] **Original and several surrogate classifiers for CART.**

The difficulty with cross-validation is illustrated in Figure 10, which shows decision regions obtained from a sample using the method of *classification and regression trees* (CART). This is a classification-tree approach that iteratively partitions the feature space by perpendicular splits based on an "impurity function" and defines the resulting classifier on each leaf (cell) of the partition by majority vote. Figure 10(a) shows the classifier designed from the sample data, and Figure 10(b)–(d) shows a few *surrogate* classifiers designed from the sample after a point (circle) has been removed. Note how different the surrogates are from the designed classifier. The leave-one-out estimator is obtained from averaging the resubstitution errors of all the surrogates, many of which have little relation to the actual classifier whose error we desire. Although cross validation does not perform well for small samples, it can be beneficial for modest-sized samples in which one wishes to use all of the data for design so as to obtain the best possible classifier. For larger samples, the variances of the cross-validation estimators get smaller and better performance is achieved. Finally, recalling our discussion of feature selection, since cross validation estimates the expected error owing to the classification rule, feature selection must be repeatedly done each time points are left out in the estimation procedure.

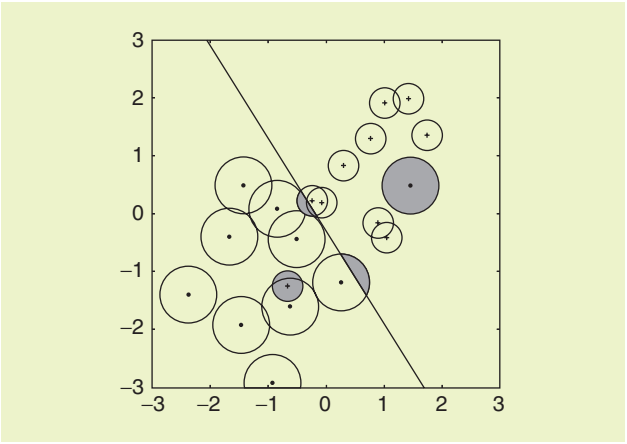
Bootstrap is a general resampling strategy that can be applied to error estimation [30]. A bootstrap sample consists of n equally likely draws with replacement from the original sample S_n. Some points may appear multiple times, whereas others may not appear at all. For the basic bootstrap estimator, $\hat{\varepsilon}_n^b$, the classifier is designed on the bootstrap sample and tested on the points left out. This is done repeatedly, and the bootstrap estimate is the average error made on the left-out points. $\hat{\varepsilon}_n^b$ tends to be a high-biased estimator of $E[\varepsilon_n]$, since the number of points available for design is on average only $0.632n$. The *.632 bootstrap estimator* tries to correct this bias via a weighted average of $\hat{\varepsilon}_n^b$ and resubstitution [31]

$$\hat{\varepsilon}_n^{b632} = 0.368\varepsilon_n^{res} + 0.632\hat{\varepsilon}_n^b.$$

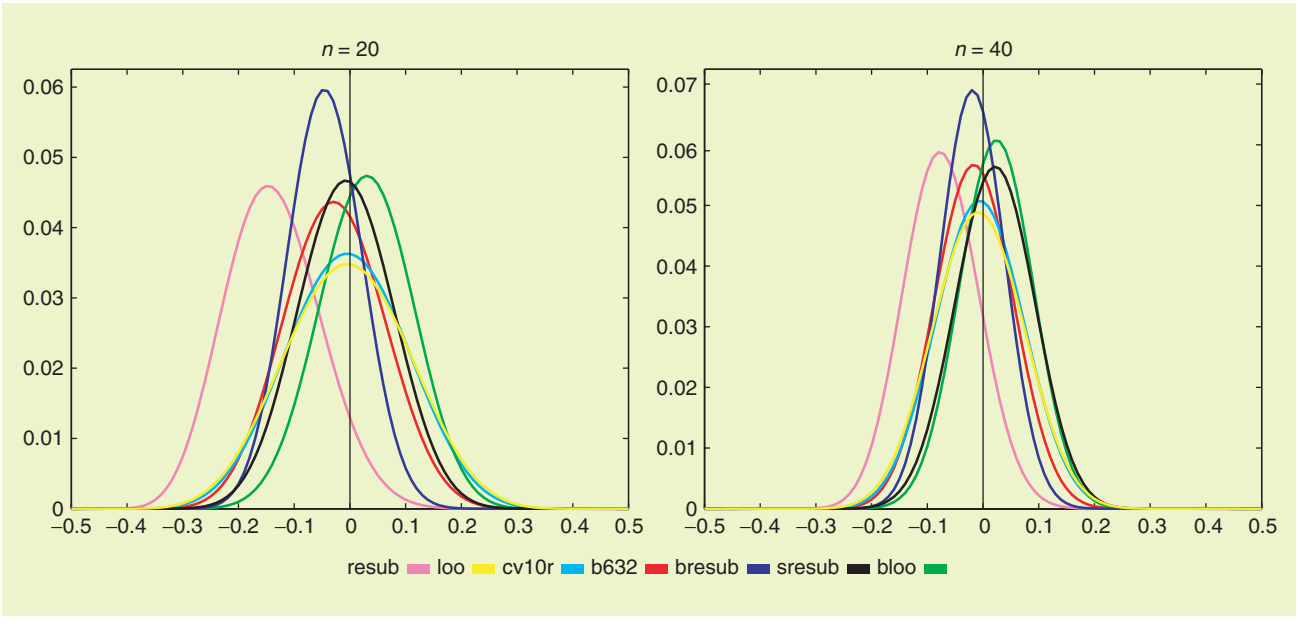
In resubstitution, there is no distinction between points close to and far from the decision boundary. The *bolstered-resubstitution* estimator is based on the heuristic that, relative to making an error, more confidence should be attributed to points far from the decision boundary than points close to it

[32]. This is achieved by placing a distribution, called a *bolstering kernel*, at each point and estimating the error by integrating each bolstering kernel over the decision region for which the point does not properly belong and then averaging the integrals. A key issue is the amount of bolstering (spread of the bolstering kernels), and a method has been proposed to compute this spread based on the data. Figure 11 illustrates the error for linear classification when the bolstering kernels are uniform circular distributions. When resubstitution is heavily low-biased, it may not be good to spread incorrectly classified data points because that increases the optimism of the error estimate (low bias). The *semibolstered-resubstitution* estimator results from not bolstering (no spread) for incorrectly classified points. Bolstering can be applied to any error-counting estimation procedure. *Bolstered leave-one-out* estimation involves bolstering the resubstitution estimates on the surrogate classifiers.

To demonstrate small-sample error-estimator performance, we provide simulation results for the empirical distribution of $\hat{\varepsilon}_n - \varepsilon_n$, in which the error estimator $\hat{\varepsilon}_n$ is one of the following: resubstitution (resub), leave-one-out (loo), ten-fold cross-validation with ten repetitions (cv10r), .632 bootstrap (b632), bolstered resubstitution (bresub), semibolstered resubstitution (sresub), or bolstered leave-one-out (bloo). Bolstering utilizes Gaussian bolstering kernels. The simulations use data from a study that analyzes a large number of microarrays prepared with RNA from breast tumor samples from each of 295 patients [4]. Of the 295 microarrays, 115 belong to the "good-prognosis" class and 180 belong to the "poor-prognosis" class. The simulations use log-ratio gene-expression values associated with the top five genes, as ranked by a correlation-based measure. For each case, 1,000 observations of size $n = 20$ and $n = 40$ are drawn independently from the pool of 295 microarrays. Sampling is stratified, with half of the sample points drawn from each of the two prognosis classes. The true error for each observation of size n is approximated by a holdout estimator, whereby the $295 - n$ sample points not drawn are used as the test set (a good approximation of the true error, given the large test sample). This allows computation of the empirical deviation distribution for each error estimator using the considered classification rules. Since the observations are not independent, there is a degree of inaccuracy in the computation of the deviation distribution; however, for sample sizes $n = 20$ and $n = 40$ out of a pool of 295 sample points, the amount of overlap between samples is small. Figures 12 and 13 display plots of the empirical deviation distributions for LDA and CART, respectively, obtained by fitting beta densities to the raw data. Better performance is indicated by a narrow distribution centered close to zero. Note the low bias of resubstitution and the high variance of the cross-validation estimators. These are



[FIG11] Bolstered resubstitution error for linear classification and circular uniform bolstering kernels.



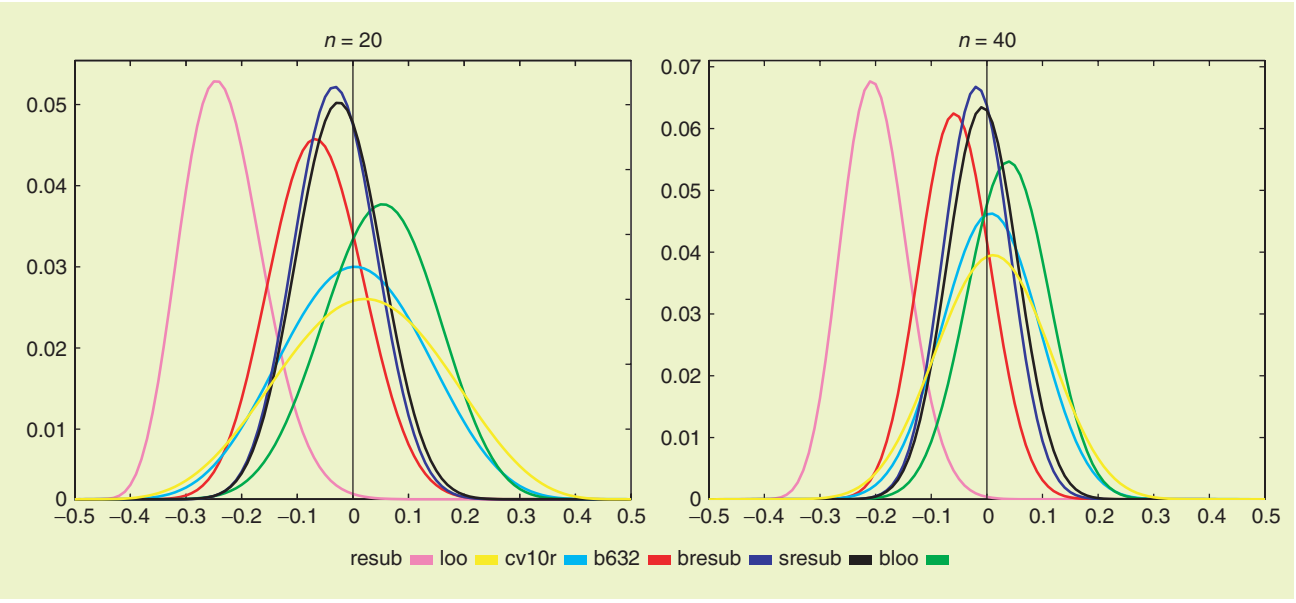
[FIG12] Beta-fit deviation distributions for LDA with sample sizes $n = 20$ and $n = 40$.

generally outperformed by the bootstrap and bolstered estimators; however, specific performance advantages depend heavily on the classification rule.

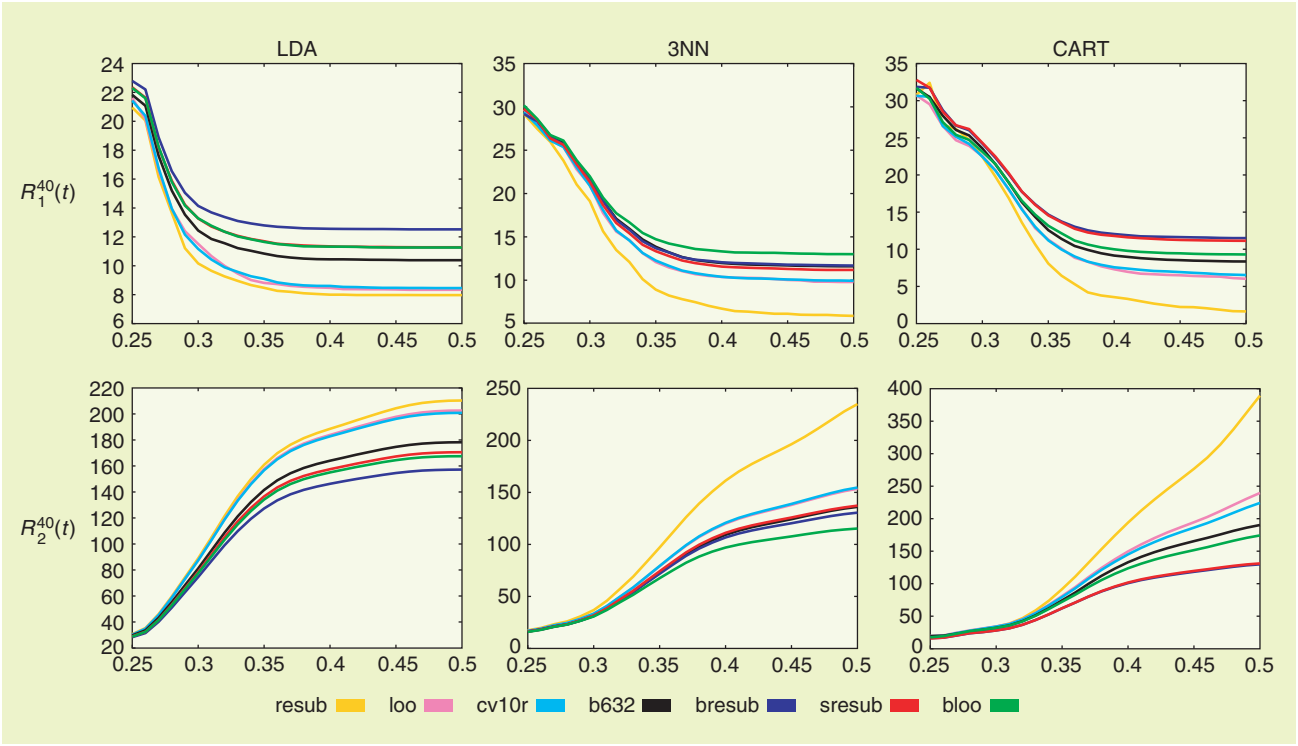
Computation time is important when judging an error estimator. Of those in the preceding demonstration, resubstitution is the fastest estimator. Leave-one-out is fast for a small number of samples, but its performance quickly degrades as the number of samples increases. The ten-fold cross-validation and bootstrap estimators are the slowest. Bolstered resubstitution can be hundreds of times faster than the bootstrap estimator.

In addition to constructing gene-expression classifiers, one may desire to simply rank gene sets based on their ability to classify phenotypes. Since there may be many gene sets that can provide good discrimination, one may wish to find sets composed of genes for which there is evidence of their molecular relationship with the phenotype of interest. The idea is that good feature sets may provide good candidates for diagnosis and therapy. Given a family of gene sets discovered by some classification rule, the issue is to rank them based on error. Thus, a natural measure of worth for an error estimator is its ranking accuracy for feature sets [33]. The measure will depend on the classification rule and the feature-label distribution. Consider two 20-dimensional, unit-variance spherical Gaussian class conditional distributions with means at $\delta\mathbf{a}$ and $-\delta\mathbf{a}$, where $\mathbf{a} = (a_1, a_2, \ldots, a_n)$, $|\mathbf{a}| = 1$, and $\delta > 0$ is a separation parameter. The Bayes classifier is a hyperplane perpendicular to the axis joining the means. The best feature set of size k corresponds to the k largest parameters among $\{a_1, a_2, \ldots, a_n\}$. We consider all feature sets of size three. For each sample of size 30, we obtain the LDA, 3NN, and CART classifiers, and for each of these we obtain the true error from the distribution and estimated errors based on resubstitution, cross-validation, bootstrap, and bolstering. We use two measures of merit. Each compares ranking based on true and estimated errors, under the condition that the true error is less that t. $R_1^K(t)$ is the number of feature sets in the truly top K feature sets that are also among the top K feature sets based on error estimation. It measures how well the error estimator finds top feature sets. $R_2^K(t)$ is the mean-absolute rank deviation for the K best feature sets. Figure 14 shows graphs obtained by averaging these measures over many samples. Cross-validation is generally poorer than .632 bootstrap, whereas the bolstered estimators are generally better.

When selecting features via an algorithm like SFFS that employs error estimation within the algorithm, one should expect the choice of error estimator to affect feature selection to a degree that is dependent on the classification rule and feature-label distribution [34]. To illustrate the issue, we consider the spherical 20-dimensional Gaussian class conditional distributions used to exhibit feature-set ranking, SFS and SFFS feature selection, and the LDA and 3NN rules. We consider selecting four features from samples of size 30. Table 1 gives the average true errors of the feature sets found by SFS, SFFS, and exhaustive search using various error estimators. The top row gives the average true error when the true error is used in feature selection. This is for comparison purposes only because, in practice, one cannot use the true error during feature selection. Note that both SFS and SFFS perform close to exhaustive searches when the true error is used. Of key interest is that the choice of error estimator can make a greater difference than the manner of feature selection. For instance, for LDA, an exhaustive search using leave-one-out results in average true error 0.2224, whereas SFFS using bolstered resubstitution yields an average true error of only 0.1918. SFFS using semibolstered resubstitution (0.2016) or bootstrap (0.2129) is also superior to exhaustive search using leave-one-out, although not as good as bolstered resubstitution. In the case of 3NN, once again SFFS with either bolstered resubstitution, semibolstered



[FIG13] Beta-fit deviation distributions for CART with sample sizes $n = 20$ and $n = 40$.



[FIG14] **Feature-set-ranking for LDA, 3NN, and CART.**

resubstitution, or bootstrap outperforms a full search using leave-one out. Extensive study is needed to gauge the impact of estimation on the many proposed methods of feature selection.

ANALYTIC RESULTS

The advent of massively parallel implementation has in recent years made possible broad simulation studies for key issues such as design cost, feature optimality, and comparative error estimation. Nevertheless, analytic results are desirable, and here much work remains. As noted previously, there is a great deal of literature on bounding the design cost, and there is much current activity in this field [35]. Moreover, historically there has been substantial work on obtaining exact representations and approximations for sample size and dimensionality issues, the focus being on QDA and LDA, owing to the form of their discriminants [36]. For instance, for LDA with Gaussian class conditional distributions possessing the identity covariance matrix and equal class probabilities, and estimation using the sample moments, there exists an approximation whose accuracy depends on the sample size and the number of features for $E[\Delta_{n,C}]$ that shows $E[\Delta_{n,C}] \approx O(d^2/n)$ [37]. Based on simulation, the approximation appears to be decent if the sample size modestly exceeds the number of features; however, this is often not the case with expression-based classification. In this section we consider two recent analytic results that have consequences for estimating the optimal number of features and for comparing the performance of error estimators in the case of small samples.

To apply QDA to sample data, the discriminant $Q(\mathbf{x})$ is replaced by its sample-based estimate, $Q_n(\mathbf{x})$, which is a random function whose distribution describes its probabilistic behavior. Recently, stochastic representations have been derived for $Q_{0,n}(\mathbf{x})$ and $Q_{1,n}(\mathbf{x})$, the conditional discriminants given class 0 and class 1, respectively [38]. These distributions can be used to study issues related to small-sample QDA. For instance, although the distributions are quite complicated, one can exactly express their means and variances, and thereby approximate them with Gaussian distributions to obtain approximate analytic solutions for the optimal number of features in small-sample settings [39].

Although simulation studies with regard to error estimation are beneficial, one would like to attain analytic results regarding estimator performance. We consider the root-mean-square (RMS) difference between the error and the error estimate, given by

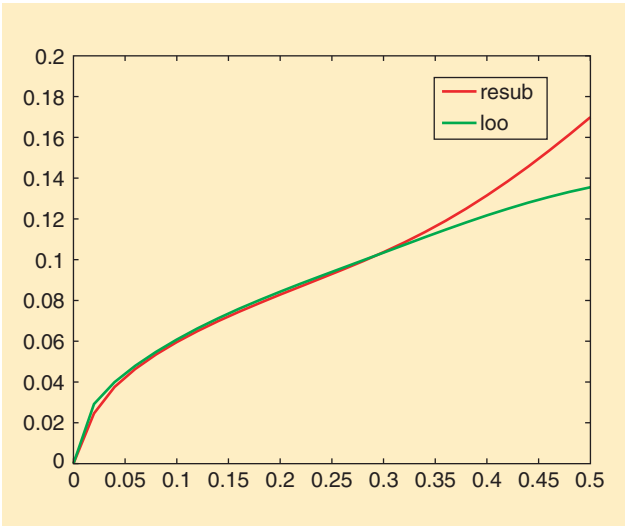
[TABLE 1] **ERROR RATES FOR FEATURE SELECTION USING VARIOUS ERROR ESTIMATORS.**

	LDA			3NN		
	EXHAUST	SFS	SFFS	EXHAUST	SFS	SFFS
TRUE	0.1440	0.1508	0.1494	0.1525	0.1559	0.1549
RESUB	0.2256	0.2387	0.2345	0.2620	0.2667	0.2543
LOO	0.2224	0.2403	0.2294	0.2301	0.2351	0.2364
CV5	0.2289	0.2367	0.2304	0.2298	0.2314	0.2375
B632	0.2190	0.2235	0.2129	0.2216	0.2192	0.2201
BRESUB	0.1923	0.2053	0.1918	0.2140	0.2241	0.2270
SRESUB	0.1955	0.2151	0.2016	0.2195	0.2228	0.2230

$$\mathrm{RMS}(\hat{\varepsilon}_n) = \sqrt{E[|\hat{\varepsilon}_n - \varepsilon_n|^2]},$$

where the expectation is with respect to the joint distribution of the random sample and both the classification rule and feature-label distribution are implicit in the notation. Consider *multinomial discrimination*, for which the feature components are random variables whose range is the discrete set $\{0, 1, \ldots, b-1\}$. This corresponds to choosing a fixed partition in $\mathcal{R}^d$ with b cells. The *histogram rule* assigns to each cell the majority label in the cell. In the case of this rule, for resubstitution and leave-one-out, there exist distribution-free upper bounds for the RMS [6]. As in the case of design-cost bounds, these tend to be impractical for small samples.

In the case of the histogram rule for multinomial discrimination, there exist exact analytic formulations of the RMS for resubstitution and leave-one-out [40]. Rather than just give some anecdotal examples of the RMS for different distributions, we consider a parametric Zipf model, which is a power-law discrete distribution where the parameter controls the difficulty of classification. Figure 15 shows the RMS as a function of the expected true error computed for a number of distinct models of the parametric Zipf model for $n = 40$ and $b = 8$. Their performances are virtually the same (resubstitution slightly better) for $E[\varepsilon_n] \leq 0.3$, which practically means equivalent performance in any situation where there is acceptable discrimination. For $b = 4$ (not shown), resubstitution outperforms leave-one-out across the entire error range and for $b = 16$ (not shown) the complexity of the model in comparison to the sample size is such that resubstitution is very low-biased and leave-one-out has better performance. If we fix the expected error at a value giving modest discrimination, $E[\varepsilon_n] = 0.2$, and consider different sample sizes, then resubstitution outperforms leave-one-out for $b \leq 7$, $b \leq 8$, and $b \leq 9$ at $n = 20$, $n = 40$, and $n = 60$, respectively. If we think of the Boolean model for gene regulation (to be discussed), then $b = 2, 4, 8$, and 16 correspond to connectivity one, two, three, and four, respectively, connectivity being the number of genes that predict the state of any other gene in the network. Since in practice connectivity is often bounded by three, this means that, relative to leave-one-out, resubstitution can provide equivalent estimation of prediction error at practical error levels for three predictors and better prediction error at all error levels for one and two predictors. Given the need to sometimes estimate the errors of tens of thousands of predictor functions, there is an enormous computational benefit in using resubstitution, in addition to the better prediction for one and two predictors.



[FIG15] Analytic error curves for resubstitution and leave-one-out cross-validation for multinomial discrimination with eight cells.

Besides bounding the RMS, one might desire confidence that the true error is less than some function of the resubstitution error, which means finding expressions of the form

$$P\left(\varepsilon_n \leq h\left(\hat{\varepsilon}_n^{res}, n, \delta\right)\right) \geq 1 - \delta.$$

This problem has also received much attention [41].

While the literature on bounds is rich, the bounds are typically irrelevant to GSP because GSP generally deals with small samples. Certainly, improved bounds would be useful, but numerous bounds have been shown to be tight ; in these cases, there is no hope to make them useful for GSP. We believe that what is required is more attention to exact representations and approximations suitable for small samples. No doubt, many of these issues are difficult, but the rewards relating to model choice and validity will be well worth the effort.

CLUSTERING

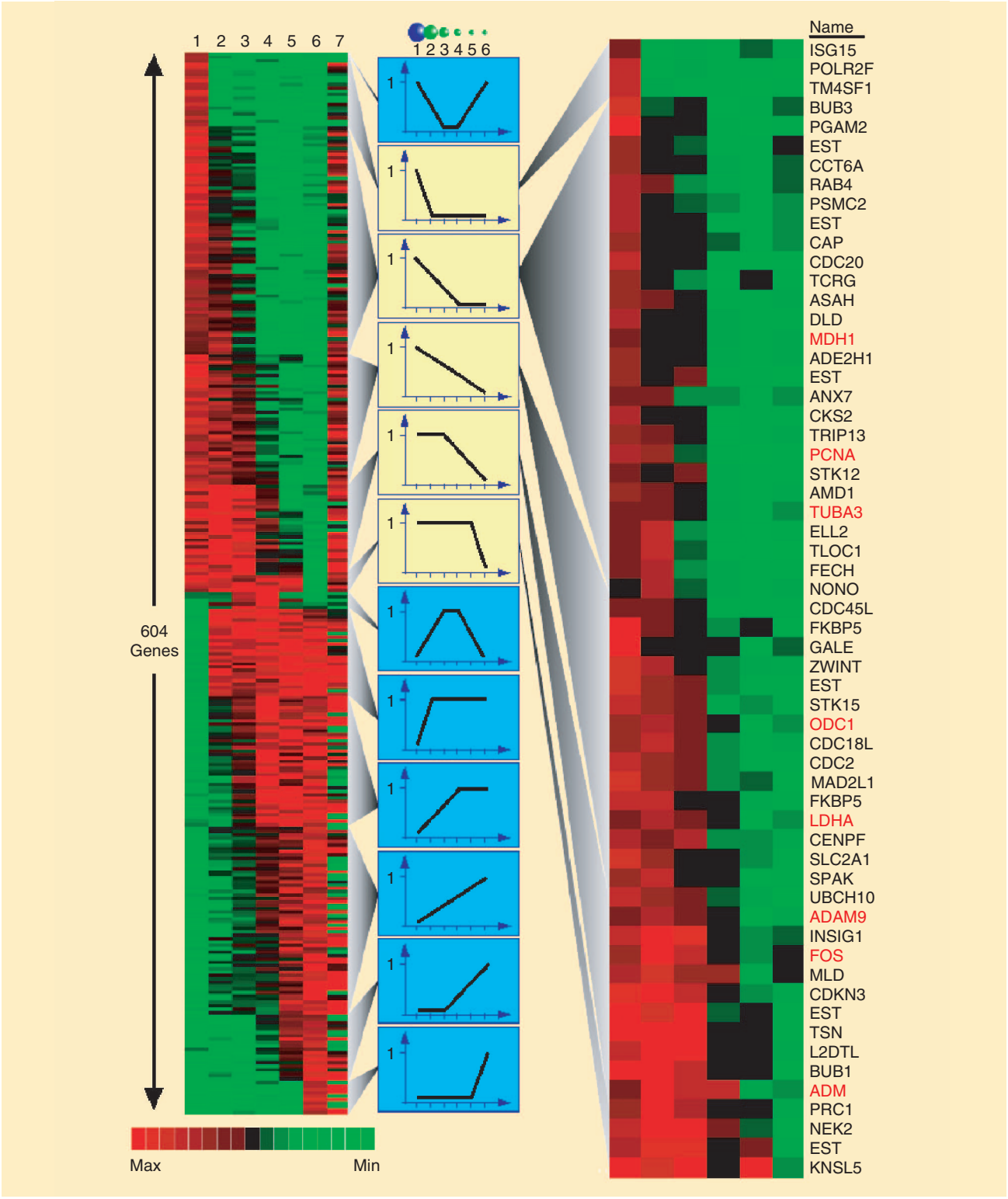
A cluster operator takes a set of data points and partitions the points into clusters (subsets). Clustering has become a popular data-analysis technique in genomic studies using gene-expression microarrays. The process of time-series clustering groups together genes whose expression levels exhibit similar behavior through time. Similarity indicates possible coregulation. Another way to use expression data is to take expression profiles over various tissue samples and then cluster these samples based on the expression levels for each sample. This approach offers the potential to discriminate pathologies based on their differential patterns of gene expression.

Figure 16 shows a clustering of gene-expression time-course profiles, where the clusters have been found by matching with the templates in the second column. These profiles reflect the response to androgen deprivation therapy for advanced prostate cancer, with the profiles in the third column corresponding to 59 androgen-responsive genes; the known androgen receptor targets have their names in red [42].

Figure 17 shows a sample of 31 patients (columns) suffering from B-cell lymphoma and the expression profiles (rows) of 25 genes across the sample. The samples have been hierarchically clustered and the clusters correspond perfectly to two types of lymphoma, DLBCL and MCL [43]. The genes have also been hierarchically clustered, and it appears from the figure that the red-labeled genes are up-regulated for MCL and down-regulated for DLBCL, whereas the green-labeled genes

are up-regulated for DLBCL and down-regulated for MCL. Thus, the different gene clusters seem to "classify" the lymphomas. Perhaps more precisely, the clustering might provide feature selection in the sense that actual classification might be accomplished by a two-gene feature set, one red-labeled and one green-labeled.

Despite the popularity of clustering, one must pose the epistemological question: What is the scientific content of the



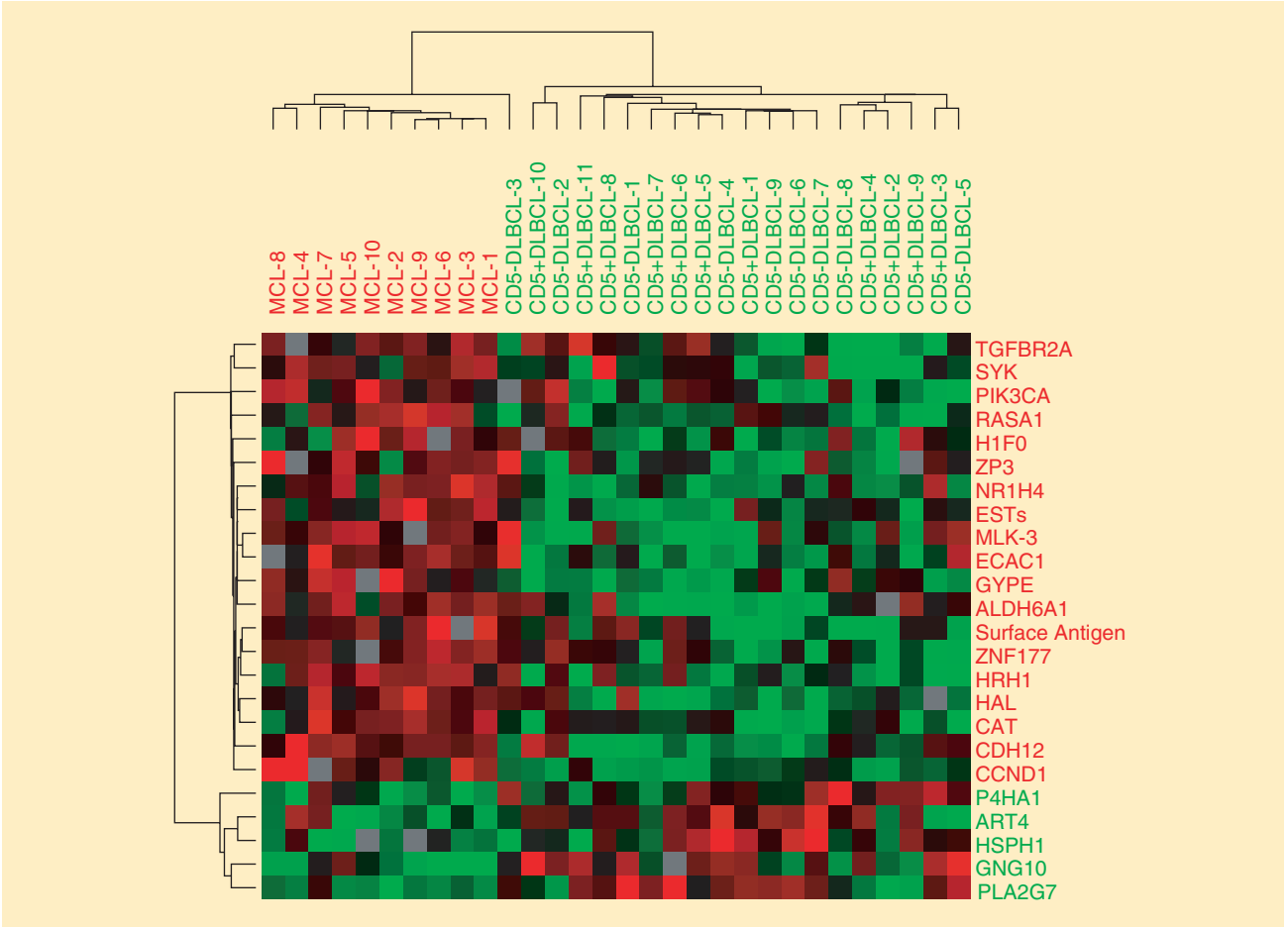
[FIG16] Template-based clustering of gene-expression time-course profiles reflecting the response to androgen deprivation therapy for advanced prostate cancer.

output of a clustering algorithm? With classification, two fundamental characteristics are exhibited: 1) classifier error can be estimated under the assumption that the sample data arise from an underlying feature-label distribution, and 2) given a family of classifiers, sample data can be used to learn the optimal classifier in the family. A classifier is a mathematical model that provides a decision procedure relative to real-world measurements. The model represents scientific knowledge to the extent that it has predictive capability. As with any scientific model, the classifier has two parts: the mathematical structure and the mechanism by which it is tested, the purpose of testing (error estimation) being to measure the goodness of the model. Historically, clustering has generally lacked both fundamental characteristics of classification. In particular, it lacks inference in the context of a probability model. Jain et al. write, "Clustering is a subjective process; the same set of data items often needs to be partitioned differently for different applications" [44]. In the context of gene-expression microarrays, Kerr and Churchill write, "How does one make statistical inferences based on the results of clustering?" [45]. Indeed, how is one going to judge the worth of clustering algorithms unless it is based on their inference capabilities? This difficulty is illustrated in Figure 18, where two seemingly good visualizations produce two very different partitions.

Many validation techniques have been proposed for evaluating clustering results; however, these are generally based on the degree to which clusters derived from sample data satisfy certain heuristic criteria. This is significantly different than classification, where the error of a classifier is given by the probability of an erroneous decision. We are confronted here by a basic issue of scientific epistemology, and the matter can only be resolved by a sound mathematical framework. Subjective appreciations are useful in the formulation of hypotheses, but these are constitutive of scientific knowledge only if they are set in a predictive framework.

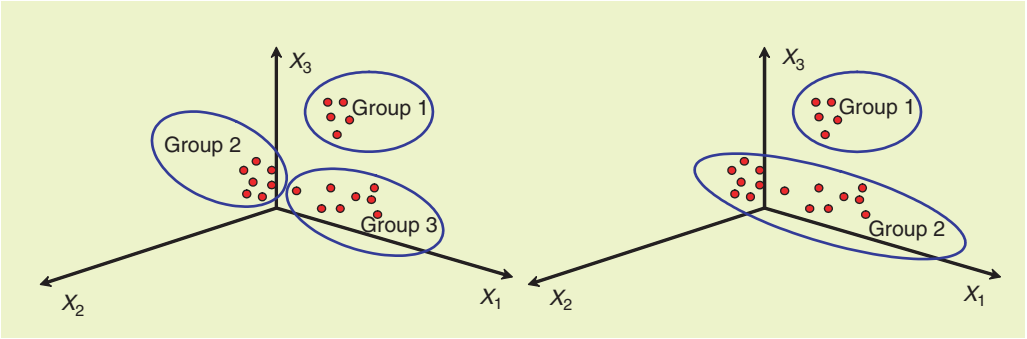
Regarding learning, clustering has sometimes been referred to as "unsupervised learning." Even if we do not argue that such a term is an oxymoron, there is certainly no learning going on when clustering is used in the typical manner. With clustering, an operator is applied to a point set, and it yields a partition of the point set. If there is no randomization within the algorithm, then the operator is simply a function on point sets; if there is randomization within the algorithm, then the operator is a random function on point sets. In either event, there is no learning.

If we wish to estimate the error of a cluster operator, then we must assume that clusters resulting from the operator can be compared to the correct clusters for the data set in the



[FIG17] Hierarchical clustering of expression profiles of patients suffering from two types of lymphoma: DLBCL and MCL.

context of a probability distribution, thereby providing an error measure. If we assume that data points are generated from different probability distributions, and these are known, then we can generate independent synthetic data to test the performance of a cluster operator. A sample of point sets is generated, the algorithm is applied to each point set and the clusters are evaluated relative to the known partition according to the distributions, and the errors are averaged over the point sets composing the sample [46].



[FIG18] Two clusterings of the same data.

The key to a general probabilistic theory of clustering, including error estimation and learning, is to recognize that classification theory is based on operators on random variables and that the theory of clustering needs to be based on operators on random point sets. The predictive capability of a clustering algorithm must be measured by the decisions it yields regarding the partitioning of random point sets. Once this is recognized, the path to the development of error estimators for clustering accuracy and rules for learning clustering operators from data is open, including finding Bayes cluster operators (optimal clustering algorithms) [47]. Although the formulation of a predictive theory is dictated by the fact that clustering algorithms are set operators, having formulated a probabilistic theory, an enormous amount of research remains to be done: developing the mathematical theory, inventing clustering rule models to learn cluster operators, and devising experimental methods, the latter being especially difficult given the historical nonscientific use of clustering and the challenging environment of random sets. Of particular importance is the development of validity-type measures that can be applied to one point set and that possess some quantitative relation to the clustering error rate.

GENETIC REGULATORY NETWORKS

The design of analytical and computational tools for the modeling and analysis of gene regulation can help to understand gene function and unravel the mechanisms underlying regulation [48], [49]. This understanding will have a significant impact on the development of techniques for drug testing and therapeutic intervention for treating human diseases [50]. Two related aspects of a genetic regulatory system determine its dynamical behavior and, therefore, must be modeled and analyzed: the connectivity structure and the set of interactions between the elements [51], [52]. There have been numerous attempts to model the dynamical behavior of genetic regulatory networks, ranging from deterministic to fully stochastic, using either discrete-time or continuous-time descriptions of gene interactions (see [53] for a literature review).

Given that genes communicate via the proteins they encode and that protein production (transcription and translation) is controlled by a multitude of biochemical reactions, which are in turn influenced by factors both internal and external to the cell, one can assume that the gene expression of a particular gene i appears as a random function $X_i(t)$ of the cell's internal and external environments. Thus, to investigate the dynamics of a genetic regulatory network, one must construct a good mathematical model for the dynamical behavior of the gene-expression vector $\mathbf{X}(t) = (X_1(t), X_2(t), \ldots, X_N(t))$ for the N genes interacting in the network. The goodness of a model can be considered with respect to several criteria: the level of detailed description of the biochemical reactions involved in gene regulation, model complexity, model parameter estimation, and, most importantly, the predictive power of the model. The stochastic-differential-equation model is arguably the most detailed description of the dynamics of $\mathbf{X}(t)$, since it could embed, at least in principle, all of the information about the biochemical reactions involved in the gene interactions. At the same time, this kind of model has high complexity, and the estimation of its parameters cannot be done without reliable time series data, and a good amount of it. Ultimately, the balance between available data, estimation techniques, and model complexity determines the usefulness of a given model.

While it might be tempting to design a model that captures, at least in theory, the gene interactions on a very fine scale, one should be aware of the increased demand for larger data sets and the requirement that model parameters must be determined so that the model generates solutions consistent with observable data. In this article, we focus on coarse models of genetic interaction designed using the limited amount of microarray gene expression data typically available. Paradigms that have been considered in this context include directed graphs, Bayesian networks, Boolean networks, generalized logical networks, and, most recently, probabilistic Boolean networks. Here we will highlight the main research issues using Boolean and probabilistic Boolean networks. Similar issues arise in the case of other networks.

PROBABILISTIC BOOLEAN NETWORKS

A Boolean network is defined by a set of nodes, $V = \{x_1, x_2, \ldots, x_n\}$, and a list of Boolean functions, $F = \{f_1, f_2, \ldots, f_n\}$

[54], [55]. Each x_k represents the state (expression) of a gene g_k where $x_k = 1$ or $x_k = 0$, depending on whether the gene is expressed or not expressed. The Boolean functions represent the rules of regulatory interaction between genes. Network dynamics result from a synchronous clock with times $t = 0, 1, 2, \ldots$, and the value of gene g_k at time $t+1$ is determined by $x_k(t+1) = f_k(x_{k1}, x_{k2}, \ldots, x_{k,m(k)})$, where the nodes in the argument of f_k form the regulatory set for x_k (gene g_k). The numbers of genes in the regulatory sets define the connectivity of the network, with maximum connectivity often limited. At time point t, the state vector $\mathbf{x}(t) = (x_1(t), \ x_2(t), \ldots, x_n(t))$ is called the *gene activity profile* (*GAP*). The functions together with the regulatory sets determine the network wiring. A Boolean network is a very coarse model; nonetheless, it facilitates understanding of the generic properties of global network dynamics [56], [57], and its simplicity mitigates data requirements for inference. A Boolean network is based on the premise that genes interact with each other through Boolean logic. Recent work using the NCI 60 Anti-Cancer Drug Screen has demonstrated that such interactions can be detected in gene expression data [58].

Microarray technology yields simultaneous measurements of expression status for thousands of genes and can be utilized for network inference [59]. By viewing gene status across different conditions, it is possible to establish relationships between genes that show variable status across the conditions. Owing to limited replications, we assume that gene expression data are quantized based on some statistical analysis of the raw data. One way to establish multivariate relationships among genes is to quantify how the estimate for the expression status of a particular *target gene* can be improved through knowledge of the status of some other *predictor genes*. This is formalized via the *coefficient of determination* (*CoD*) [60], which is essentially a nonlinear, multivariate generalization of the familiar goodness of fit measure in linear regression. For our purposes, it is sufficient to note that the CoD measures the degree to which the best estimate for the transcriptional activity of a target gene can be improved using the knowledge of the transcriptional activity of some predictor genes, relative to the best estimate in the absence of any knowledge of the transcriptional activity of the predictors. The CoD is a number between zero and one, a higher value indicating a tighter relationship. Given a target gene, several predictor sets may provide equally good estimates of its transcriptional activity, as measured by the CoD. Moreover, one may rank several predictor sets via their CoDs. Such a ranking provides a quantitative measure to determine the relative ability of each predictor set to improve the estimate of the transcriptional activity of the particular target gene. While attempting to infer intergene relationships, it makes sense to not put all our faith in one predictor set. Instead, for a particular target gene, a better approach is to consider a number of predictor sets with high CoDs. The consideration of each retained predictor set as indicative of the transcriptional activity of the target gene with a probability proportional to its CoD constitutes feature selection for gene prediction.

Having inferred intergene relationships, this information can be used to model the evolution of the gene activity profile over time. It is unlikely that the determinism of the Boolean-network model will be concordant with the data. One could pick the predictor set with the highest CoD, but as noted previously, there are usually a number of almost equally performing predictor sets, and the CoDs we have for them are only estimates from the data. By associating several predictor sets with each target gene, it is not possible to obtain with certainty the transcriptional status of the target gene at the next time point; however, one can compute the probability that the target gene will be transcriptionally active at time $t+1$ based on the gene activity profile at time t. The time evolution of the gene activity profile then defines a stochastic dynamical system. Since the gene activity profile at a particular time point depends only on the profile at the immediately preceding time point, the dynamical system is Markovian. Such systems can be studied in the established framework of Markov chains and Markov decision processes. These ideas are mathematically formalized in *probabilistic Boolean networks* (*PBNs*) [61], [62]. In a PBN, the transcriptional activity of each gene at a given time point is a Boolean function of the transcriptional activity of the elements of its predictor sets at the previous time point. The choice of Boolean function and associated predictor set can vary randomly from one time point to another in accordance with the CoD-based selection probabilities associated with the different predictor sets. This defines an *instantaneously random* PBN.

Instead of simply assigning Boolean functions at each time point according to CoD ranking, one can take the perspective that the data come from distinct sources, each representing a *context* of the cell. From this viewpoint, the data derive from a family of deterministic networks and, were we able to separate the samples according to the contexts from which they have been derived, there would in fact be CoDs with value one, indicating deterministic biochemical activity for the wiring of a particular constituent network. Under this perspective, the only reason that it is not possible to find predictor sets with CoD equal (or very close to) one is because they represent averages across the various cellular contexts with their correspondingly various wirings. This perspective results in the view that a PBN is a collection of Boolean networks in which one constituent network governs gene activity for a random period of time before another randomly chosen constituent network takes over, possibly in response to some random event such as an external stimulus. Since the latter is not part of the model, network switching is random. This model defines a *context-sensitive* PBN. The probabilistic nature of the constituent choice reflects the fact that the system is open, not closed. The context-sensitive model reduces to the instantaneously random model by having network switching at every time point.

Given a Boolean network, one can partition the state-space into a number of attractors along with their basins of attraction. The attractors characterize the long-run behavior of the network and have been conjectured by Kauffman to be indicative of the cell type and phenotypic behavior of the cell [56]. For instance, it

is thought that apoptosis and cell differentiation correspond to some singleton attractors and their basins, while cell proliferation corresponds to a cyclic attractor along with its associated basin [57]. Changes in the Boolean functions, via mutations or rearrangements, can lead to a rewiring in which attractors appear that are associated with tumorigenesis. This is likely to lead to a cancerous phenotype unless the corresponding basins are shrunk via new rewiring, so that the cellular state is not driven to a tumorigenic phenotype, or, if already in a tumorigenic attractor, the cell is forced to a different state by flipping one or more genes. The objective of cancer therapy would be to use drugs to do one or both of the above. These ideas for Boolean networks can be generalized to PBNs by noting that the dynamic behavior of PBNs can be described by Markov chains, so that a PBN has equivalence classes of communicating states analogous to the basins of attraction for Boolean networks. Similarly, since all the states in an equivalence class communicate, there is a steady-state distribution local to each equivalence class so that the long-run behavior within that class can be studied. Furthermore, by assuming that each gene has a small probability of undergoing a random flip that leaves the network wiring unchanged, as in the case of activation or inactivation owing to external stimuli such as mutagens, heat stress, etc. [56], the overall Markov chain is ergodic, which then guarantees the existence of a global steady-state distribution [63].

While we are limiting ourselves to binary networks, much of the theory and application goes over directly to their extension to *probabilistic gene regulatory networks* (*PGRNs*), the only difference being that quantization for PGRNs need not be binary, but can be any finite quantization. A particularly important case is ternary quantization, where expression levels take on the values $+1$ (up regulated), -1 (down regulated), and 0 (invariant). PGRNs are commonly called probabilistic Boolean networks to emphasize their logical nature, such as in the case of ternary logic.

Owing to complexity issues that arise for both computational and statistical reasons, network compression is an important problem in network modeling, the point being to reduce the size of a given network while preserving some desirable properties of the original network. For PBNs, one approach involves a projection mapping, defined to reduce the complexity of a PBN while maintaining consistency with the original probability structure of the PBN [64]. This mapping transforms a given PBN into a new one with one less gene while preserving the probability structure of the original PBN. In the process, the deleted gene becomes a latent variable in the new network, with the result being that every predictor in the original network that had the deleted gene as an essential variable is replaced by two predictors in the new network. These two predictors capture the differences in the state transitions corresponding to the two different possible values for the latent gene. The projection mapping can be repeatedly applied to achieve the desired reduction in the number of genes. An obvious drawback of the projection mapping is that the number of genes in the PBN is reduced at the expense of an exponential increase in the number of predictors and, hence, the number of constituent Boolean networks for the new PBN. To remedy the situation, a *reduction mapping* technique has been introduced that does not necessarily preserve the probability structure of the original PBN, but achieves reduction in PBN complexity while ensuring that the reduced PBN is "close" to the original one as measured by an appropriately defined metric [65]. The two compression approaches just discussed focus on reducing the complexity of a PBN that has already been inferred from the data. A different approach, motivated by the MDL principle and mentioned when we discussed complexity regularization, aims to penalize network complexity when inferring the network itself [28]. Since the network complexity is explicitly penalized in the cost function during the inference process itself, the designed network cannot be too complex. The complexity-reduction techniques proposed thus far only scratch the surface. There need to be developed better methods that depend on minimizing some difference metric between the original and reduced networks and which preserve (to the extent possible) the dynamical structure of the original network.

NETWORK INFERENCE

For genetic regulatory networks to be of practical benefit, there must be methods to design them based on experimental data. We confront three impediments:

- model complexity
- limited data
- lack of appropriate time-course data to model dynamics.

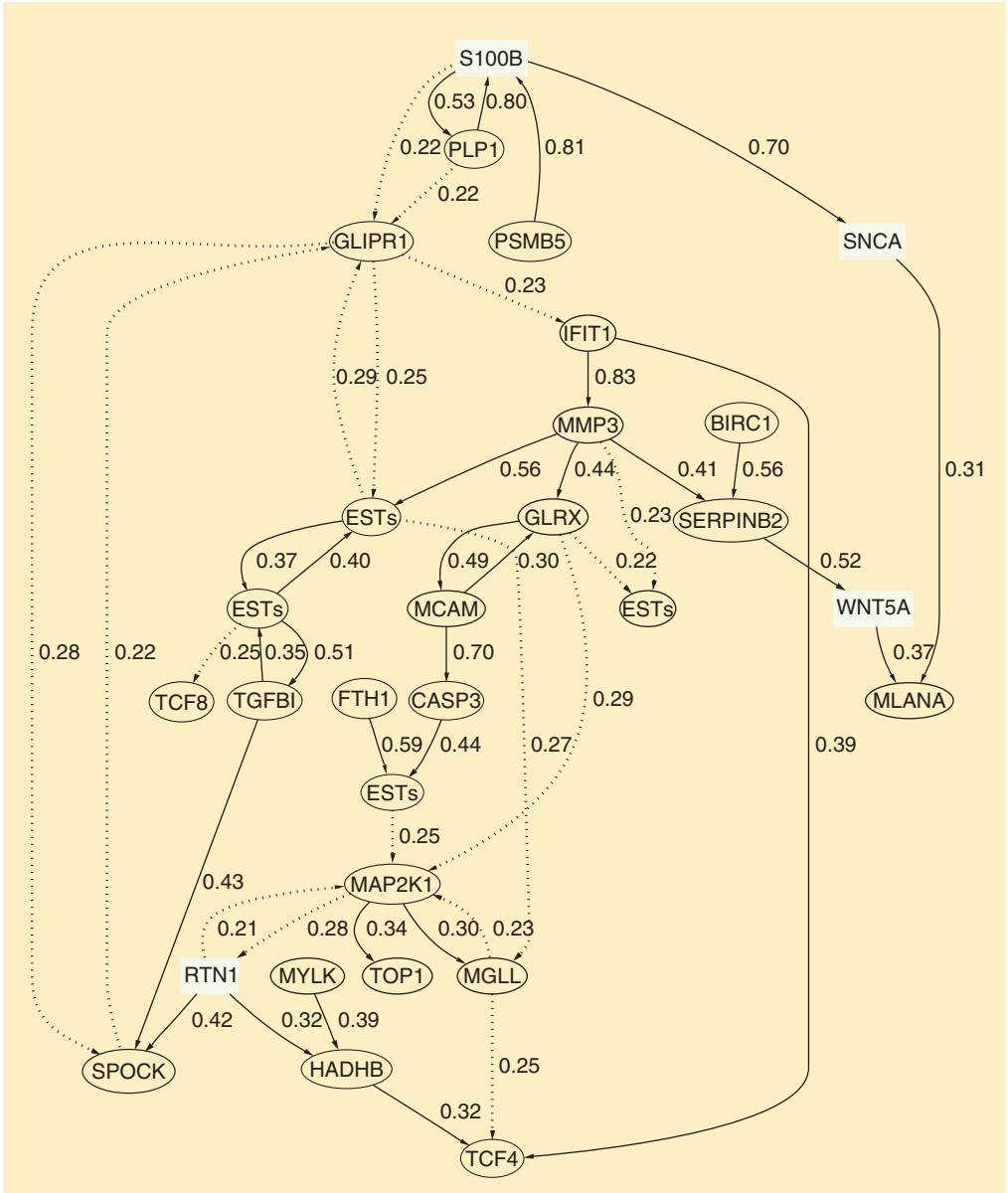
A key research issue in GSP has been the inference of genetic regulatory networks, and several approaches to the *network inference problem* have been proposed in the literature, most of which are based on gene-expression microarray data. Here, we briefly outline some of the proposed methods for PBNs and the rationale behind each of them (there also having been substantial study of inferring Boolean networks [66], [67]).

As first proposed, the inference of the PBN is carried out using the CoD [61]. For each gene in the network, a number of high-CoD predictor sets are found and these high-CoD predictor sets determine the evolution of the activity status of that particular gene. Furthermore, the selection probability of each predictor set for a target gene is assumed to be the ratio of the CoD of that predictor set to the sum of the CoDs of all predictor sets used for that target gene. This approach makes intuitive sense since it is reasonable to assign the selection probability of each predictor set in a PBN to be proportional to its predictive worth as quantified by the CoD.

A second approach to PBN construction uses mutual information clustering and reversible-jump Markov chain Monte Carlo (MCMC) predictor design [68]. First, mutual information minimization clustering is used to determine the number of possible parent gene sets and the input sets of gene variables corresponding to each gene. Thereafter, each (predictor) function from the possible parent gene sets to each target gene is modeled by a perceptron consisting of a linear term and a nonlinear term, and a reversible-jump MCMC technique is used to calculate the model order and the parameters. Finally, the selection probability for each predictor set is calculated using the ratio of the CoDs.

In a third approach to inferring PBNs, subnetworks are constructed in the context of a directed graph by starting with a seed consisting of one or more genes believed to participate in a viable subnetwork [69]. The choice of seed represents prior partial knowledge about the genetic regulatory network of interest. Given the seed, new genes are iteratively joined in a manner that enhances subnetwork autonomy. The proposed algorithm has been applied using both the CoD and the Boolean-function influence [61], which measures interaction between genes. The algorithm has the benefit of producing a collection of small, tightly knit autonomous subnetworks as opposed to one massive network with a large number of genes. Such small subnetworks are more amenable to modeling and simulation studies. When properly seeded, they are more likely to capture a small set of genes that may be maintaining a specific core regulatory mechanism.

Figure 19 shows a melanoma subnetwork grown from four seed genes: WNT5A, RTN1, S100B and SNCA. The CoD has been used as the measure of gene interaction and the gray boxes denote the seed genes, while the white ellipses are the genes added by the algorithm. The solid lines represent strong connections (connection strengths exceeding 0.3) while the dotted lines represent weak connections (connection strengths between 0.2 and 0.3). This network reveals some very interesting insights that are highly consistent with prior biological knowledge derived from earlier gene expression studies using melanoma cell lines [70], [71]. For instance, it is known that the WNT5A gene product has the capability to drive aspects of cell motility and invasiveness. That being the case, it is to be expected that genes playing a part in either mediating extracellular matrix remodeling/interaction, such as MMP3 (matrix metalloproteinase 3), SERPINB2 (serine (or cysteine) proteinase inhibitor), and MCAM (melanoma adhesion molecule), or mediating cellular movement, such as MYLK (myosin light polypeptide kinase), would share regulatory information with WNT5A. On the other hand, it is not known how WNT5A regulation is coupled to other genes playing a part in melanoma cell proliferation, such as MAP2K1 (mitogen-activated protein kinase kinase 1) and the regulation of apoptosis such as CASP3 (cystein aspartate protease 3) and BIRC1 (baculoviral IAP repeat-containing 1). Nevertheless, it is quite possible that high-level coordination of these activities exists through either specific circuitry or as a consequence of differing extracellular interactions that arise from metastatic cell movement.



[FIG19] A melanoma subnetwork grown from four seed genes: WNT5A, RTN1, S100B and SNCA.

A key issue in network design arises because much of the currently available gene-expression data comes to us from the *steady-state* phenotypic behavior and does not capture any temporal history. Consequently, the

process of inferring a PBN, which is a dynamical system, from steady-state data is a severely *ill-posed inverse problem*. Steady-state behavior constrains the dynamical behavior of the network but does not determine it and, therefore, building a dynamical model from steady-state data is a kind of overfitting. It is for this reason that a designed network should be viewed as providing a regulatory structure that is consistent with the observed steady-state behavior. Also, it is possible that several networks may emerge as candidates for explaining the steady-state data. Under the assumption that we are sampling from the steady state, a key criterion for checking the validity of a designed network is that much of its steady-state mass lies in the states observed in the sample data because it is expected that the data states consist mostly of attractor states [72].

A number of recent papers have focussed on network inference, keeping in mind that most of the data states correspond to steady-state behavior. In one of these, a fully Bayesian approach has been proposed to construct probabilistic gene regulatory networks that emphasize network topology [73]. The method computes the possible parent sets of each gene, the corresponding predictors, and the associated probabilities based on a nonlinear perceptron model, using a reversible jump MCMC technique. An MCMC method is employed to search the network configurations to find those with the highest Bayesian scores to construct the PGRNs. This method has been applied to a melanoma cell line data set. The steady-state distribution of the resulting model contains attractors that are either identical or very similar to the states observed in the data, and many of the attractors are singletons, which mimics the biological propensity to stably occupy a given state. Furthermore, the connectivity rules for the most optimal generated networks constituting the PGRN were found to be remarkably similar, as would be expected for a network operating on a distributed basis, with strong interactions between the components.

Two other approaches to PBN inference are currently under development. In the first, algorithms are developed to attain Boolean networks satisfying biologically related constraints such as limited attractor structure, transient time, and connectivity [74]. In the second, network design is carried out by attributing the apparent inconsistencies in the data to different underlying contexts for the network; this is done in such a way as to minimize the number of contexts needed to resolve apparent data inconsistencies while at the same time making the probabilistic structure of the designed network compatible with the empirical distribution of the designed network [75].

If we consider network inference from the general perspective of an ill-posed inverse problem, then one can formalize inference by postulating criteria that constitute a solution space in which a designed network must lie. For this, we propose two types of criteria.

- *Constraint criteria* are composed of restrictions on the form of the network, such as biological and complexity constraints.
- *Operational criteria* are composed of relations that must be satisfied between the model and the data.

Examples of constraint criteria include limits on connectivity and attractor cycles. One example of an operational criterion is some degree of concordance between sample and model CoDs, and another is the requirement that data states are attractor states in the model. The inverse problem may still be ill-posed with such criteria, but all solutions in the resulting space can be considered satisfactory relative to the requirements imposed by the criteria.

Before leaving inference, we note that in addition to the ongoing effort to infer PBNs, there has been a long effort to infer Bayesian and dynamic Bayesian networks (DBNs) [76]–[78]. Not only do DBNs and PBNs represent stochastic models for regulation, they are closely related according to the following theorem: PBNs and discrete-valued DBNs whose initial and transition networks are assumed to have only within and between consecutive slice connections, respectively, can represent the same joint distribution over their corresponding variables [79]. This theorem holds open the possibility of applying design and intervention methods for one model to the other, where it is necessary to recognize that the mapping between PBNs and DBNs is many-to-one, so that a DBN does not specify a specific PBN.

INTERVENTION

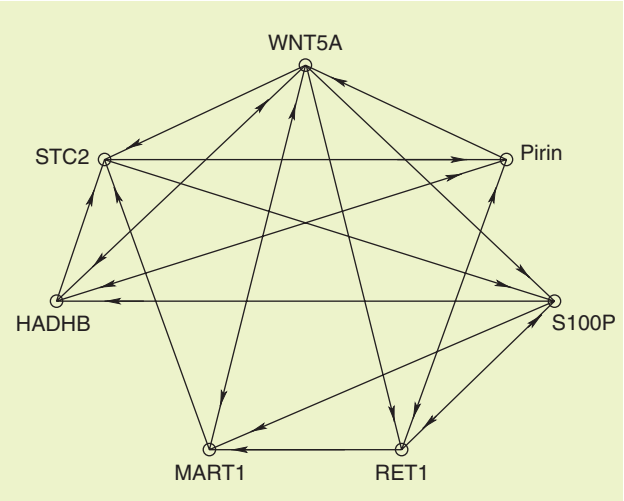
The ultimate objective of genetic regulatory network modeling is to use the network to design different approaches for affecting the evolution of the gene activity profile of the network. To date, such intervention studies have been carried out on PBNs using three different approaches: 1) resetting the state of the PBN, as necessary, to a more desirable initial state and letting the network evolve from there [63], 2) changing the steady-state (long run) probability distribution of the network by minimally altering its rule-based structure [80], and 3) manipulating external (control) variables that affect the transition probabilities of the network and can, therefore, be used to desirably affect its dynamic evolution over a finite time horizon [81].

We briefly describe the results found in [81], where an intervention study was carried out using a PBN derived from gene expression data collected in a study of metastatic melanoma [70]. In this expression-profiling study, the abundance of mRNA for the gene WNT5A was found to be highly discriminating between cells with properties typically associated with high metastatic competence versus those with low metastatic competence. These findings were validated and expanded in a second study [71], in which experimentally increasing the levels of the Wnt5a protein secreted by a melanoma cell line via genetic engineering methods directly altered the metastatic competence of that cell as measured by the standard in vitro assays for metastasis. Furthermore, it was found that an intervention that blocked the Wnt5a protein from activating its receptor by the use of an antibody that binds the Wnt5a protein could substantially reduce Wnt5a's ability to induce a metastatic phenotype. This suggests the creation of a control strategy that uses an intervention that reduces the WNT5A gene's action in affecting biological regulation, since the available data suggests that disruption of this influence could reduce the chance of a melanoma metastasizing—a desirable outcome.

To this end, a seven-gene network, including the activity of the WNT5A gene, was derived from the available gene expression data. This network, along with the multivariate relationships between the genes, is shown in Figure 20. For each gene in this network, the two best two-gene predictors were used and their associated CoDs were computed. This information was used to obtain the transition probabilities for the Markov chain associated with the PBN. The intervention problem was then posed as a finite-horizon optimal control problem. The performance index, or cost function, was chosen to reflect the tradeoffs between the intervention effort and the terminal penalty associated with ending up in an undesirable (bad) state at the end of the control horizon. Since the control objective is to reduce the activity of the WNT5A gene, the entire state space was partitioned into good and bad regions, the latter being characterized by WNT5A overexpression. Bad states were assigned higher terminal penalties than good states, and the optimization problem was solved by dynamic programming. Two possible interventions were considered: intervening with Wnt5a directly (through its antibody) and intervening through another gene called pirin. In each case, it was found that the network with control performed better (in a probabilistic sense) than the network without control, so that the control objective was met. Furthermore, controlling WNT5A directly yielded better performance than trying to control it through pirin, which agrees with intuitive expectations.

The intervention approaches 1) and 3) above do not attempt to alter the steady-state behavior of the network while approach 2) attempts to increase the steady-state probability mass in the desirable states. However, all of these approaches are essentially first-cut solutions and will have to be improved upon. For instance, approach 2) uses a brute-force search algorithm, and a more systematic approach will have to be found by which one can increase the steady-state probability mass in the desirable set of states while correspondingly decreasing the mass in the undesirable ones.

Motivated by biological considerations, the initial result on intervention presented here has been subsequently extended in several directions. In one approach, the optimal intervention algorithm has been modified to accommodate the case where the entire state vector (gene activity profile) is not available for measurement [82]. In a second, the intervention results have been extended to context-sensitive PBNs, which we believe are a closer approximation at modeling biological reality [83]. Several open issues, however, remain. These issues, some of which we now discuss, will have to be successfully tackled before the intervention methods discussed here find application in actual clinical practice.



[FIG20] WNT5A network.

Methodical assignment of terminal penalties: The formulation of the optimal control problem assumes that there is a terminal penalty associated with each state of the PBN. However, the assignment of these terminal penalties for cancer therapy is by no means a straightforward task. The reason is that, while the intervention will be carried out only over a finite horizon, one would like to continue to enjoy the benefits in the steady state. For such purposes, the kind of terminal penalty used for the melanoma cell line study of [81] is inadequate since it fails to capture the steady-state behavior once the intervention has ceased. To remedy the situation, one possibility is to assign terminal penalties based on equivalence classes. The results of preliminary simulation studies in this regard appear to be encouraging [84].

Choice of control inputs: In the case of the melanoma cell line study presented in [81], one of the genes in the PBN, namely pirin, was used as a control input. A question arises as to which gene (or genes) should be used as the control input. Of course, one consideration is to use genes for which inhibitors or enhancers are readily available. However, even if such a gene is chosen, how can we be certain that it is capable of controlling some other gene(s)? Although the answer is not clear at this stage, we do believe that the traditional control theoretic concepts such as *controllability* and *observability* [85] may yield some useful insights. Another possibility is to use the concept of gene influence [61], an approach that has been preliminarily explored in [83].

Intervening to alter the steady-state behavior: Given a Boolean network, one can partition the state-space into a number of attractors along with their basins of attraction. The attractors characterize the long-run behavior of the Boolean network and have been conjectured by Kauffman to be indicative of the cell type and phenotypic behavior of the cell. Consequently, a reasonable objective of therapeutic intervention could be to alter the attractor landscape in the associated Boolean network. Such an idea can be generalized to PBNs and a brute-force approach aimed at such intervention has been proposed [80]. A more systematic approach for affecting the steady-state behavior needs to be developed. Perhaps this could be achieved by exploiting and further developing existing results in the control literature.

The optimal intervention strategies obtained thus far assume known transition probabilities and pertain to a finite horizon problem of known length. Their extension to the situation where the transition probabilities and the horizon length are unknown is also a topic for further investigation. Finally, the intervention results obtained to date correspond to the

following stages in standard control design: modeling, controller design, and verification of the performance of the designed controller *via computer simulations*. The designed controllers will have to be successfully implemented in practical studies, at least with cancer cell lines, before the benefits of using engineering approaches in translational medicine become transparent to the biological and medical communities. A considerable amount of effort needs to be focused on this endeavor.

ACKNOWLEDGMENTS

This work was supported by the National Cancer Institute, the National Human Genome Research Institute, the Translational Genomics Research Institute, the University of Texas M.D. Anderson Cancer Center, the Plant, Soil and Nutrition Laboratory of the USDA Agricultural Research Service, and the National Science Foundation under grants ECS-0355227 and CCF-0514644.

AUTHORS

Edward R. Dougherty is a professor in the Department of Electrical Engineering at Texas A&M University in College Station, Texas. He is director of the Genomic Signal Processing Laboratory at Texas A&M University and director of the Computational Biology Division of the Translational Genomics Research Institute in Phoenix, Arizona. He holds a Ph.D. in mathematics from Rutgers University and an M.S. in computer science from Stevens Institute of Technology. He is author of 12 books, editor of five others, and author of more than 170 journal papers. He is an SPIE Fellow and a recipient of the SPIE President's Award. He served as editor of the *Journal of Electronic Imaging* for six years. He has contributed extensively to the statistical design of nonlinear operators for image processing and the consequent application of pattern recognition theory to nonlinear image processing. His current research is focused on genomic signal processing, with the central goal being to model genomic regulatory mechanisms for the purposes of diagnosis and therapy.

Aniruddha Datta received the B.Tech. degree in electrical engineering from the Indian Institute of Technology, Kharagpur, in 1985, the M.S.E.E. degree from Southern Illinois University, Carbondale, in 1987, and the M.S. (applied mathematics) and Ph.D. degrees from the University of Southern California in 1991. In August 1991, he joined the Department of Electrical Engineering at Texas A&M University, where he is currently a professor. His areas of interest include adaptive control, robust control, PID control, and genomic signal processing. From 2001–2003, he worked on cancer research as a postdoctoral trainee under a National Cancer Institute (NCI) Training Grant. He is the author of *Adaptive Internal Model Control*, (Springer-Verlag, 1998), a coauthor of *Structure and Synthesis of PID Controllers* (Springer-Verlag, 2000), and a coauthor of *PID Controllers for Time Delay Systems* (Birkhauser, 2004). He was an associate editor of *IEEE Transactions on Automatic Control* and is an associate editor of *IEEE Transactions on Systems, Man and Cybernetics-Part B*. He is a Senior Member of the IEEE.

Chao Sima is a Ph.D. student under the supervision of Dr. E.R. Dougherty in the Department of Electrical Engineering at Texas A&M University in College Station. He received his B.E. degree in 1995 at Xi'an Jiaotong University, P.R. China. His current research interests include error estimation and application of pattern recognition in genomic signal processing.

REFERENCES

[1] E.R. Dougherty and A. Datta, "Genomic signal processing: Diagnosis and therapy," *IEEE Signal Processing Mag.*, vol. 22, no. 1, pp. 107–112, 2005.

[2] J. Chen, E.R. Dougherty, S. Demir, C. Friedman, C. Li, and S. Wong, "Grand challenges for multimodal bio-medical systems," *IEEE Circuits Syst. Mag.*, vol. 5, no. 2, pp. 46–52, 2005.

[3] I. Hedenfelk, D. Duggan, Y. Chen, M. Radmacher, M. Bittner, R. Simon, P. Meltzer, B. Gusterson, M. Esteller, M. Raffeld, Z. Yakhini, A. Ben-Dor, E. Dougherty, J. Kononen, L. Bubendorf, W. Fehrle, S. Pittaluga, S. Gruvverger, N. Loman, O. Johannsson, H. Olsson, B. Wifond, G. Sauter, O.P. Kallioniemi, B. Wilfond, A. Borg, and J. Trent., "Gene expression profiles in hereditary breast cancer," *New Eng. J. Med.*, vol. 344, pp. 539–548, 2001.

[4] M.J. van de Vijver, Y.D. He, L.J. van't Veer, H. Dai, A.A.M. Hart, D.W. Voskuil, G.J. Schreiber, J.L. Peterse, C. Roberts, M.J. Marton, M. Parrish, D. Atsma, A. Witteveen, A. Glas, L. Delahaye, T. van der Velde, H. Bartelink, S. Rodenhuis, E.T. Rutgers, S.H. Friend, and R. Bernards, "A gene-expression signature as a predictor of survival in breast cancer," *New Eng. J. Med.*, vol. 347, no. 25, pp. 1999–2009, 2002.

[5] E.R. Dougherty, "Small sample issues for microarray-based classification," *Comp. Funct. Genomics*, vol. 2, pp. 28–34, Feb. 2001.

[6] L. Devroye, L. Gyorfi, and G. Lugosi, *A Probablistic Theory of Pattern Recognition*. New York: Springer, 1996.

[7] V.N. Vapnik and A. Chervonenkis, "On the uniform convergence of relative frequencies of events to their probabilities," *Theory Prob. Applic.*, vol. 16, no. 2, pp. 264–280, 1971.

[8] V.N. Vapnik and A. Chervonenkis, *Theory of Pattern Recognition*. Moscow: Nauka, 1974.

[9] D.M. Titterington, "Common structure of smoothing techniques in statistics," *Int. Statist. Rev.*, vol. 53, no. 2, pp. 141–170, 1985.

[10] J.H. Friedman, "Regularized discriminant analysis," *J. Amer. Statist. Assoc.*, vol. 84, no. 405, pp. 165–175, 1989.

[11] M. Skurichina, R.P.W. Duin, and S. Raudys, "K-nearest neighbors noise injection in multilayer perceptron training," *IEEE Trans. Neural Networks*, vol. 11, no. 2, pp. 504–511, 2000.

[12] S. Kim, E.R. Dougherty, J. Barrera, Y. Chen, M. Bittner, and J.M. Trent, "Strong feature sets from small samples," *Comput. Biol.*, vol. 9, no. 1, pp. 127–146, 2002.

[13] C.M. Bishop, *Neural Networks for Pattern Recognition*. New York: Oxford Univ. Press, 1995.

[14] J. Sietsma and R.J.F. Dow, "Neural network pruning—Why and how," in *Proc. IEEE Int. Conf. Neural Networks*, 1998, vol. 1, pp. 325–333.

[15] V.N. Vapnik, *Estimation of Dependencies Based on Empirical Data*. New York: Springer-Verlag, 1982.

[16] G. Lugosi and A. Nobel, "Adaptive model selection using empirical complexities," *Ann. Statist.*, vol. 27, no. 6, pp. 1830–1864, 1999.

[17] P. Bartlett, S. Boucheron, and G. Lugosi, "Model selection and error estimation," *Mach. Learn.*, vol. 48, no. 1–3, pp. 85–113, 2002.

[18] J. Rissanen, "Modeling by shortest data description," *Automatica*, vol. 14, no. 5, pp. 465–471, 1978.

[19] I. Tabus and J. Astola, "On the use of MDL principle in gene expression prediction," *Appl. Signal Process.*, vol. 2001, no. 4, pp. 297–303, 2001.

[20] G.F. Hughes, "On the mean accuracy of statistical pattern recognizers," *IEEE Trans. Inform. Theory*, vol. 14, no. 1, pp. 55–63, 1968.

[21] A.K. Jain and B. Chandrasekaran, "Dimensionality and sample size considerations in pattern recognition practice," in *Classification, Pattern Recognition and Reduction of Dimensionality*, vol. 2, *Handbook of Statistics*, P.R. Krishnaiah, L.N. Kanal, Eds., Amsterdam: North-Holland, 1982, pp. 835–856.

[22] T. Cover and J. Van Campenhout, "On the possible orderings in the measurement selection problem," *IEEE Trans. Syst., Man, Cybern.*, vol. 7, no. 9, pp. 657–661, 1977.

[23] J. Hua, Z. Xiong, J. Lowey, E. Suh, and E.R. Dougherty, "Optimal number of features as a function of sample size for various classification rules," *Bioinform.*, vol. 21, no. 8, pp. 1509–1515, 2005.

[24] P.M. Narendra and K. Fukunaga, "A branch and bound algorithm for feature subset selection," *IEEE Trans. Comput.*, vol. 26, no. 9, pp. 917–922, 1977.

[25] P. Pudil, J. Novovicova, and J. Kittler, "Floating search methods in feature selection," *Pattern Recognit. Lett.*, vol. 15, no. 11, pp. 1119–1125, 1994.

[26] A.K. Jain and D. Zongker, "Feature selection—Evaluation, application, and small sample performance," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 153–158, 1997.

[27] M. Kudo and J. Sklansky, "Comparison of algorithms that select features for pattern classifiers," *Pattern Recognit.*, vol. 33, no. 1, pp. 25–41, 2000.

[28] I. Tabus, J. Rissanen, and J. Astola, "Classification and feature gene selection using the normalized maximum likelihood model for discrete regression," *Signal Process.*, vol. 83, no. 4, pp. 713–727, 2003.

[29] U.M. Braga-Neto and E.R. Dougherty, "Is cross-validation valid for small-sample microarray classification?," *Bioinform.*, vol. 20, no. 3, pp. 374–380, 2004.

[30] B. Efron, "Bootstrap methods: Another look at the jacknife," *Ann. Statist.*, vol. 7, no. 1, pp. 1–26, 1979.

[31] B. Efron, "Estimating the error rate of a prediction rule: Improvement on cross-validation," *J. Amer. Statist. Soc.*, vol. 78, no. 382, pp. 316–331, 1983.

[32] U.M. Braga-Neto and E.R. Dougherty, "Bolstered error estimation," *Pattern Recognit.*, vol. 37, no. 6, pp. 1267–1281, 2004.

[33] C. Sima, U.M. Braga-Neto, and E.R. Dougherty, "Superior feature-set ranking for small samples using bolstered error estimation," *Bioinform.*, vol. 21, no. 7, pp. 1046–1054, 2005.

[34] C. Sima, S. Attoor, U.M. Braga-Neto, J. Lowey, E. Suh, and E.R. Dougherty, "Impact of error estimation on feature-selection algorithms," *Pattern Recognit.*, to be published.

[35] O. Bousquet, S. Boucheron, and G. Lugosi, "Theory of classification: A survey of recent advances," *ESAIM Prob. Statist.*, to be published.

[36] S. Raudys and V. Pikelis, "On dimensionality, sample size, classification error, and complexity of classification algorithms in pattern recognition," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 2, no. 3, pp. 242–252, 1980.

[37] K. Fukunaga and R.R. Hayes, "Effects of sample size in classifier design," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 11, no. 8, pp. 873–885, 1989.

[38] H.R. McFarland and D. Richards, "Exact misclassification probabilities for plug-in normal quadratic discriminant functions II. The heterogeneous case," *Multivar. Anal.*, vol. 82, no. 2, pp. 299–330, 2002.

[39] J. Hua, Z. Xiong, and E.R. Dougherty, "Determination of the optimal number of features for quadratic discriminant analysis via the normal approximation to the discriminant distribution," *Pattern Recognit.*, vol. 38, no. 3, pp. 403–421, 2005.

[40] U.M. Braga-Neto and E.R. Dougherty, "Exact performance of error estimators for discrete classifiers," *Pattern Recognit.*, vol. 38, no. 11, pp. 1799–1814, 2005.

[41] J. Langford, "Tutorial on practical prediction theory for classification," *Machine Learn. Res.*, vol. 6, pp. 211–244, 2005.

[42] S. Mousses, U. Wagner, Y. Chen, J.W. Kim, L. Bubendorf, M. Bittner, T. Pretlow, A.G. Elkahloun, J.B. Trepel, and O.P. Kallioniemi, "Failure of hormone therapy in prostate cancer involves systematic restoration of androgen responsive genes and activation of rapamycin sensitive signaling," *Oncogene*, vol. 20, no. 46, pp. 6718–6723, 2001.

[43] T. Kobayashi, M. Yamaguchi, S. Kim, J. Morikawa, S. Ueno, E. Suh, E.R. Dougherty, I. Shmulevich, H. Shiku, and W. Zhang, "Gene expression profiling identifies strong feature genes that classify *de novo* $CD5^+$ and $CD5^-$ diffuse large B-cell lymphoma and mantle cell lymphoma," *Cancer Res.*, vol. 63, no. 1, pp. 60–66, 2003.

[44] A.K. Jain, M.N. Murty, and P.J. Flynn, "Data clustering: A review," *ACM Comput. Surv.*, vol. 31, no. 3, pp. 264–323, 1999.

[45] M.K. Kerr and G.A. Churchill, "Bootstrapping cluster analysis: Assessing the reliability of conclusions from microarray experiments," *Proc. Nat. Acad. Science*, vol. 98, no. 16, pp. 8961–8966, 2001.

[46] E.R. Dougherty, J. Barrera, M. Brun, S. Kim, R.M. Cesar, Y. Chen, M. Bittner, and J.M. Trent, "Inference from clustering with application to gene-expression microarrays," *Comput. Biol.*, vol. 9, no. 1, pp. 105–126, 2002.

[47] E.R. Dougherty and M. Brun, "A probabilistic theory of clustering," *Pattern Recognit.*, vol. 37, no. 5, pp. 917–925, 2004.

[48] D. Endy and R. Brent, "Modelling cellular behavior," *Nature*, vol. 409, no. 6818, pp. 391–395, 2001.

[49] J. Hasty, D. McMillen, F. Isaacs, and J. Collins, "Computational studies of gene regulatory networks: In numero molecular biology," *Nature Rev. Genetics*, vol. 2, no. 4, pp. 268–279, 2001.

[50] R. Somogyi and L.D. Greller, "The dynamics of molecular networks: Applications to therapeutic discovery," *Drug Discov. Today*, vol. 6, no. 24, pp. 1267–1277, 2001.

[51] M. Newman, "The structure and function of complex networks," *SIAM Rev.*, vol. 45, no. 2, pp. 167–256, 2003.

[52] S. Strogatz, "Exploring complex networks," *Nature*, vol. 410, no. 6825, pp. 268–276, 2001.

[53] H. de Jong, "Modelling and simulation of genetic regulatory systems: A literature review," *Comput. Biol.*, vol. 9, no. 1, pp. 67–103, 2002.

[54] S.A. Kauffman, "Metabolic stability and epigenesis in randomly constructed genetic nets," *Theor. Biol.*, vol. 22, pp. 437–467, 1969.

[55] S.A. Kauffman, "Homeostasis and differentiation in random genetic control networks," *Nature*, vol. 224, no. 215, pp. 177–178, 1969.

[56] S.A. Kauffman, *The Origins of Order: Self-Organization and Selection in Evolution*. New York: Oxford Univ. Press, 1993.

[57] S. Huang, "Gene expression profiling, genetic networks, and cellular states: An integrating concept for tumorigenesis and drug discovery," *Molec. Med.*, vol. 77, no. 6, pp. 469–480, 1999.

[58] R. Pal, A. Datta, A.J. Fornace, M.L. Bittner, and E.R. Dougherty, "Boolean relationships among genes responsive to ionizing radiation in the NCI 60 ACDS," *Bioinform.*, vol. 21, no. 8, pp. 1542–1549, 2005.

[59] M. Schena, D. Shalon, R. Davis, and P.O. Brown, "Quantitative monitoring of gene expression patterns with a complementary DNA microarray," *Science*, vol. 270, no.5235, pp. 467–470, 1995.

[60] E.R. Dougherty, S. Kim, and Y. Chen, "Coefficient of determination in nonlinear signal processing," *Signal Process.*, vol. 80, no. 10, pp. 2219–2235, 2000.

[61] I. Shmulevich, E.R. Dougherty, S. Kim, and W. Zhang, "Probabilistic Boolean networks: A rule-based uncertainty model for gene regulatory networks," *Bioinform.*, vol. 18, no. 2, pp. 261–274, 2002.

[62] I. Shmulevich, E.R. Dougherty, and W. Zhang, "From Boolean to probabilistic boolean networks as models of genetic regulatory networks," *Proc. IEEE*, vol. 90, no. 11, pp. 1778–1792, 2002.

[63] I. Shmulevich, E.R. Dougherty, and W. Zhang, "Gene perturbation and intervention in probabilistic Boolean networks," *Bioinform.*, vol. 18, no. 10, pp. 1319–1331, 2002.

[64] E.R. Dougherty and I. Shmulevich, "Mappings between probabilistic Boolean networks," *Signal Process.*, vol. 83, no. 4, pp. 799–809, 2003.

[65] I. Ivanov and E.R. Dougherty, "Reduction mappings between probabilistic Boolean networks," *Appl. Signal Process.*, vol. 1, no. 1, pp. 125–131, 2004.

[66] H. Lahdesmaki, I. Shmulevich, and O. Yli-Haria, "On learning gene regulatory networks under the Boolean network model," *Machine Learn.*, vol. 52, no. 1–2, pp. 147–167, 2003.

[67] I. Shmulevich, A. Saarinen, O. Yli-Harja, and J. Astola, "Inference of genetic regulatory networks under the best-fit extension paradigm," in *Computational and Statistical Approaches to Genomics*, W. Zhang and I. Shmulevich, Eds. Boston: Kluwer, 2002, pp. 197–210.

[68] X. Zhou, X. Wang, and E.R. Dougherty, "Construction of genomic networks using mutual-information clustering and reversible-jump Markov-chain-Monte-Carlo predictor design," *Signal Process.*, vol. 83, no. 4, pp. 745–761, 2003

[69] R.F. Hashimoto, S. Kim, I. Shmulevich, W. Zhang, M.L. Bittner, and E.R. Dougherty, "Growing genetic regulatory networks from seed genes," *Bioinform.*, vol. 20, no. 8, pp. 1241–1247, 2004.

[70] M. Bittner, P. Meltzer, Y. Chen, Y. Jiang, E. Seftor, M. Hendrix, M. Radmacher, R. Simon, Z. Yakhini, A. Ben-Dor, N. Sampas, E.R. Dougherty, E. Wang, F. Marincola, C. Gooden, J. Leuders, A. Glatfelter, P. Pollock, J. Carpten, E. Gillanders, D. Leja, K. Dietrich, C. Beaudry, M. Berens, D. Alberts, and V. Sondak, "Molecular classification of cutaneous malignant melanoma by gene expression profiling," *Nature*, vol. 406, no. 6795, pp. 536–540, 2000.

[71] A.T. Weeraratna, Y. Jiang, G. Hostetter, K. Rosenblatt, P. Duray, M. Bittner, and J.M. Trent, "Wnt5a signalling directly affects cell motility and invasion of metastatic melanoma," *Cancer Cell*, vol. 1, no. 3, pp. 279–288, 2002.

[72] S. Kim, H. Li, E.R. Dougherty, N. Cao, Y. Chen, M. Bittner, and E. Suh, "Can Markov chain models mimic biological regulation?" *Biol. Syst.*, vol. 10, no. 4, pp. 437–458, 2002.

[73] X. Zhou, X. Wang, R. Pal, I. Ivanov, M. Bittner, and E.R. Dougherty, "A Bayesian connectivity-based approach to constructing probabilistic gene regulatory networks," *Bioinform.*, vol. 20, no. 17, pp. 2918–2927, 2004.

[74] R. Pal, I. Ivanov, A. Datta, and E.R. Dougherty, "Generating Boolean networks with a prescribed attractor structure," to be published.

[75] E. Dougherty and Y. Xiao, "Design of probabilistic gene networks under the requirement of contextual data consistency," submitted for publication.

[76] N. Friedman, M. Linial, I. Nachman, and D. Pe'er, "Using Bayesian networks to analyze expression data," *Comput. Biol.*, vol. 7, no. 3–4, pp. 601–620, 2000.

[77] P. Pe'er, A. Regev, G. Elidan, and N. Friedman, "Inferring subnetworks from perturbed expression profiles," *Bioinform.*, vol. 17, pp. 215–224, 2001.

[78] D. Husmeier, "Sensitivity and specificity of inferring genetic regulatory interactions from microarray experiments with dynamic Bayesian networks," *Bioinform.*, vol. 19, no. 17, pp. 2271–2282, 2003.

[79] H. Lahdesmaki, S. Hautaniemi, I. Shmulevich, and O. Yli-Harja, "Relationships between probabilistic Boolean networks and dynamic Bayesian networks as models of gene regulatory networks," *Signal Process.*, to be published.

[80] I. Shmulevich, E.R. Dougherty, and W. Zhang, "Control of stationary behavior in probabilistic Boolean networks by means of structural intervention," *Biol. Syst.*, vol. 10, no. 4, pp. 431–446, 2002.

[81] A. Datta, A. Choudhary, M.L. Bittner, and E.R. Dougherty, "External control in Markovian genetic regulatory networks," *Machine Learn.*, vol. 52, no. 1–2, pp. 169–191, 2003.

[82] A. Datta, A. Choudhary, M.L. Bittner, and E.R. Dougherty, "External control in markovian genetic regulatory networks: The imperfect information case," *Bioinform.*, vol. 20, no. 6, pp. 924–930, 2004.

[83] R. Pal, A. Datta, M.L. Bittner, and E.R. Dougherty, "Intervention in context-sensitive probabilistic Boolean networks," *Bioinform.*, vol. 21, no. 7, pp. 1211–1218, 2005.

[84] A. Choudhary, A. Datta, M.L. Bittner, and E.R. Dougherty, "Assignment of terminal penalties in controlling genetic regulatory networks," in *Proc. American Control Conf.*, 2005, pp. 417–422.

[85] R.E. Kalman, "Canonical structure of linear dynamical systems," in *Proc. Nat. Acad. Sci.*, 1962, vol. 48, pp. 596–600. SP