Complex Function Sets Improve Symbolic Discriminant Analysis of Microarray Data

David M. Reif[1], Bill C. White[1], Nancy Olsen[2], Thomas Aune[2], and Jason H. Moore[1]

[1]Program in Human Genetics, Department of Molecular Physiology and Biophysics
Vanderbilt University, Nashville, TN, USA
{Reif,BWhite,Moore}@phg.mc.Vanderbilt.edu
[2 1]Program in Human Genetics, Department of Medicine
Vanderbilt University, Nashville, TN, USA 37232-0700
{Nancy.Olsen,Thomas.Aune}@Vanderbilt.edu

Abstract. Our ability to simultaneously measure the expression levels of thousands of genes in biological samples is providing important new opportunities for improving the diagnosis, prevention, and treatment of common diseases. However, new technologies such as DNA microarrays are generating new challenges for variable selection and statistical modeling. In response to these challenges, a genetic programming-based strategy called symbolic discriminant analysis (SDA) for the automatic selection of gene expression variables and mathematical functions for statistical modeling of clinical endpoints has been developed. The initial development and evaluation of SDA has focused on a function set consisting of only the four basic arithmetic operators. The goal of the present study is to evaluate whether adding more complex operators such as square root to the function set improves SDA modeling of microarray data. The results presented in this paper demonstrate that adding complex functions to the terminal set significantly improves SDA modeling by reducing model size and, in some cases, reducing classification error and runtime. We anticipate SDA will be an important new evolutionary computation tool to be added to the repertoire of methods for the analysis of microarray data.

1 Introduction

Biomedicine is at a critical inflection point in the relationship between the amount of data it is possible to collect and our understanding of that data. For the first time in history, we are undergoing an information explosion and an understanding implosion. That is, new technologies are making it possible to collect data at a much faster rate than we can understand it. For example, DNA microarray technology facilitates the simultaneous measurement of the expression levels of tens of thousands of genes in biological samples [1]. As a result of this explosion of genetic information, bioinformatics and computational biology are faced with two important challenges. First, what are the most appropriate statistical and computational modeling approaches for relating gene expression data with clinical endpoints? Second, what are the most appropriate computational search strategies for identifying combinations of gene expression variables from an effectively infinite search space?

E. Cantú-Paz et al. (Eds.): GECCO 2003, LNCS 2724, pp. 2277–2287, 2003.

In response to these challenges, Moore et al. [2,3] have developed a machine learning strategy called symbolic discriminant analysis (SDA) for the automatic selection of gene expression variables and mathematical functions for statistical modeling of clinical endpoints. One advantage of this approach is that it uses the parallel search features of genetic programming [4], or GP, that are desirable for microarray data analysis [5]. Another important advantage is that no *a priori* assumptions are made about the functional form of the statistical model. This is important if the relationships between gene expression variables and clinical endpoints are complex and nonlinear, with interactions playing a more important role than the independent main effects of each gene. It is anticipated that such complex interactions among genes are common and play an important role in susceptibility to multifactorial diseases [6,7]. Indeed, applications of SDA to identifying patterns of gene expression that are associated with human leukemia and autoimmune diseases has revealed nonadditive relationships between gene expression variables that would not have been identified using a parametric statistical approach such as linear discriminant analysis [2,3,8].

The initial development of SDA has focused on a function set consisting of only four arithmetic functions. The goal of the present study is to evaluate whether adding complex functions such as square root to the terminal set improves SDA modeling. Adding complex functions will be considered an improvement if 1) they significantly decrease classification error of SDA models, 2) they significantly reduce the size of SDA models in terms of the overall number of nodes and the node depth, and/or 3) they significantly reduce the computational time required to complete a certain number of GP generations. For this study, we evaluate the addition of complex functions using real microarray data from human autoimmune disease that has been previously analyzed using SDA with just arithmetic functions [3]. We begin in Section 2 with an overview of the SDA approach. In Section 3, we provide an overview of the microarray data. In Section 4, we describe the details of the GP. In Section 5, we describe the experimental design and statistical analysis. A summary and discussion of the results are presented in Sections 6 and 7 respectively. The results presented in this paper demonstrate that adding complex functions to the terminal set significantly reduces the size of SDA models and, in some cases, reduces the GP runtime and classification error.

2 An Overview of Symbolic Discriminant Analysis

2.1 Introduction to Symbolic Discriminant Analysis

An important limitation of parametric statistical approaches such as linear discriminant analysis and logistic regression is the need to pre-specify the functional form of the model. To address this limitation, Moore et al. [2,3] developed SDA for automatically identifying the optimal functional form and coefficients of discriminant functions that may be linear or nonlinear. This is accomplished by providing a list of mathematical functions and a list of explanatory variables that can be used to build discriminant scores. Similar to symbolic regression [4], GP is utilized to perform a

parallel search for a combination of functions and variables that optimally discriminate between two endpoint groups. GP permits the automatic discovery of symbolic discriminant functions that can take any form defined by the mathematical operators provided. GP builds symbolic discriminant functions using expression trees. Each expression tree has a mathematical function at the root node and all other nodes. Terminals in the expression tree are comprised of gene expression variables and constants. The primary advantage of this approach is that the functional form of the statistical model does not need to be pre-specified. This is important for the identification of combinations of expressed genes whose relationship with the endpoint of interest may be non-additive or nonlinear [6,7].

In its first implementation, SDA used leave one out cross-validation (LOOCV) to estimate the classification and prediction error of SDA models [2]. With LOOCV, each subject is systematically left out of the SDA analysis as an independent data point (i.e. the testing set) used to assess the predictive accuracy of the SDA model. Thus, SDA is run on a subset of the data (i.e. the training set) comprised of n-1 subjects. The model that classifies subjects in the training set with minimum error is selected and then used to predict the group membership of the single independent testing subject. This is repeated for each of the possible training sets yielding n SDA models. Moore et al. [2] selected LOOCV because it is an unbiased estimator of model error [9]. However, it should be noted that LOOCV may have a large variance due to similarity of the training datasets [9,10] and the relatively small sample sizes that are common with microarray experiments. It is possible to reduce the variance using perhaps 5-fold or 10-fold cross-validation. However, these procedures may lead to biased estimates and may not be practical when the sample size is small [9].

In this first implementation of SDA, models were selected that had low classification and prediction errors as evaluated using LOOCV. The end result of this initial implementation of SDA is an ensemble of models from separate cross validation divisions of the data. In fact, over k runs of n-fold cross validation, there are a maximum of kn possible models generated assuming a 'best' model for each interval is identifiable. This is a common result when GP is used with cross-validation methods because of its stochastic elements. In response to this, Moore et al. [3] and Moore [8] developed a strategy for SDA modeling that involves evaluating the consistency with which each gene was identified across each of the LOOCV trials. This new statistic is referred to as cross validation consistency (CVC) and is similar to the approach taken by Ritchie et al. [11] for the identification of gene-gene interactions in epidemiological study designs. The idea here is that genes that are important for discriminating between biological or clinical endpoint groups should consistently be identified regardless of the LOOCV dataset. The number of times each gene is identified is counted and this value compared to the value expected 5% of the time by chance were the null hypothesis of no association true. This empirical decision rule is established by permuting the data 1,000 or more times and repeating SDA analysis on each permuted dataset as described above. In this manner, a list of statistically significant genes derived from SDA can be compiled.

Once a list of statistically significant genes or variables is compiled, a symbolic discriminant function that maximally describes the entire dataset can then be derived.

This is accomplished by rerunning SDA multiple times on the entire dataset using only the list of statistically significant candidate genes identified in the CVC analysis. A symbolic discriminant function that maximizes the distance between distributions of symbolic discriminant scores between the two endpoint groups is selected. In this manner, a single 'best' symbolic discriminant function can be identified and used for prediction in independent datasets. This modeling process is designed to limit false-positive results and provide an objective way of dealing with different GP results in different cross validation divisions of the data. Moore [8] has suggested this approach may be useful for other GP-based modeling strategies.

2.2 Advantages of Symbolic Discriminant Analysis

As discussed by Moore et al. [3], there are two important advantages of SDA over traditional multivariate methods such as linear discriminant analysis [12-14]. First, SDA does not pre-specify the functional form of the model. For example, with linear discriminant analysis, the discriminant function must take the form of an additive linear equation. This limits the models to linear additive functions of the explanatory variables. With SDA, the basic mathematical building blocks are defined and then flexibly combined with explanatory variables to derive the best discriminant function.

The second advantage of SDA is the automatic selection of variables from a list of thousands. Traditional model fitting involves stepwise procedures that enter a variable into the model and then keep it in the model if it has statistically significant marginal or independent main effect [15]. Interaction terms are only evaluated for those variables that are already in the model. This deals with the combinatorial problem of selecting variables, however, variables whose effects are primarily through interactions with other variables will be missed. This may be an unreasonable assumption for most complex biological systems. The SDA approach employs a parallel machine learning approach to selecting variables that permits interactions to be modeled in the absence of marginal effects. For example, both of the SDA models of human autoimmune disease identified by Moore et al. [3] involve multiplicative relationships among the gene expression variables (see Figure 1 below). No one variable contributes to the discriminant function independently of the others. For comparison, Moore et al. [3] analyzed the data using stepwise LDA [14]. Only one of the genes was identified by both SDA and LDA, and this was one of the genes that had a statistically significant main effect in a univariate analysis. Thus, SDA identified a set of genes that was not identified by LDA.

2.3 Disadvantages of Symbolic Discriminant Analysis

Although SDA has several important advantages over traditional multivariate statistical methods, there are several disadvantages [3]. First, there is no guarantee that GP will find the optimal solution. Heuristic searches tend to sacrifice finding an optimal solution in favor of tractability [16]. Implementing an evolutionary algorithm in parallel certainly improves the chances of finding an optimal solution [17], but it is not a certainty. This is due to the stochastic nature of GP. The initial populations of solutions are randomly generated and the recombination and mutation occurs at ran-

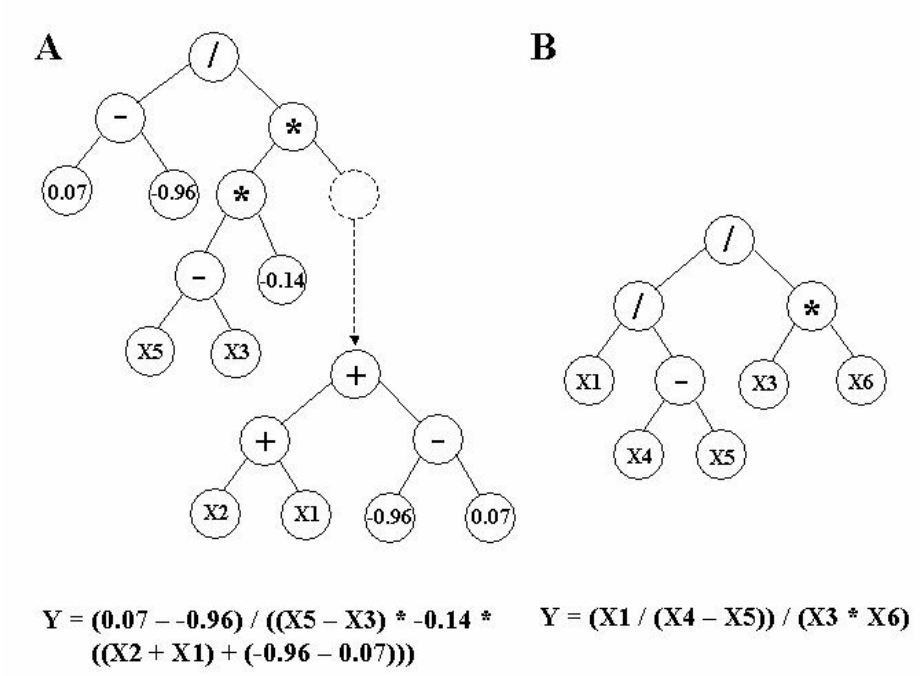

Fig. 1. Symbolic discriminant functions of gene expression variables in the form of expression trees and mathematical equations for discriminating rheumatoid arthritis from normal (A) and systemic lupus erythematosus from normal (B) [3].

dom positions in the binary expression trees. Further, there may be a stochastic component to how the highest fit individuals are selected. For these reasons, evolutionary algorithms should be run multiple times with multiple parallel populations.

A second disadvantage of this approach is the computational requirement. LDA can be performed on a standard desktop workstation while SDA requires a parallel computer cluster for optimal performance. When the number of genes to be evaluated is large, the power of a parallel computer is required. Although such systems are fairly inexpensive to build, the time investment to establish and manage a parallel computing farm may be prohibitive to some.

A third disadvantage is the complexity of the symbolic discriminant functions obtained. An attractive feature of LDA is the simplicity of the models, which facilitates interpretation. SDA has the potential to generate rather large models.

The goal of the present study is to evaluate whether adding more complex functions to the terminal set overcomes some of the disadvantages of SDA described by Moore et al. [3]. Specifically, does SDA with complex functions find better models faster? Also, are the SDA models identified smaller?

3 An Overview of the Microarray Data Used

A detailed description of the data is provided by Maas et al. [20]. Briefly, gene expression was measured in peripheral blood mononuclear cells (PBMC) from normal individuals (n=12) and patients with rheumatoid arthritis (RA) (n=7) or systemic lupus erythematosus (SLE) (n=9) using cDNA microarrays. Normal individuals (i.e. controls) were examined before and after routine immunization with flu vaccine to allow comparison of normal immune response genes to those that are differentially regulated in autoimmune disease. Reproducibility of the experimental method was first established by performing four hybridizations to separate microarrays using the same RNA sample. Data were normalized against a common control and linear regression analysis was utilized to estimate reproducibility. All hybridizations were highly reproducible with R^2 values ranging from 0.87 to 0.99. In the present study, we used SDA to identify symbolic discriminant functions that differentiate RA from normal and SLE from normal.

4 Details of the Genetic Programming (GP) Strategy

We used genetic programming or GP [4] in the present study to optimize the selection of gene expression variables and mathematical operators for building symbolic discriminant functions. We implemented the GP optimization of SDA using the lil-gp software package [18] modified to operate in parallel using the parallel virtual machine (PVM) message-passing library. The parallel GP was run on two processors of a 110-processor Beowulf-style parallel computer system running the Linux operating system. Each node has two Pentium III 600Mhz processors, 256 Mb RAM, a network card, and a 10 Gb hard drive. A total of two demes were used each consisting of 100 individuals for total population size of 200. We allowed the GP to run a total of 100 iterations with migration between each population every 25 iterations. A recombination frequency of 0.9 was used along with a reproduction rate of 0.1. Table 1 summarizes the GP parameters.

Table 1. The GP parameter settings for the SDA analyses.

Objective	Identify optimal SDA models
Fitness function	Classification error
Number of runs	100 per dataset
Stopping criteria	Classification error = 0
Population size	200
Number of demes	2
Generations	100
Selection	Fitness proportionate
Crossover probability	0.9
Reproduction probability	0.1

5 Experimental Design and Data Analysis

SDA was run with and without complex functions in two study designs. In the first study design, we compared normal samples and rheumatoid arthritis (RA) samples. In the second study design, we compared normal samples and systemic lupus erythematosus (SLE) samples. For each of the two study designs, we ran SDA a total of 100 times with just the basic arithmetic operators in the terminal set (+, -, *, /). Next, we ran SDA a total of 100 times for each comparison with a set of complex functions in addition to the basic arithmetic operators in the terminal set. The complex function set included square, square root, logarithm, exponential, absolute value, sine, and cosine. The goal of the statistical analysis was to determine whether the addition of more complex functions to the terminal set 1) significantly decreases the classification error of SDA models as assessed by leave one out cross validation (LOOCV), 2) significantly reduces the size of SDA models in terms of the overall number of nodes and the node depth, and/or 3) significantly reduces the computational time required to complete a certain number of GP generations. Across the 100 runs in each study design, we specifically compared the average classification error as assessed by LOOCV, the average runtime of the GP, the average node count, and the average node depth. In addition, we compared the average number of gene expression variables, constants, and mathematical functions in the best SDA models. A one-tailed Student's t-test was used to test the null hypothesis that complex functions do not decrease classification error, reduce SDA model size, or reduce computation time. When the assumptions of normality or equal variance were violated, a nonparametric Wilcoxon rank-sum test was carried out. All results were considered statistically significant at the 0.05 level.

6 Results

Table 2 summarizes the average classification error, GP runtime, node count, node depth, number of gene expression variables, number of constants, and number of mathematical functions in SDA models with and without complex functions for the RA vs. normal comparison. Table 3 summarizes the same information for the SLE vs. normal comparison. For both comparisons, the majority of measures were significantly improved with the addition of complex functions to the terminal set. For the RA vs. normal comparison, only classification error was not improved. For the SLE vs. normal comparison, only GP runtime was not improved. The most striking difference is that the SDA models are significantly smaller when complex functions are used.

Table 2. Comparison of mean SDA performance measures for modeling with and without complex functions in the terminal set (RA vs. normal).

Performance Measure	Arithmetic Functions		Complex Functions		p-value
	n	mean	n	mean	
Classification error	100	<.01	100	<.01	.5656
GP runtime (sec.)	100	67.04	100	66.80	<.0001
Node count	100	37.79	100	17.31	<.0001
Node depth	100	6.71	100	7.64	<.0001
Number of variables	100	4.83	100	2.37	<.0001
Number of constants	100	17.21	100	4.65	<.0001
Number of functions	100	16.21	100	10.17	<.0001

Table 3. Comparison of mean SDA performance measures for modeling with and without complex functions in the terminal set (SLE vs. normal).

Performance Measure	Arithmetic Functions		Complex Functions		p-value
	n	mean	n	mean	
Classification error	100	.06	100	.03	<.0001
GP runtime (sec.)	100	73.05	100	73.09	.5372
Node count	100	40.46	100	24.51	<.0001
Node depth	100	6.95	100	9.72	<.0001
Number of variables	100	5.51	100	4.10	<.0001
Number of constants	100	18.21	100	6.28	<.0001
Number of functions	100	17.21	100	14.07	<.0001

7 Discussion

Symbolic discriminant analysis (SDA) was developed as an attempt to deal with the challenges of selecting subsets of gene expression variables and features that facilitate the classification and prediction of biological and clinical endpoints [2,3,8]. Motivation for the development of SDA came from the limitations of traditional parametric approaches such as linear discriminant analysis and logistic regression. Application of SDA to high-dimensional microarray data from several human diseases has demonstrated that this approach is capable of identifying biologically relevant genes from among thousands of candidates. Further, because SDA makes no assumptions about the functional form of the model, it is capable of modeling complex nonlinear relationships between gene expression variables and clinical endpoints [2,3,8]. The initial studies using SDA used only the four basic arithmetic functions in the terminal set. The goal of the present study is to determine whether adding more complex functions such as logarithm and square root improve SDA modeling of microarray data. Adding complex function was considered an improvement if 1) they significantly decrease the classification error of SDA models as assessed by leave one out cross validation, 2) they significantly reduce the size of SDA models in terms of the overall

number of nodes and the node depth, and/or 3) they significantly reduce the computational time required to complete a certain number of GP generations. The results presented in this paper demonstrate that adding complex functions to the terminal set significantly improves SDA modeling by reducing model size and, in some cases, reducing classification error and runtime. Thus, the primary conclusion of this study is that a richer set of functions should be included in the terminal set.

The results of this study address several of the primary disadvantages of the SDA approach outlined in Section 2.3 and by Moore et al. [3]. First, the finding that adding complex functions reduces model size and runtime is important since a primary concern is that the search space is rugged and effectively infinite [5]. Further, there is no guarantee that GP will identify an optimal solution. Thus, anything that can be done to reduce the size of the search space without compromising the flexibility and power of the modeling approach is desirable. The availability of complex functions in the terminal set reduces the search space that would have been necessary to construct the same complex functions from the basic arithmetic operators. In both comparisons, the average number of nodes in the expression trees of the best SDA models was reduced by approximately 50%.

The results of this study also address the disadvantage of increased computational time associated with SDA modeling. It is certainly true that SDA is more computationally intensive than methods such as LDA. In the RA versus normal comparison, the GP runtime was significantly less when complex functions were used. A time saving of even seconds for a single run can be very important when computational methods such as permutation testing are used. Indeed, Moore et al. [3] propose permutation testing as a strategy for carrying our formal hypothesis testing with the SDA approach. This entails running SDA 1000 or more times on randomized datasets to create an empirical distribution of the test statistic being used under the null hypothesis of no association. Thus, a time savings of even one second per run would save 1000 seconds or approximately 16.7 minutes. A time savings of one minute would save 16.7 hours. These time savings could be very important, especially if computational resources are at a minimum or if many such runs need to performed on many different datasets.

The final disadvantage addressed by this study is model complexity. As with any GP-based modeling procedure, SDA models can be quite large and complicated. We demonstrated that adding complex functions significantly reduced model size in both comparisons. Smaller models, even with more complex functions, are easier to interpret because there are fewer variable interrelationships that need to be interpreted. Indeed, in both comparisons, the average number of variables included in the optimal SDA models was reduced by approximately 25-50% with the addition of complex functions. Further, the average number of constants and functions was also significantly reduced.

In the present study, we only carried out the very first few steps in the SDA modeling process outlined in Section 2 and by Moore et al. [3] in order to compare the features of the models during cross validation. In future studies, we will carry out the full SDA analysis of this dataset and others using the more complex functions. This will allow us to compare the final model obtained with those found previously (see

Figure 1 for example). Additionally, it will be very important to assess the prediction error of models with complex functions. This will be possible when multiple, independently collected datasets are available from the same study. We anticipate SDA models with complex functions will have a lower prediction error because the overall models are likely to be smaller and simpler. However, this is a working hypothesis that still needs to be tested.

As suggested by Moore and Parker [5], GP-based modeling strategies are expected to have a large impact on the statistical and computational analysis of high-dimensional datasets derived from high-throughput technologies such as DNA microarrays. The inherent parallel or beam search strategy used by GP may be necessary for traversing effectively infinite rugged search spaces. Indeed, GP is finding its way into a number of bioinformatics and computational biology studies [19]. This study significantly improves SDA modeling through the addition of complex functions to the terminal set. We anticipate SDA will continue to be an important methodology for the analysis of microarray data.

Acknowledgements. This work was supported by generous funds from the Robert J. Kleberg, Jr. and Helen C. Kleberg Foundation. This work was also supported by the Arthritis Foundation and National Institutes of Health grants AR02027, AR41943, DK58749, HL68744, CA90949, CA95103, and CA98131.

References

1. Schena, M., Shalon, D., Davis, R.W., Brown, P.O.: Quantitative monitoring of gene expression patterns with a complementary DNA microarray. Science 270 (1995) 467–470
2. Moore, J.H., Parker, J.S., Hahn, L.W.: Symbolic discriminant analysis for mining gene expression patterns. In: De Raedt, L., Flach, P. (eds) Lecture Notes in Artificial Intelligence 2167, pp 372–81, Springer-Verlag, Berlin (2001)
3. Moore, J.H., Parker, J.S., Olsen, N., Aune, T. Symbolic discriminant analysis of microarray data in autoimmune disease. Genetic Epidemiology 23 (2002) 57–69
4. Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. The MIT Press, Cambridge London (1992)
5. Moore, J.H., Parker, J.S.: Evolutionary computation in microarray data analysis. In: Lin, S. and Johnson, K. (eds): Methods of Microarray Data Analysis. Kluwer Academic Publishers, Boston (2001)
6. Templeton, A.R.: Epistasis and complex traits. In: Wade, M., Brodie III, B., Wolf, J. (eds.): Epistasis and Evolutionary Process. Oxford University Press, New York (2000)
7. Moore, J.H., Williams, S.M.: New strategies for identifying gene-gene interactions in hypertension. Annals of Medicine 34 (2002) 88–95
8. Moore, J.H.: Cross validation consistency for the assessment of genetic programming results in microarray studies. In: Raidl, G. et al. (eds) Lecture Notes in Computer Science 2611, in press, Springer-Verlag, Berlin (2003).
9. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, New York (2001)

10. Devroye, L., Gyorfi, L., Lugosi, G.: A Probabilistic Theory of Pattern Recognition. Springer-Verlag, New York (1996)
11. Ritchie, M.D., Hahn, L.W., Roodi, N., Bailey, L.R., Dupont, W.D., Plummer, W.D., Parl, F.F. and Moore, J.H.: Multifactor dimensionality reduction reveals high-order interactions among estrogen metabolism genes in sporadic breast cancer. American Journal of Human Genetics 69 (2001) 138–147
12. Fisher, R.A.: The Use of Multiple Measurements in Taxonomic Problems. Ann. Eugen. 7 (1936) 179–188
13. Johnson, R.A., Wichern, D.W.: Applied Multivariate Statistical Analysis. Prentice Hall, Upper Saddle River (1998)
14. Huberty, C.J.: Applied Discriminant Analysis. John Wiley & Sons, Inc., New York (1994)
15. Neter, J., Wasserman, W., Kutner, M.H.: Applied Linear Statistical Models, Regression, Analysis of Variance, and Experimental Designs. 3rd edn. Irwin, Homewood (1990)
16. Langley, P.: Elements of Machine Learning. Morgan Kaufmann Publishers, Inc., San Francisco (1996)
17. Cantu-Paz, E.: Efficient and Accurate Parallel Genetic Algorithms. Kluwer Academic Publishers, Boston (2000)
18. http://garage.cps.msu.edu/software/software-index.html
19. Fogel, G.B., Corne, D.W.: Evolutionary Computation in Bioinformatics. Morgan Kaufmann Publishers, Inc., San Francisco (2003)
20. Maas, K., Chan, S., Parker, J., Slater, A., Moore, J.H., Olsen, N., and Aune, T.M.: Cutting edge: molecular portrait of human autoimmunity. Journal of Immunology 169 (2002) 5–9