Genomic computing: explanatory modelling for functional genomics

Richard J. Gilbert
Institute of Biological Sciences, Cledwyn Building, University of Wales, Aberystwyth SY23 3DD
UK
rcg@aber.ac.uk
Phone +44 1970 622353
http://gepasi.dbs.aber.ac.uk/rcg/

Jem J. Rowland
Dept. of Computer Science, University of Wales, Aberystwyth SY23 3DB
UK
jjr@aber.ac.uk
Phone +44 1970 622445
http://www.aber.ac.uk/~jjr/

Douglas B. Kell
Institute of Biological Sciences, Cledwyn Building, University of Wales, Aberystwyth SY23 3DD
UK
dbk@aber.ac.uk
Phone +44 1970 622334
http://gepasi.dbs.aber.ac.uk/

Abstract

Many newly discovered genes are of unknown function. DNA microarrays are a method for determining the expression levels of all genes in an organism for which a complete genome sequence is available. By comparing the expression changes under different conditions it should be possible to assign functions to these genes. However, many hundreds of thousands of data points may be produced over a series of experiments. Genetic programming provided simple explanatory rules for gene function from such datasets, where previous approaches had not succeeded.

1 INTRODUCTION

The greatest challenge to biological science over the next decade will probably be in the field of functional genomics. Now that the DNA sequences of entire genomes are becoming available, we are discovering how incomplete is our knowledge of biological systems. When the total number of genes now known to exist in a genome are taken into consideration, we find that the genes which have well-defined functions are very much in the minority. The purpose of functional genomics is to determine the biological roles of the many previously unknown genes (Bork *et al*., 1998; Brent, 2000; Dyer, Cohen & Herrling, 1999; Hieter & Boguski, 1997; Oliver, 1996; Oliver *et al*., 1998).

There has been an explosion in the capabilities of the experimental techniques available to molecular biologists for the study of functional genomics. Advances in DNA sequencing and the biomolecular analysis of whole-cells and cell extracts has led to the generation of data sets the size and complexity of which easily defy conventional analysis. There is now a crucial need for new methods of computational analysis able to extract meaningful information from these types of data, to enable biologists to take full advantage of the new experimental technologies.

One of the most critical features required of a numerical analysis method for functional genomics is that it is **explanatory**, *i.e.* that the information it uncovers is readily apparent to the investigator. With the exception of rule induction methods, most learning methods based on (*e.g.*) statistical and neural computing approaches (Weiss & Kulikowski , 1991) do not provide explanatory models of the type required for a functional genomics study. Whilst rule induction methods are able to generate explanatory models, they do not in general scale well with increasing data set dimensionality. Genetic programming (GP) has been shown to be able to provide explanatory models for datasets with extremely high dimensionality (Koza, 1992; Koza, 1994; Koza *et al*., 1999), and so was chosen to perform the functional genomics analysis presented here.

2 DNA MICROARRAY HYBRIDISATION

DNA microarray hybridisation is a new technology which provides biologists with the ability to measure the expression levels of many thousands of genes in a single experiment. Each data point from a DNA microarray hybridisation experiment represents the ratio of the (transcriptional) expression levels of a particular gene under two or more different experimental conditions. For example, cells grown in media lacking the amino acid tryptophan would be expected to show different expression levels in the genes responsible for tryptophan metabolism as compared to cells grown in complete media. DNA microarray hybridisation allows the differences in expression levels to be quantified for every gene in the cells, potentially under many hundreds of different experimental conditions, and so provides the sort of information which will lead to the characterisation of genes which currently have unknown functions. However, this useful information is frequently obscured by the vast complexity of the data generated by this type of experiment.

A microarray hybridisation experiment is performed on a "DNA chip" containing many hundreds or thousands of

individual spots. Short, single-stranded DNA molecules are attached to the chip surface, either using a process akin to the photolithographic manufacture of semiconductors (de Saizieu *et al*., 1998; Lipshutz *et al*., 1999; Lockhart *et al*., 1996) or (more conveniently) by directly spotting DNA molecules to produce an ordered array (Brown & Botstein, 1999; DeRisi, Iyer & Brown, 1997). Each spot is designed to contain a unique sequence, complementary to an individual gene from the genome of the organism being studied. A microarray chip therefore comprises a grid of spots, each of which hopefully binds DNA whose sequence matches a single gene (Figure 1).

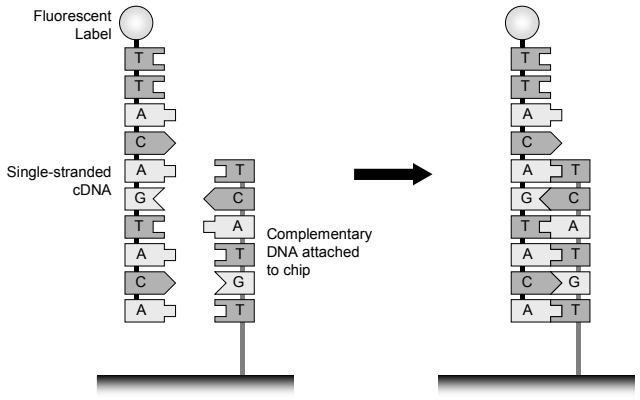


Figure 1: DNA microarray molecular recognition.

When a gene is expressed within a cell, the information it encodes is copied from the genomic DNA to intermediary molecules called messenger RNA (mRNA) which are subsequently translated into the functional protein molecules. Amongst other things, this is an amplification mechanism: a single-copy gene can be transcribed into many thousands of mRNA copies, each of which can then be translated into many thousands of protein molecules. Thus, the amount of mRNA transcribed from each gene at any given time can give an indication of its relative metabolic activity.

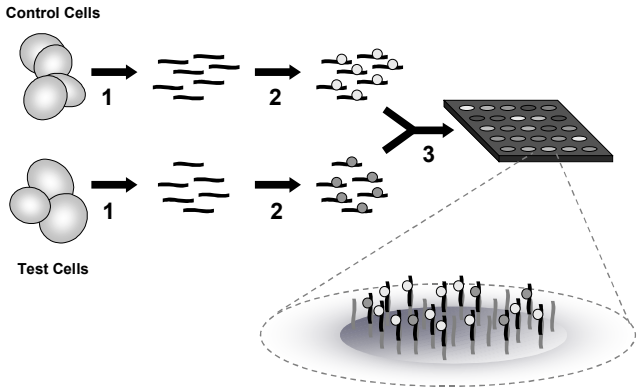


Figure 2: A typical DNA hybridisation microarray experiment.

To examine differential expression levels, cells are grown under two different conditions (Figure 2). The total mRNA complement of the cells are extracted (step 1), and single-stranded DNA copies (cDNA) are made using a reverse transcriptase enzyme (step 2). The cDNA copies are labelled with fluorescent markers, *e.g.* green for the control cells and red for the test cells. The labelled cDNA transcripts are then mixed together and applied to a microarray chip, where they bind to appropriate spots in a sequence-specific manner. Since the amount of cDNA produced for each gene is determined by the amount of mRNA in the cell extract, each spot will bind red- or green-labelled cDNA with a ratio reflecting the relative expression levels in the original cell samples. Thus, the colour of a spot (red, green, yellow, orange, *etc.*) is an indication of the expression level ratio for a particular gene between the two cell samples, and the spot's absolute fluorescence intensity reflects the total expression level relative to the other genes in the genome. For organisms whose genomes have been sequenced, the microarray can be constructed to contain spots for every known gene in their genome, such that this technique can provide data describing the expression levels of the entire genome in a single experiment. Figure 3 shows a typical microarray image, with the entire yeast genome (6116 genes, 96 intergenic regions and 349 control spots) represented on an 81 × 81 spot array. A typical microarray study would generate several dozen of these images, one for each growth condition (or time point of a time-course experiment), leading to data sets comprising several hundred thousand individual measurements.

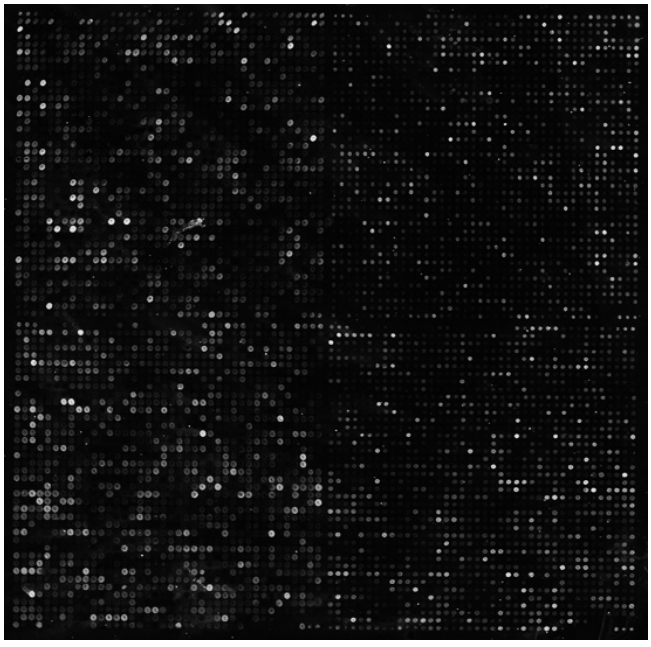


Figure 3: A microarray image of the entire yeast genome (from (DeRisi *et al*., 1997).

Although **supervised** learning methods are necessarily better for the analysis of such microarray data, and have occasionally been used (Alizadeh *et al*., 2000; Brown *et al*., 1999; Brown *et al*., 2000; Golub *et al*., 1999), these require knowledge of class membership for the examples in the training set (Kell & King, 2000). Consequently, most analyses have used **unsupervised**, clustering methods (Chu *et al*., 1998; Eisen *et al*., 1998; Spellman *et al*., 1998; Tamayo *et al*., 1999).

The data set analysed in this study (available at *http://rana.stanford.edu/clustering/*) comprised expression data for 2467 genes from the baker's yeast *Saccharomyces cerevisiae*, measured in 79 different DNA microarray hybridisation experiments (Eisen *et al*., 1998). Six classes ("Histone", "Proteasome", "TCA Pathway", "Respiratory Complex", "Ribosome" and "HTH-containing") were learnt from these data, based on known functional assignments provided in the MIPS Yeast Genome Database (*http://www.mips.biochem.mpg.de/proj/yeast/*). The first five classes are 'functional', in that in each case the members all share a common biological function. The sixth class is conventionally considered to be structural, describing the presence of a particular folding motif (a helix-turn-helix, HTH) within the protein. Being based on structure rather than function, this class has previously been considered a "control" class for the other five (Brown *et al*., 1999; Brown *et al*., 2000), but it is in fact another functional class since all the training examples are also members of the "Transcription factor" class. The 79 different experimental conditions included a time-course study of the cell division cycle after synchronisation with α-factor arrest, a cell division cycle after synchronisation by centrifugal elution, a cell division cycle measured using a temperature-sensitive *cdc*15 mutant, sporulation, heat shock, reducing shock, cold shock and diauxic shift. Sporulation is the generation of a yeast spore by meiosis, and diauxic shift is a switch from anaerobic (fermentation) to aerobic (respiration) metabolism. Heat, cold and reducing shocks are various physical ways to stress the yeast cells.

3 GENETIC PROGRAMMING

The GP implementation used in this study, developed from an earlier implementation (Gilbert *et al*., 1998; Gilbert *et al*., 1997; Johnson *et al*., 2000; Jones *et al*., 1998; Taylor *et al*., 1998; Woodward, Gilbert & Kell, 1999), was capable of deriving multiple classifier rules incorporating non-linear multivariate regression and automatic variable selection simultaneously. It used a linear model representation, was written in ANSI C, and was run on Pentium III based PC compatibles under Windows NT 4.0 and on DEC Alpha LX-164 based workstations under Linux 5.2.

The GP used the arithmetic operator functions *add*, *subtract*, *multiply*, and *protected divide* and a Boolean '*if greater than or equal to'* function. The *if* function returned a value of 1.0 if the first argument was greater than or equal to the second argument, and 0.0 otherwise. To avoid possible numeric overflows, a *protected divide* function was used which returned a numerical value of 10^{15} for divisions with a denominator $\leq 10^{-15}$. Additional protection from floating-point errors for the other operator functions was enforced by clipping the return value of each node into the range $\pm 10^{15}$.

Terminals comprised either floating-point constants (initialised randomly in the range -10.0 to 10.0 but subsequently allowed to mutate to any desired value) or input variables (corresponding to one of the 79 expression level measurements which comprised the observations for each gene). The expression level change, $E_{(i,j)}$, for gene j was defined as the logarithm of the ratio of j's expression level under condition i to its expression level in the reference state (Eisen *et al*., 1998). This log ratio is positive if gene j is induced (the expression is increased with respect to the reference state), and negative if it is repressed (the expression is decreased). $E_{(i,j)}$ is zero for genes which do not alter their expression level under the different experimental conditions. For this data set, $E_{(i,j)}$ ranged from -5.64 to 5.88, representing expression level changes over almost 6 orders of magnitude. Approximately 1.9% of the data set comprised missing data, *i.e.* points for which experimentally-determined expression level changes were unavailable. These missing points were assigned the mean values for column i (*i.e.* the mean expression level change for all the genes under condition i).

The GP initialised the population with individuals comprising random classifier rules. The performance of each individual was assessed by counting the number of correctly-classified genes for a training set comprising 152 representative genes from the six functional classes and 152 non-class member genes. The generalising ability of the GP models was assessed using a test set comprising a further 76 class members and 76 non-members. For each class, an equal number of members and non-members was included in the training and test sets. No information from the test set was used to guide the evolution of the GP. The number of genes used in the training and test sets are listed in Table 1.

Functional Class	Number in Training Set	Number in Test Set
Histone	16	6
Proteasome	48	22
TCA Pathway	22	12
Respiratory Complex	40	20
Ribosome	162	80
HTH-containing	22	10

Table 1: The number of genes used to train and test the GP.

Many of the classes had an extremely small number of training examples, and it was to the credit of the GP method that it was able to form good classifier rules using such limited information.

Since the dataset contained multiple classes, class membership was defined in the training examples by assigning several target output values, with 1.0 for members of each class, and 0.0 for non-members. Class membership was not exclusive: some genes were members of more than one class (e.g. gene *YDR178W* , coding for the succinate dehydrogenase anchor subunit, was a member of both the "TCA Pathway" and "Respiratory Complex" classes). For this reason, each individual comprised six independently-evolving classifier rules, one for each class in the training set. A correct classification (*i.e.* a "hit") was assigned when the output value of the GP-derived rule for any given class was within 0.01 of the appropriate target output. If all six rules returned *false* (*i.e.* an absolute value < 0.01), the example was deemed to have an unknown function. The fittest individuals were those that gave the greatest number of hits for the training set examples.

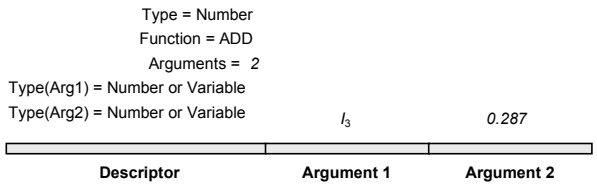
It has been observed that GP-generated rules, if allowed to evolve unchecked, tend to become longer and more arithmetically complex as the evolution proceeds, a phenomenon known as bloat. This increase in complexity reduces the interpretability of the expressions generated and is likely to lead to over-fitting of the model to the training set. To combat this, a penalty of $0.01 \times N$, where N represents the number of nodes used in the rule, was incorporated into the fitness calculation. This ensured that, for a given number of hits, a simpler rule would be chosen over a longer one. As an additional complexity constraint, the rules were limited to a maximum of 100 nodes.

The size constraints on the GP rules meant that even the longest rules could use only a small subset of the available input variables comprising the dataset. The GP was therefore compelled to perform an automatic selection of the variables it used, resulting in predictive models with significantly lower dimensionality (*i.e.* using far fewer variables) than that of the dataset as a whole. The automatic variable-selection ability of the GP approach is one of the main benefits of using this as a predictive modelling method.

The GP used five demes (sub-populations) each of 7,500 individuals. Each run was allowed to continue for a maximum of 2,000 generations, but the final model was typically produced after about 300 generations, as the GP was also allowed to terminate when 100% hits were reached for the training set. Every 10 generations, the best 5% of the individuals in each of the four satellite demes replaced the worst 5% in a central deme *seriatim*. The best 5% from this deme updated this central deme then replaced the worst 5% in each of the satellite demes, resulting in every deme containing the best 5% of the population as a whole. This divergent evolution and migration strategy has been shown to be more effective at solving high-dimensional problems than a conventional single-population (Whitlock & Barton, 1997).

During each generation, 1,500 new individuals were created by single-point mutation, and 3,000 by single-point crossover. Parental selection was proportional to fitness, and new individuals were retained in the deme if their fitness was higher than that of the current-worst individual, maintaining a population size of 7,500.

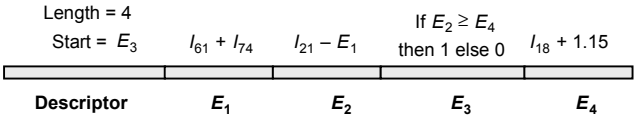
The GP used a linear representation for the classifier rules, with each rule comprising an array of function nodes and arguments. Each node comprised a descriptor block which determined the node's properties, and (if appropriate) a list of arguments. In the example shown in Figure 4, the node performs a numeric processing function: it returns a number which is the sum of its two arguments, each of which may be a floating-point number or an input variable.



$$output = I_3 + 0.287$$

Figure 4: An ADD function node from the GP function set.

Like the function nodes, every rule had a descriptor block which determined its structure (Figure 5). Although the representation was linear, the flow of operation through the rule may have been non-linear: the rule may have included multiple conditional branches, which might have led to infinite loops. These were trapped by limiting the maximum CPU time available for the execution of each rule. The input variables for the rule (I_n in the examples) obtained their values from the experimental data set.



$$\texttt{if}\,(I_{21}-(I_{61}+I_{74})) \geq (I_{18}+1.15)\,\texttt{then}\,true\,\texttt{else}\,false$$

Figure 5: The internal representation of a typical rule.

4 RESULTS AND DISCUSSION

The data set was modelled by the GP system for five replicate runs. Each run produced rules able to classify correctly genes from the six functional classes based on their expression patterns as determined by DNA microarray hybridisation. The classification accuracies for the five runs are presented in Table 2.

None of the runs produced a set of six rules with perfect classifying ability, demonstrating that the globally optimum set of rules was not found (although the rules from the best run, number 3, misclassified no training set examples, and only two from the test set). Over the five runs, at least one "perfect" rule was found for each class with the exception of the "Ribosome" class where at least one test set example was misclassified in every run. However, this was not a case of an outlier due to experimental error, as the misclassified genes were different for each of the five "Ribosome" rules, an indication of the difficulty of the task presented to the GP system.

Histone		Proteasome		TCA Pathway	
Train	**Test**	**Train**	**Test**	**Train**	**Test**
16/16	6/6	48/48	21/22	22/22	12/12
16/16	6/6	47/48	21/22	22/22	12/12
16/16	6/6	48/48	22/22	22/22	12/12
16/16	6/6	47/48	21/22	22/22	12/12
16/16	6/6	47/48	21/22	22/22	12/12

Respiratory Complex		Ribosome		HTH-containing	
Train	**Test**	**Train**	**Test**	**Train**	**Test**
40/40	20/20	162/162	79/80	22/22	9/10
40/40	20/20	161/162	78/80	22/22	10/10
40/40	20/20	162/162	79/80	22/22	9/10
40/40	20/20	160/162	79/80	22/22	10/10
40/40	20/20	161/162	79/80	22/22	10/10

Table 2: Classification accuracies for the GP models.

It was hoped that a close examination of the structures of the classifier rules would lead to new insights into biological systems at the genomic level, and this was indeed found to be true for many of the rules generated by even this relatively superficial investigation. For example, the structurally-simplest rules were derived for the "TCA Pathway" class. In four of the five runs, the final model classified this set of samples using the following rule:

`if` *alpha*[35] ≥ *alpha*[49] `then` "TCA Pathway" `else` "Unknown"

Thus, all the TCA pathway genes may be correctly classified using just the 35-minute and 49-minute points from the α-factor cell division cycle experiment (the type of observation which may guide future experimental work). The TCA genes show a decrease in expression levels over the 35 to 49-minute time period, whilst every non-TCA gene in both the training and test sets shows an increase in expression levels (for clarity we show only the averages for the class members and non-members – Figure 6).

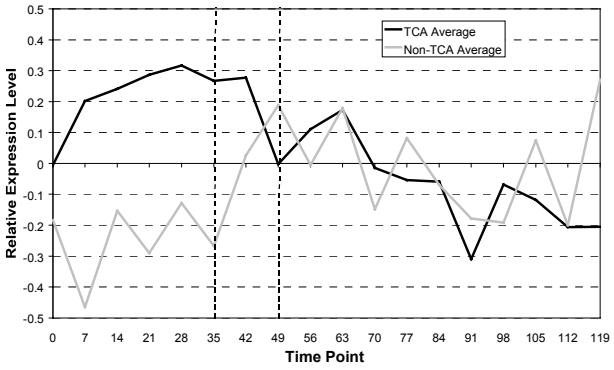


Figure 6: The averaged expression levels for TCA and non-TCA genes after α-factor synchronisation.

The GP analysis had therefore discovered the fact that a decrease in expression levels between 35 and 49 minutes after α-factor synchronisation is a behaviour apparently unique to the TCA genes which, as far as the authors are aware, has not been previously reported in the literature. This is just one example of the wealth of information which may be discovered by using GP as an explanatory modelling system for this kind of data, and similar biological insights could be provided by many of the other rules.

At present, the main objective of a functional genomics study is to assign functions to genes which are known to exist but which have never been characterised experimentally. The GP rules have the ability to perform this task as well. The rule for the "Ribosome" class derived during run 1 was:

`if` (*elution*[30]-*heat*[20]-*diauxic*[b]) ≥ (*alpha*[119]+1.15335) `then` "Ribosome" `else` "Unknown"

If this rule is applied to the data set as a whole, then 291 genes are selected from the 2,467 with available expression level data as belonging to the "Ribosome" class. According to the MIPS database classification, 121 genes are truly members of this class, and all of these are indeed found by the rule. Of the other 170 genes flagged as behaving like ribosomal proteins, most have known or suspected functions. Interestingly the majority of these (approximately 102 genes) appear to have functionally-similar properties to ribosomes (which are the organelles responsible for translating mRNA into protein). For example, genes involved in protein, amino acid and nucleoside synthesis, RNA processing, and translation / transcription control are all particularly prevalent in this list. It appears that the strict MIPS classifications may, at least in this case, be too narrow, a general problem of assigning class membership in

this type of problem (Kell & King, 2000). Although trained on genes encoding proteins which physically belong to the ribosome complex itself, the functional rule appears to be selecting genes with a functionally supportive role as also being members of this class. Of particular interest are the 12 genes of unknown function which this rule selects as being functionally-related to the "Ribosome" genes (*YCL054W*, *YDR087C*, *YDR446W*, *YGR041W*, *YKL009W*, *YKL191W*, *YLR384C*, *YMR243C*, *YNL327W*, *YOL109W*, *YPL163C* and *YPL211W*). The authors await with interest the actual functional classifications which will eventually be assigned to these genes.

5 CONCLUSIONS

Genetic programming has here been shown to be a uniquely powerful tool for analysing functional genomics data sets. Not only can it derive classifier rules with an extremely high classification accuracy, but the structures of the rules themselves have been shown to lead to the discovery of previously unsuspected biological insights into the functioning of an organism at the whole-genome level. In contrast to the best previous analysis of these microarray expression data, we were able to learn the class of HTH proteins (transcription factors), previously considered unlearnable (Brown *et al*., 1999; Brown *et al*., 2000). We believe that this is due in part to the ability of the GP to select those subsets of the available variables (Miller, 1990) which can best provide a parsimonious explanation of the problem at hand, consistent with previous general findings (Seasholtz & Kowalski, 1993; Shaw *et al*., 1997).

We consider that the benefits of GP analysis for functional genomics go well beyond simply assigning probable functional properties to previously-uncharacterised genes, a task for which it is evidently suitable. Its ability to perform automatic variable selection means that GP can be used to guide future experimental work by indicating which of the many possible experimental conditions are actually most appropriate, and which biochemical areas might best be studied by more conventional, deductive approaches (Kell & Mendes, 2000) if one is interested in a particular gene. The mechanistic structure of the models has been shown to provide new and unexpected details on the behaviour of genes at the functional-class level, leading to a better understanding of biological systems as a whole. Indeed, GP has the potential to lead the way in the analysis of functional genomics data, and by so doing will uncover new insights into the fundamental principles of biology as we move into the post-genomic era.

6 ACKNOWLEDGMENTS

We thank the UK BBSRC and EPSRC for their financial support.

7 REFERENCES

Alizadeh, A. A., Eisen, M. B., Davis, R. E., Ma, C., Lossos, I. S., Rosenwald, A., Boldrick, J. G., Sabet, H., Tran, T., Yu, X., Powell, J. I., Yang, L. M., Marti, G. E., Moore, T., Hudson, J., Lu, L. S., Lewis, D. B., Tibshirani, R., Sherlock, G., Chan, W. C., Greiner, T. C., Weisenburger, D. D., Armitage, J. O., Warnke, R., Levy, R., Wilson, W., Grever, M. R., Byrd, J. C., Botstein, D., Brown, P. O. & Staudt, L. M. (2000). Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature* **403,** 503-511.

Bork, P., Dandekar, T., Diaz-Lazcoz, Y., Eisenhaber, F., Huynen, M. & Yuan, Y. P. (1998). Predicting function: From genes to genomes and back. *Journal of Molecular Biology* **283,** 707-725.

Brent, R. (2000). Genomic biology. *Cell* **100,** 169-183.

Brown, M. P. S., Grundy, W. N., Lin, D., Cristianini, N., Sugnet, C., Furey, T. S., Manuel Ares, J. & Haussler, D. (1999). Support vector machine classification of microarray gene expression data. *University of Santa Cruz Technical Report UCSC-CRL-99-09* http://www.cse.ucsc.edu/research/compbio/genex/genex.ps.

Brown, M. P. S., Grundy, W. N., Lin, D., Cristianini, N., Sugnet, C., Furey, T. S., Manuel Ares, J. & Haussler, D. (2000). Knowledge-based analysis of microarray gene expression data by using support vector machines. *Proc. Natl. Acad. Sci.* **97,** 262-267.

Brown, P. O. & Botstein, D. (1999). Exploring the new world of the genome with DNA microarrays. *Nature Genetics* **21,** 33-37.

Chu, S., DeRisi, J., Eisen, M., Mulholland, J., Botstein, D., Brown, P. O. & Herskowitz, I. (1998). The transcriptional program of sporulation in budding yeast. *Science* **282,** 699-705.

de Saizieu, A., Certa, U., Warrington, J., Gray, C., Keck, W. & Mous, J. (1998). Bacterial transcript imaging by hybridization of total RNA to oligonucleotide arrays. *Nature Biotechnol.* **16,** 45-48.

DeRisi, J. L., Iyer, V. R. & Brown, P. O. (1997). Exploring the metabolic and genetic control of gene expression on a genomic scale. *Science* **278,** 680-686.

Dyer, M. R., Cohen, D. & Herrling, P. (1999). Functional genomics: from genes to new therapies. *Drug Discovery Today* **4,** 109-114.

Eisen, M. B., Spellman, P. T., Brown, P. O. & Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci.* **95,** 14863-14868.

Gilbert, R. J., Goodacre, R., Shann, B., Taylor, J., Rowland, J. J. & Kell, D. B. (1998). Genetic programing-based variable selection for high-dimensional data. In

Genetic Programming 1998: Proceedings of the Third Annual Conference, (ed. J. R. Koza, W. Banzhaf, K. Chellapilla, K. Deb, M. Dorigo, D. B. Fogel, M. H. Garzon, D. E. Goldberg, H. Iba and R. L. Riolo), pp. 109-115. Morgan Kaufmann, San Francisco.

Gilbert, R. J., Goodacre, R., Woodward, A. M. & Kell, D. B. (1997). Genetic programming: A novel method for the quantitative analysis of pyrolysis mass spectral data. *Analytical Chemistry* **69,** 4381-4389.

Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., Coller, H., Loh, M. L., Downing, J. R., Caligiuri, M. A., Bloomfield, C. D. & Lander, E. S. (1999). Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science* **286,** 531-537.

Hieter, P. & Boguski, N. (1997). Functional genomics: it's all how you read it. *Science* **278,** 601-602.

Johnson, H. E., Gilbert, R. J., Winson, M. K., Goodacre, R., Smith, A. R., Rowland, J. J., Hall, M. A. & Kell, D. B. (2000). Explanatory analysis of the metabolome using genetic programming of simple, interpretable rules. *Genetic Progr. Evolvable Machines* **1,** in the press.

Jones, A., Young, D., Taylor, J., Kell, D. B. & Rowland, J. J. (1998). Quantification of microbial productivity via multi-angle light scattering and supervised learning. *Biotechnol. Bioeng.* **59,** 131-143.

Kell, D. B. & King, R. D. (2000). On the optimization of classes for the assignment of unidentified reading frames in functional genomics programmes: the need for machine learning. *Trends in Biotechnol.* **18,** 93-98.

Kell, D. B. & Mendes, P. (2000). Snapshots of systems: metabolic control analysis and biotechnology in the post-genomic era. In *Technological and Medical Implications of Metabolic Control Analysis* (ed. A. Cornish-Bowden and M. L. Cárdenas), pp. 3-25 (and see http://gepasi.dbs.aber.ac.uk/dbk/mca99.htm). Kluwer Academic Publishers, Dordrecht.

Koza, J. R. (1992). *Genetic programming: on the programming of computers by means of natural selection*. MIT Press, Cambridge, Mass.

Koza, J. R. (1994). *Genetic programming II: automatic discovery of reusable programs*. MIT Press, Cambridge, Mass.

Koza, J. R., Bennett, F. H., Keane, M. A. & Andre, D. (1999). *Genetic Programming III : Darwinian Invention and Problem Solving*. Morgan Kaufmann, San Francisco.

Lipshutz, R. J., Fodor, S. P. A., Gingeras, T. R. & Lockhart, D. J. (1999). High density synthetic oligonucleotide arrays. *Nature Genetics* **21,** 20-24.

Lockhart, D. J., Dong, H. L., Byrne, M. C., Follettie, M. T., Gallo, M. V., Chee, M. S., Mittmann, M., Wang, C. W., Kobayashi, M., Horton, H. & Brown, E. L. (1996). Expression monitoring by hybridization to high-density oligonucleotide arrays. *Nature Biotechnology* **14,** 1675-1680.

Miller, A. J. (1990). *Subset selection in regression*. Chapman and Hall, London.

Oliver, S. G. (1996). From DNA sequence to biological function. *Nature* **379,** 597-600.

Oliver, S. G., Winson, M. K., Kell, D. B. & Baganz, F. (1998). Systematic functional analysis of the yeast genome. *Trends Biotechnol.* **16,** 373-378.

Seasholtz, M. B. & Kowalski, B. (1993). The parsimony principle applied to multivariate calibration. *Anal. Chim. Acta.* **277,** 165-177.

Shaw, A. D., diCamillo, A., Vlahov, G., Jones, A., Bianchi, G., Rowland, J. & Kell, D. B. (1997). Discrimination of the variety and region of origin of extra virgin olive oils using C-13 NMR and multivariate calibration with variable reduction. *Analytica Chimica Acta* **348,** 357-374.

Spellman, P. T., Sherlock, G., Zhang, M. Q., Iyer, V. R., Anders, K., Eisen, M. B., Brown, P. O., Botstein, D. & Futcher, B. (1998). Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Molecular Biology of the Cell* **9,** 3273-3297.

Tamayo, P., Slonim, D., Mesirov, J., Zhu, Q., Kitareewan, S., Dmitrovsky, E., Lander, E. S. & Golub, T. R. (1999). Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. *Proc. Natl. Acad. Sci. U.S.A.* **96,** 2907-2912.

Taylor, J., Goodacre, R., Wade, W. G., Rowland, J. J. & Kell, D. B. (1998). The deconvolution of pyrolysis mass spectra using genetic programming: application to the identification of some *Eubacterium* species. *FEMS Microbiology Letters* **160,** 237-246.

Weiss, S. H. & Kulikowski , C. A. (1991). *Computer systems that learn: classification and prediction methods from statistics, neural networks, machine learning, and expert systems*. Morgan Kaufmann Publishers, San Mateo, CA.

Whitlock, M. C. & Barton, N. H. (1997). The effective size of a subdivided population. *Genetics* **146,** 427-441.

Woodward, A. M., Gilbert, R. J. & Kell, D. B. (1999). Genetic programming as an analytical tool for non-linear dielectric spectroscopy. *Bioelectrochem. Bioenerg.* **48,** 389-396.