Anal. Chem. **1997,** *69,* 4381−4389

Genetic Programming: A Novel Method for the Quantitative Analysis of Pyrolysis Mass Spectral Data

Richard J. Gilbert,[†] Royston Goodacre,*[,‡] Andrew M. Woodward,[§] and Douglas B. Kell[||]

Institute of Biological Sciences, University of Wales, Aberystwyth, SY23 3DA, U.K.

A technique for the analysis of multivariate data by genetic programming (GP) is described, with particular reference to the quantitative analysis of orange juice adulteration data collected by pyrolysis mass spectrometry (PyMS). The dimensionality of the input space was reduced by ranking variables according to product moment correlation or mutual information with the outputs. The GP technique as described gives predictive errors equivalent to, if not better than, more widespread methods such as partial least squares and artificial neural networks but additionally can provide a means for easing the interpretation of the correlation between input and output variables. The described application demonstrates that by using the GP method for analyzing PyMS data the adulteration of orange juice with 10% sucrose solution can be quantified reliably over a 0−20% range with an RMS error in the estimate of ∼1%.

Genetic programming (GP) is an evolutionary technique which uses the concepts of Darwinian selection[1] to generate and optimize a desired computational function or mathematical expression.[2−5] An initial random population of individuals, each encoding a function or expression, is generated and their fitness to reproduce the desired output is assessed. New individuals are generated either by mutation (the introduction of one or more random changes to a single parent individual) or by crossover (randomly rearranging functional components between two or more parent individuals). The fitness of the new individuals is assessed, and the best individuals from the total population become the parents of the next generation. This process is repeated until either the desired result is achieved or the rate of improvement in the population becomes zero. It has been shown[2] that if the parent individuals are chosen according to their fitness values, the genetic method can approach the theoretical optimum efficiency for a search algorithm. Note that GP differs from genetic algorithms (GAs)[6−11] in that GAs have a fixed mapping between the genes that are exchanged and manipulated and the variables or attributes of the problem space.

There is a continuing requirement for rapid, accurate, automated methods to characterize biological systems, for instance, in determining whether a particular foodstuff has the provenance claimed for it or whether it has been adulterated with or substituted by a lower grade material. One approach to the solution of these problems has exploited pyrolysis mass spectrometry (PyMS) and various chemometric methods allowing the assessment of the adulteration or otherwise of extra virgin olive oil with lower grade seed oils,[12,13] the contamination of goats' or ewes' milk with cows' milk to below 1%,[14] and the geographical origin of olive oils to be elucidated.[15] More recently, PyMS has been used quantitatively to assess the adulteration of orange juice.[16]

PyMS is a high-resolution technique and in combination with modern supervised learning techniques, such as artifical neural networks (ANNs) and partial least squares (PLS), it has been shown that it is possible to gain accurate and precise *quantitative* information about the chemical constituents of biological samples.[17−19] However, the interpretation of the calibration models from ANNs and PLS is often difficult. To simplify the deconvolution of such complex spectra it is necessary to develop a system that itself produces "rules" that are readily comprehensible.

In the present study, we therefore exploit GP for the quantitative assessment of the adulteration of orange juice with beet sucrose and compare the results with those obtained previously using the classical approaches of ANNs and PLS. Furthermore, the use of the output from GPs allowed, at least to some degree,

[†] E-mail: rcg@aber.ac.uk.
[‡] E-mail: rrg@aber.ac.uk.
[§] E-mail: azw@aber.ac.uk.
[||] E-mail: dbk@aber.ac.uk.

(1) Darwin, C. *On the origin of species by means of natural selection*; John Murray, 1859.
(2) Koza, J. R. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*; MIT Press: Cambridge, MA, 1992.
(3) Koza, J. R. *Genetic Programming II: Automatic Discovery of Reusable Programs*; MIT Press: Cambridge, MA, 1994.
(4) Koza, J. R. *Stat. Comput.* **1994**, *4*, 87−112.
(5) Koza, J. R.; Goldberg, D. E.; Fogel, D. B.; Riolo, R. L. *Genetic Programming 1996: proceedings of the first annual conference*; MIT Press: Cambridge, MA, 1996.
(6) Goldberg, D. E. *Genetic algorithms in search, optimization and machine learning*; Addison-Wesley: Reading, MA, 1989.
(7) Holland, J. H. *Adaption in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*; MIT Press: Cambridge, MA, 1992.
(8) Mitchell, M. *An Introduction to Genetic Algorithms*; MIT Press: Boston, 1995.
(9) Lucasius, C. B.; Kateman, G. *Chemom. Intell. Lab. Syst.* **1993**, *19*, 1−33.
(10) Lucasius, C. B.; Kateman, G. *Chemom. Intell. Lab. Syst.* **1994**, *25*, 99−145.
(11) Broadhurst, D.; Goodacre, R.; Jones, A.; Rowland, J. J.; Kell, D. B. *Anal. Chim. Acta*, in press.
(12) Goodacre, R.; Kell, D. B.; Bianchi, G. *Nature* **1992**, *359*, 594−594.
(13) Goodacre, R.; Kell, D. B.; Bianchi, G. *J. Sci. Food Agric.* **1993**, *63*, 297−307.
(14) Goodacre, R. *Appl. Spectrosc.* **1997**, *51*, 1144−1153.
(15) Salter, G. J.; Lazzari, M.; Giansante, L.; Goodacre, R.; Jones, A.; Surricchio, G.; Kell, D. B.; Bianchi, G. *J. Anal. Appl. Pyrolysis* **1997**, *40/41*, 159−170.
(16) Goodacre, R.; Hammond, D.; Kell, D. B. *J. Anal. Appl. Pyrolysis* **1997**, *40/41*, 135−158.
(17) Goodacre, R.; Neal, M. J.; Kell, D. B. *Anal. Chem.* **1994**, *66*, 1070−1085.
(18) Goodacre, R.; Trew, S.; Wrigley-Jones, C.; Neal, M. J.; Maddock, J.; Ottley, T. W.; Porter, N.; Kell, D. B. *Biotechnol. Bioeng.* **1994**, *44*, 1205−1216.
(19) Goodacre, R.; Trew, S.; Wrigley-Jones, C.; Saunders, G.; Neal, M. J.; Porter, N.; Kell, D. B. *Anal. Chim. Acta* **1995**, *313*, 25−43.

S0003-2700(97)00460-5 CCC: $14.00

the rational *deconvolution* of the spectra in terms of which masses were important.

EXPERIMENTAL SECTION

Preparation of Adulterated Orange Juice. Twenty-five oranges (Outspan; navelate from South Africa) were bought from a local supermarket and were hand squeezed to give ~2 L of raw material as originally reported.[16] This was then centrifuged at 6000*g* for 20 min to remove pith and particulates (the pellet was found gravimetrically to be 6.36% w/w). Next a 10% solution of beet sucrose ("Silver Spoon", British Sugar) was prepared in distilled water and was used to adulterate the orange juice from 0 to 20% in steps of 0.5%; these 41 binary mixtures prepared therefore spanned the region 0−20 g·L^{-1} of added sucrose.

Pyrolysis Mass Spectrometry (PyMS). A 1 μL aliquot of the above materials was evenly applied on to iron−nickel foils to give a thin uniform surface coating. Prior to pyrolysis, the samples were oven-dried at 50 °C for 30 min. Each sample was analyzed in triplicate. The pyrolysis mass spectrometer used was a Horizon Instruments PYMS-200X (Horizon Instruments Ltd., Ghyll Industrial Estate, Heathfield, E. Sussex, U.K.); for full operational procedures, see refs 17 and 20. The sample tube carrying the foil was heated, prior to pyrolysis, at 100 °C for 5 s. Curie-point pyrolysis was at 530 °C for 3 s, with a temperature rise time of 0.5 s. The data from PyMS were collected over the *m/z* range 51−200 and were normalized as a percentage of total ion count to remove the most direct influence of sample size per se. Typical mass spectra of pure orange juice and the pure adulterant sucrose are shown in Figure 1.

Variable Selection. The full PyMS spectral data sets contained 150 *m/z* input variables. A large number of input variables such as this leads to a high dimensionality in the search space which must be traversed by the GP, with a consequent increase in the time taken to derive low root-mean-square (RMS) error expressions. To combat this, two methods to reduce the number of variables were employed.

(1) *Product moment correlation* (PMC) is a method that uses linear transformations to decide which variables (x) are most strongly related to the output data (y) being modeled. The inputs were ranked according to their PMC. The PMC (R) may be calculated as follows:

$$R = \frac{C_{xy}}{\sqrt{C_{xx}C_{yy}}}$$

where

$$C_{xy} = \left(\sum_{i=1}^{n} x_i y_i\right) - n(\bar{x}\bar{y})$$

$$C_{xx} = \left(\sum_{i=1}^{n} x_i^2\right) - n(\bar{x})^2$$

$$C_{yy} = \left(\sum_{i=1}^{n} y_i^2\right) - n(\bar{y})^2$$

R takes the sign of C_{xy} and thus ranges from −1 to +1, that is from a perfect negative to a perfect positive correlation.

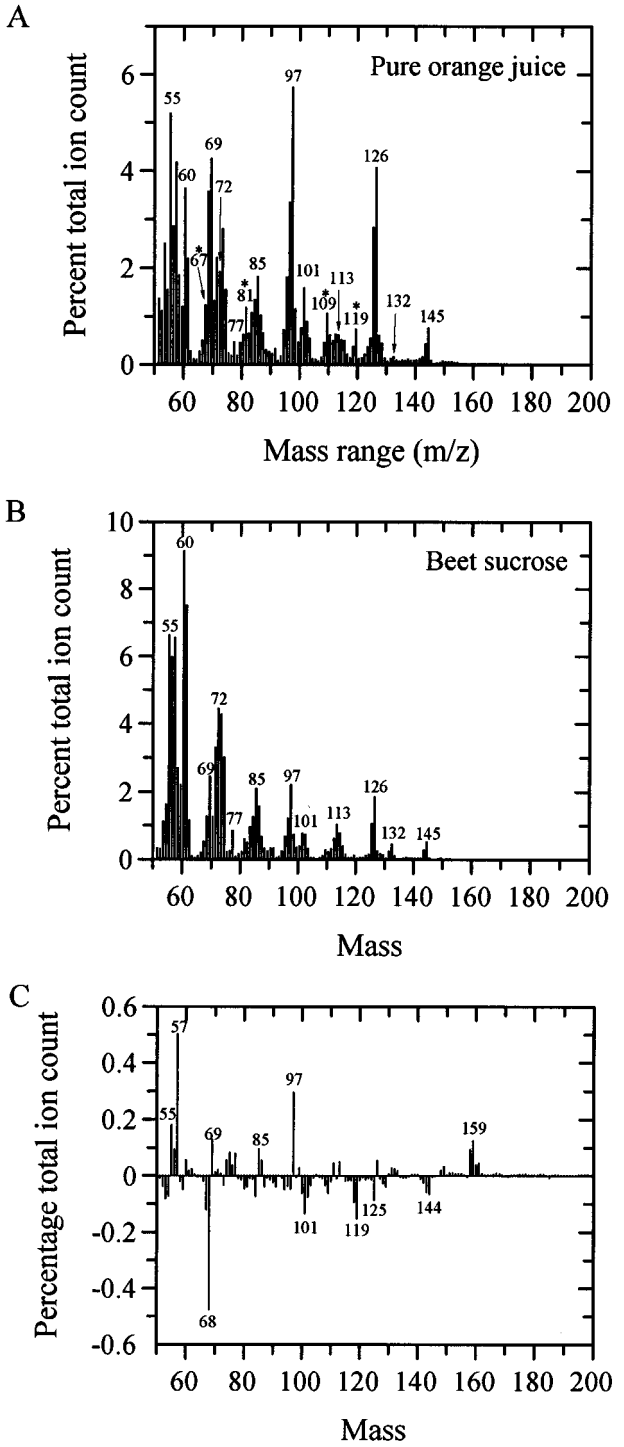

Figure 1. Representative pyrolysis mass spectra of pure orange juice (A) and beet sucrose (B). Masses marked with an asterisk are characteristic of orange juice and the others marked are from sucrose. C is the subtraction spectrum of the normalized average of three pyrolysis mass spectra of pure orange juice from the same adulterated with 2% sucrose.

(2) *Mutual information* (MI)[21,22] is a generalized version of correlation.[23] Whereas correlation assumes linear relationships and Gaussian-distributed data, MI makes no assumptions about

(20) Goodacre, R.; Kell, D. B. *Anal. Chem.* **1996**, *68*, 271−280.

(21) Shannon, C. E.; Weaver, W. *The mathematical theory of communication*; University of Illinois Press: Urbana, IL, 1949.

(22) Applebaum, D. *Probability and Information: an integrated approach*; University Press: Cambridge, MA, 1996.

(23) Battiti, R. *IEEE Trans. Neural Networks* **1994**, *5*, 537−50.

the two data series being compared. It is based on calculating the information content of one signal that is also contained in the other.

Mutual information is derived by calculating the probability distributions of the two series, $p(x)$, $p(y)$, and $p(x,y)$. It then compares the joint probability $p(x,y)$ with $p(x)p(y)$. For statistically independent data, $p(x)p(y) = p(x,y)$.[23] Hence, if these quantities are not the same, there is a dependence between the two data series and this dependence is free from all prior assumptions about its form.

Since the standard way of producing probability distributions by making histograms only works well for dense data, we used a method based on kernel density estimation[24] (and see http://euler.ntu.ac.uk/ccb/html/densest.html), which is a convolution of a smoothing function with the ordinate of the data.[22] These probability distributions are then used to form the mutual information, $I(x,y)$:[23]

$$I(x,y) = \sum_x \sum_y [P(x,y) \log_2\{P(x,y)/P((x)P(y)\}]$$

The MI is high if one data series provides a lot of information about the other and low if it provides little. Input variables can be selected in a multivariate problem by deriving $I(x,y)$ for each of them and picking those for which this value is greatest. In a purely linear Gaussian situation, $I(x,y)$ reduces to correlation and provides identical results.

MI values were calculated using Matlab version 4.2c. l (The MathWorks, Inc., 24 Prime Par Way, Natick, MA, USA), which runs under Microsoft Windows NT on an IBM-compatible PC and were used to rank the inputs accordingly.

Multivariate Regression by Genetic Programming. In order to implement a genetic optimization of mathematical expressions or computer code, it is necessary to formulate the expression in a notation that is amenable to mutation and crossover. Attempting a genetic optimization using just the "text" of a function either in standard mathematical notation or computer program code will result, in all likelihood, in the generation of nonfunctional individuals. To overcome this, the genetic program method uses the concept of a *function tree*, comprising *nodes* and *terminals*.[2]

A *terminal* is a logical unit containing an operator function (i.e., executable program code) which returns a single number: either a numeric constant or the value of an input variable.

A *node* is a logical unit comprising an operator function and one or more arguments, each of which are themselves either a node or a terminal. The return value of a node is calculated by calling its operator function, which then calls the operator functions of its arguments in order to obtain its own input values.

The principle of *closure*, by which all nodes accept and return data of the same type, is what allows the genetic program to change and rearrange function trees while retaining a logically consistent structure. A typical function tree is shown in Figure 2.

Operator functions may perform standard mathematical operations such as "*A* + *B*", "*sin(A)*", or "*(A*B)* + *C*" or program functions such as "*if A <= B then return C else return D*", "*while A > B call C*", or "*print A*", where *A*, *B*, *C*, and *D* are the node's

(24) Beardah, C. C.; Baxter, M. J. In *Analecta Praehistorica Leidensia 28, Interfacing the Past, Computer Applications and Quantitative Methods in Archaeology CAA95*; Kammermans, H., Fennema, K., Eds.; Institute of Prehistory, University of Leiden Press: Leiden, 1996; pp 179−184.

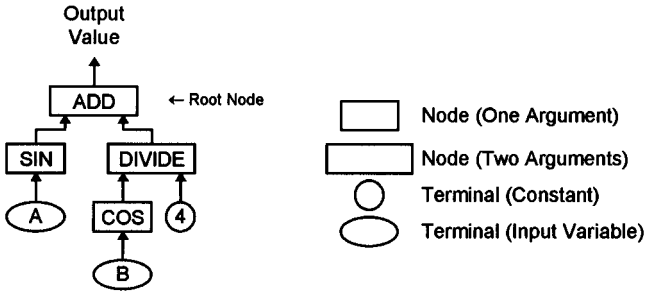

Figure 2. A typical function tree. The tree shown here represents the expression (*sin*(*A*) + (*cos*(*B*)/*4*)), where *A* and *B* are input variables. To evaluate the expression, the operator function of the root node is called. When an operator function is called, it calls the operator functions of its argument nodes to obtain its own input values. It then calculates its output value and returns it to the calling function. Terminals are nodes that have no arguments: they return either a numeric constant or the value of an input variable.

arguments. The advantage of encoding the function in this way is that mutations can be made simply by changing a node's arguments or operator function, and crossovers can be performed by replacing one or more nodes from one individual with those from another.

For the GP implementation described here, which was capable of performing nonlinear multivariate regressions, the node operator functions were add, subtract, multiply, divide, inverse ($1/x$), negate ($-x$), square, square root, absolute value ($|x|$), exponent (e^x), natural logarithm, sine, cosine, and tangent. The first four functions on this list take two arguments, the others just one (although the GP program as implemented allowed functions to take any number of arguments). The return values of each node were clipped into the range $\pm 10^{15}$ to avoid possible numeric overflows, and all trigonometric functions took their argument to be in radians. It was possible to disable operator functions selectively in order to simplify the rationalization of the GP-generated expressions and to reduce the dimensionality of the search space of the optimisation task. However, this incurs the penalty of the GP needing to approximate any "disabled" nonlinear relationships between the inputs and outputs by using the linear operators, resulting in the generation of longer and less accurate expressions.

The GP generated initial individuals with random function trees and assessed their fitness using a scoring function that compared each individual's estimate of the output with observed values. It then carried out a number of reproductive generations using these individuals in order to derive an expression describing the relationship between one or more input variables and the desired output.

A generation comprised sorting the individuals in a population in descending order of fitness, mutating the best 30% of the individuals (with the new individuals replacing the worst in the population if found to be fitter), re-sorting in descending fitness order a second time, and then crossing-over 30% of the population (with the parents chosen from the best 30% and the offspring replacing the worst individuals as before). This process was reiterated until (a) an expression was found with a fitness that matched the desired value to within a specified tolerance (an RMS error of less than 0.1; not actually reached with the data sets described here), (b) the rate of improvement fell to a low value (arbitrarily, a zero fitness score change over 50 generations), or (c) a specified maximum number of generations was reached

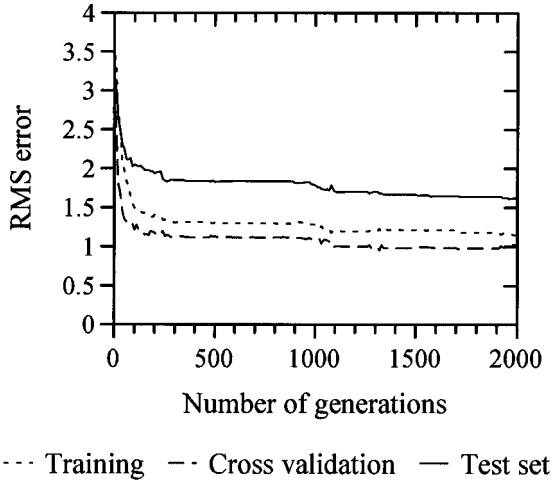

Figure 3. A learning curve for the GP. A linear GP was run for 2000 generations, using the PC data sets. The final RMS errors in the estimates for this run were 1.15 for the training set, 1.03 for the cross-validation set, and 1.25 for the test set. The curve illustrates the fact that a GP is able to escape from local minima: until ~1000 generations, the RMS errors had settled to 1.30 for the training set, 1.12 for the cross-validation set, and 1.38 for the test set. Eventually, the GP was able to find a lower minimum.

(arbitrarily set to 5000 generations). This process of "learning" can be plotted as a learning curve where the RMS errors in the training, cross-validation, and test sets are plotted against the number of generations. A typical learning curve for the experimental data used in this study is shown in Figure 3.

Six populations of 250 individuals each were run in parallel. After every 10 generations, the population containing the individual with the best fitness was duplicated into the other 5 populations, replacing their individuals. This increased the efficiency of traversing the search space by providing the means for a population to escape from a relatively poor local minimum.

An individual's fitness was assessed as the RMS of the difference between expected values and the GP's estimated values for a training set of data, plus the absolute value of the difference between the training set RMS error and the RMS error for a cross-validation set, plus a penalty of 0.001 multiplied by the number of nodes in the individual's function tree. Fitter individuals had lower fitness scores. The node count penalty was included to select for shorter, simpler expressions: once a low score was achieved, expressions that gave roughly the same RMS errors but had fewer nodes replaced more complex expressions.

The described results for the quantification of orange juice adulteration used a training set of 33 samples (the inputs were (a) the full PyMS spectra of 150 mass ion counts, (b) a reduced set of only 20 mass ion counts (vide infra), or (c) 20 principal components scores derived from the full PyMS spectra, for mixtures of adulterations of 0%, 2%, ..., 20% using a 10% sucrose solution; each solution was relicated three times) and a cross-validation set of 30 samples (three replicates of adulterations of 1%, 3%, ..., 19%). The current best individual of each population (as judged by its ability to give accurate estimates for the percent sucrose in orange juice; and vide infra) was then used to estimate the percent adulteration with sucrose for a test set of 60 samples (three replicates of adulterations of 0.5%, 1.5%, ..., 19.5%). The use of a cross-validation set (whose expected outputs were chosen to intercalate with those of the training set) ensured that the generated expression was able to generalize when making its estimates and reduced the likelihood of overtraining to the training set.

An alternative training method (results not shown) was to use a fitness function which, at each assessment, randomly picked 30 samples from the combined training and cross-validation sets in order to calculate the the errors in the GP's estimates. Since the "training set" was different for each fitness assessment, the problem of overtraining to any one data set was avoided, while the selective pressure for generalization was retained. This training method gave results similar to the previous method but took longer to run due to the increased overhead incurred during the fitness function evaluation step.

Terminals were initially assigned either a numeric value in the range 0−1, 0−10, or 0−100 with a probability of 0.2 each, or the value of one of the input variables with a probability of 0.4. Each input had an equal chance of being chosen.

Nodes were created with initial function trees ranging randomly from a single node (comprising a terminal) to a tree of depth 8. Operator functions were chosen randomly with equal probability.

The best 30% of each population were used to generate new individuals by mutation. When an individual was chosen for mutation, a node in its function tree was selected randomly. If the selected node was a terminal with a numeric value, then (a) its value was multiplied by a random scaling factor between 0.8 and 1.2 using a sinusoidal bell-shaped function (to select for values close to 1.0) with a probability of 0.55, (b) its value was changed to a new random constant with a probability of 0.15, (c) the node was changed to an input-variable terminal with a probability of 0.15, or (d) the node was changed to a new random tree of depth between 1 and 8 with a probability of 0.15. If the node was an input variable terminal it was (a) given the value of a different, randomly chosen, input variable with a probability of 0.7, (b) changed to a numeric constant terminal with a probability of 0.15, or (c) changed to a new tree with a probability of 0.15. If the node was not a terminal, then (a) its operator function was changed to a different one with the same number of arguments with a probability of 0.7, (b) it was changed to a new random tree with a probability of 0.15, or (c) it was changed to a terminal with a probability of 0.15. If a node was changed to a terminal, it took the same numeric value it returned as a function node. After each mutation, there was a 0.3 probability of another mutation occurring to the same individual. The fitness of the new individual was then assessed, and the worst individual in the population was replaced if the new individual was found to be fitter.

After the mutations were performed, the populations were sorted in descending order of fitness. Crossovers were then performed by randomly selecting two individuals from the best 30% of the population. Two nodes were randomly chosen, one from each "parent" function tree, and were exchanged to generate two new individuals. The new individuals replaced the worst two in the population if they had a better fitness score. An index to the "end" of the population was used to prevent the two new individuals from being replaced by a subsequent crossover during the current generation. With a probability of 0.15, the crossover was performed not with two existing individuals but with a single individual from the population and a new, random, tree of depth 1−8 nodes. The number of crossovers to perform in each generation was calculated by multiplying the population size by 0.3.

The genetic programming method is widely assumed to demand a high level of computational power. Using the parameters described, this implementation of the genetic programming technique requires only ~6 MB of RAM in which to operate and is, therefore, suitable to be run on many desktop computers in current use. The implementation described here was written in ANSI C on a Power Macintosh 7200/90 following a procedure similar to Singleton.[25] With the orange juice adulteration data sets, it typically takes ~6 h to generate an expression with a low (~1.1) RMS error in its estimations. This time is drastically reduced, to less than 45 min, when 20 selected input variables are used rather than the full 150-input data set. Expressions with RMS errors equivalent to those of ANNs and PLS (~1.7) are frequently derived in less than a few minutes using 150 or 20 data inputs. Use of the nonlinear operator set typically increases the run time ~5-fold. The probable explanation for this is that each additional operator function effectively adds an extra "degree of freedom" to the search space traversed by the GP.

Once derived, the GP-generated expressions can be incorporated into a simple program or used in a spreadsheet to estimate output values for additional input data sets without further reference to the full GP program.

RESULTS AND DISCUSSION

After collection of the data, the first stage was to observe any obvious features present in the pyrolysis mass spectra (Figure 1). From the spectrum of beet sucrose (Figure 1b), one can observe the following series of peaks as being characteristic: *m/z* 55, 60, 69, 72, 77, 85, 97, 113, 126, 132, and 145. These peaks are also seen clearly in the spectrum of pure orange juice; indeed, sucrose occurs naturally in orange juice, and of the 10 g of carbohydrates/100 g of orange juice, typically 2.9−5.6 g of this is from sucrose;[26] orange juice also contains some characteristic peaks at *m/z* 67, 81, 109, and 119 (marked with an asterisk in Figure 1A) which were absent, or at least very small, in the spectrum of beet sucrose. Figure 1C is the subtraction spectrum of the normalized average of three pyrolysis mass spectra of pure orange juice from the same adulterated with 2% sucrose (20% of the 10% solution), this spectrum highlights *m/z* 55, 57, 68, 69, 85, 97, 101, 119, 125, 144, and 159 as being important.

To observe simple relationships between our mass spectra and the level of adulteration, product moment correlation and mutual information (see above for calculations) were used to rank the masses in order of importance to the single *Y* variable (i.e., the determinand, sucrose) for *only* the training set. The first 20 masses were chosen to be (a) in order of PMC, best first *m/z* 68, 118, 119, 91, 67, 142, 101, 116, 117, 100, 143, 62, 120, 144, 90, 80, 56, 131, 57, and 109; and (b) MI, best first, *m/z* 119, 118, 68, 91, 142, 67, 101, 117, 100, 116, 143, 62, 110, 144, 61, 120, 56, 140, 80, and 131. It was interesting that of the first 20 masses, 17 were deemed to be significant by both methods, and for ease of interpretation this is detailed in Figure 4. The discrepancies were for PMC *m/z* 90, 57, and 109 (which were ranked 15, 19, and 20, respectively) and for MI *m/z* 110, 61, and 140 (ranking 13, 15, and 18, respectively). When the intensities of each of these "significant" masses were plotted against the sucrose concentrations, with the exception of *m/z* 62, 56, 131, 57, 61, and 140, which were positively correlated, all the other masses were found to be

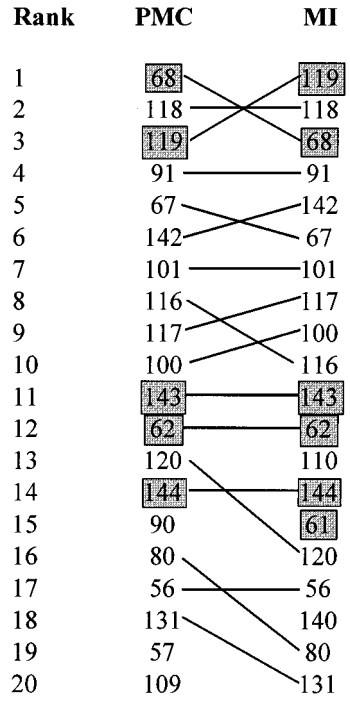

Figure 4. Ranking of the first 20 most significant masses as judged by reducing the dimensionality of the mass spectral input space by ranking variables according to the PMC or MI with the outputs. The same masses that are chosen by both methods are joined by a line and those masses that are shaded were frequently found in expressions from linear and nonlinear GPs trained with the full mass spectra.

negatively correlated with sucrose concentration, while *m/z* 90, although negatively correlated, was very noisy. That the majority of these masses were negatively correlated was hardly suprising since the samples analyzed were between 0% and 20% sucrose solution in 100%−80% orange juice, and so the major spectral components would have been due to the orange juice per se. Indeed, one can observe that of the peaks chosen by visual selection as being important components of orange juice (Figure 1a and c), *m/z* 119 (ranked third by PMC and first by MI), 67 (ranked fifth by PMC and sixth by MI), and 101 (ranked seventh by PMC and MI), were also chosen by the variable selection methods, while *m/z* 109 was found to be significantly correlated only by PMC (although ranked 20th) and *m/z* 81 was not selected by either method. Interestingly, the only visually evident sucrose-specific peak chosen by the variable selection procedures was *m/z* 57 (ranked only 19th by PMC and not at all by MI).

In order to remove the most direct influence of sample size per se the PyMS data had to be normalized, and in this instance the intensities over the *m/z* range 51−200 were normalized as a percentage of total ion count. Although this method of normalizing does have the disadvantage that if a given mass is relatively greater in a particular spectrum then all of the other masses are, although to a much lesser extent, necessarily relatively lower, we did not find this to be a problem. Indeed masses choosen by PMC, MI, and the GPs (vide infra) were all seen to be significant from viewing the spectra of orange juice and beet sucrose and the subtraction sepctrum (Figure 1).

Genetic Programming Results on the Full Mass Spectra. The next stage was to investigate the ability of a "linear" GP (that is to say a GP that used only "+", "−", "/", and "*") to assess quantitatively the level of sucrose adulteration in mixtures with pure orange juice. As detailed above, the training set comprised

(25) Singleton, A. *Byte* **1994**, *19*, 171.
(26) Robards, K.; Antolovich, M. *Analyst* **1995**, *120*, 1−28.

Table 1. Comparison between Different Analytical Methods for Estimating the Adulteration of Orange Juice

method	av test set RMS errors	approx time to compute[e]
PLS, using all 150 m/z[a]	1.26	20 s
PLS, using 20 most characteristic masses calculated by PMC[b,c]	1.25	10 s
PLS, using 20 most characteristic masses calculated by MI[b,c]	1.67	10 s
150-8-1 MLP[a]	1.64	1 min
20-4-1 MLP, using 20 mostcharacteristic masses calculated by PMC[b,d]	1.20	20 s
20-4-1 MLP, using 20 most characteristic masses calculated by MI[b,d]	1.82	20 s
8-4-1 MLP, using first 8 PCs[a]	1.61	15 s
150-50-1 RBF[a]	1.74	20 min

[a] These results are taken from ref 16. The same PyMS experimental data sets were used for all the analytical methods shown. [b] Characteristic masses were calculated using the PMC or MI as detailed in the text. [c] PLS was calibrated using 10 factors as judged by RMS error of cross-validation data set. [d] MLPs were trained to optimal point as judged by RMS error of cross-validation data set. [e] All these times were from use of an IBM-compatible PC (dual P133 processor, with 64 Mbytes RAM).

the 11 normalized triplicate pyrolysis mass spectra from mixtures containing 0, 2, 4, ..., 18, and 20% sucrose. The fitness function used during calibration was the RMS of the training set plus the absolute value of the difference between the cross-validation (1, 3, 5, ..., 17, and 19%) and training sets (this was used to enforce at least some generalization to the cross-validation set), in addition a node-count penalty was exploited to select for shorter expressions.

Due to the high dimensionality (150 inputs) of the search space traversed by the GP, calibration was necessarily slow and each run took ~6 h to converge on a Power Macintosh 7200/90. Each run evaluated, typically, 3.5×10^3 generations, each comprising 1.5×10^3 individuals, a total of $\sim 5.25 \times 10^6$ individuals per run. The GP runs were invariably able to converge successfully. The shortest expression (of 10 runs; Table 2) was

$$\begin{aligned}\%\ \text{adulteration} = {} & ((((6.823529/P_{100})/P_{51}) + (P_{143} + \\ & ((P_{143}P_{119}) + (((P_{62} - P_{51})/((P_{62} \times 2.901961) - P_{119})) + \\ & ((P_{62}P_{73})((P_{62}P_{73})((P_{62}P_{61})(((P_{143} - 1.215686)/((P_{62}P_{73}) - \\ & P_{119})) + ((((P_{62}P_{73}) - P_{100})P_{73}) + (P_{51} + ((((P_{143} \times \\ & 2.901961) + (P_{143} \times 4.117647)) - P_{100}) + ((P_{62} \times \\ & 6.431373) + ((P_{62} \times 6.823529) + (P_{143} + ((P_{143}P_{61}) + \\ & (P_{143} + (P_{51} + ((P_{51} + P_{100})(((P_{51} - P_{119}) - P_{119}) \\ & (4.352941 + (7.215686/P_{119}))))))))))))))))))) - P_{100})\end{aligned}$$

where P_n is the normalized ion count for mass n.

This expression gave an RMS error in the estimations of 1.64 for the training set, 1.72 for the cross-validation set, and 1.87 for the test set. For the test set error, the results for all 10 GPs (average 1.93 with a standard deviation of 0.11) were slightly higher than those found previously using the more classical approaches of PLS and ANNs (Table 1).

The above GP expression, and others found in this set of experiments (data not shown), often used the same mass ions many times, the most common of which chosen were m/z 51, 62, 119, and 143. On inspection of the pyrolysis mass spectra of pure orange juice and the adulterant beet sucrose (Figure 1), m/z 51 and 119 are much more abundant in pure orange juice, while m/z 62 is more characteristic of sucrose; by contrast m/z 143 is very small in both. Therefore it would appear that these GPs are at least using variables that are important and quantitative for the description of sucrose adulteration in orange juice.

Ten more GPs were calibrated using the same population parameters as detailed above with the same 150 m/z intensities, but these differed in that as well as the linear functions used above extra "nonlinear" functions were used [inverse ($1/x$), square, square root, absolute value ($|x|$), exponent (e^x), natural logarithm, sine, cosine, and tangent]. Again, despite the high dimensionaility of the inputs, these GPs were able to generalize and the shortest expression was found to be

$$\begin{aligned}\%\ \text{adulteration} = {} & ((\tan(\cos(P_{68}/0.225490))) + \\ & ((\cos(P_{61}/0.225490)) + ((\tan(\cos(P_{68}/0.225490))) + \\ & ((\cos(9.686275(P_{51} - P_{119}))) + ((\tan(\cos((9.607843(P_{51} - \\ & P_{68})) + 0.220392))) + ((\tan(\cos(\tan(\cos((P_{68}/0.225490) - \\ & (P_{51} - P_{119})))))) + ((\tan P_{61}) + \\ & ((\tan(\cos((P_{68}/0.224314) - (P_{51} - P_{119})))) + \\ & ((\tan(\cos((P_{68}/0.224314) - (P_{51} - P_{119})))) + \\ & ((\tan(\cos((9.686275(P_{51} - P_{119})) - 0.220392))) + \\ & ((\exp(\cos(P_{61}/0.225490))) + \\ & ((\cos((P_{68}/0.225490)/P_{144})) + (((P_{61}/0.225490) - P_{119}) - \\ & 0.353725)))))))))))))\end{aligned}$$

where P_n is the normalized ion count for mass n.

This expression gave an RMS error in the estimations of 1.29 for the training set, 1.19 for the cross-validation set, and 1.47 for the test set. The test set results (average 1.51 with a standard deviation of 0.11) for these 10 GPs were very similar to those found previously using the classical approaches of PLS and ANNs (Table 1). This implies that the ability to map nonlinear functions (while also mapping the linear ones) was important; indeed, functions such as sine, cosine, tangent, and exponent were found in many of these expressions. It is likely that nonlinearity was because of intermolecular reactions taking place in the pyrolysate, leading to a lack of superposition of the spectral components and to a dependence of the normalized mass spectrum on sample size, a phenomenon seen previously.[18] Unfortunately, due to the nonlinear nature of the expressions, as illustrated by the GP above, the interpretation of the GP expressions will be extremely difficult and deconvolution is not realistic, although one can clearly see that m/z 51 and 119 feature regularly (as seen with the linear GP) and that m/z 61 and 68 are also important masses for the quantification of sucrose in orange juice.

The best single GP was a nonlinear one and the RMSEP was only 1.28 (Table 2); this was better than multilayer perceptions (MLPs) trained with the standard back-propagation algorithm,[27−29] which had an RMSEP of 1.64, and radial basis functions (RBFs),[30,31]

(27) Rumelhart, D. E.; McClelland, J. L.; Group, T. P. R. *Parallel Distributed Processing, Experiments in the Microstructure of Cognition*; M.I.T. Press: Cambridge, MA, 1986.
(28) Chauvin, Y.; Rumelhart, D. E. *Backpropagation: theory, architectures and applications*; Lawrence Erlbaum Associates: Hillsdale, NJ, 1995.
(29) Werbos, P. J. *The roots of back-propagation: from ordered derivatives to neural networks and political forecasting.*; John Wiley: Chichester, UK, 1994.

Table 2. Test Set RMS Errors[a]

	full spectra		PMC reduction method		MI reduction method		PC data	
run no.	linear	nonlinear	linear	nonlinear	linear	nonlinear	linear	nonlinear
1	2.16	1.55	1.76	2.29	1.59	1.67	1.42	1.02
2	2.08	1.60	2.40	2.23	1.56	1.69	1.25	1.04
3	1.87	1.55	1.76	1.87	1.69	1.58	1.18	1.09
4	1.82	*1.28*	1.93	*1.65*	1.72	*1.57*	1.07	1.24
5	*1.80*	1.48	1.64	2.43	1.73	1.66	1.59	*0.92*
6	1.88	1.63	1.89	1.74	1.54	1.57	*1.05*	1.15
7	1.90	1.38	1.75	1.83	1.71	1.70	1.14	1.33
8	2.00	1.47	*1.62*	2.05	*1.52*	1.67	1.67	1.37
9	1.89	1.52	2.33	1.87	1.66	1.60	1.51	0.99
10	1.91	1.67	2.30	1.77	1.56	1.64	1.13	1.06
average	1.93	1.51	1.94	1.97	1.63	1.63	1.30	1.12
std dev	0.11	0.12	0.30	0.26	0.08	0.05	0.23	0.15
time[b]	6 h	6 h	45 min	45 min	45 min	45 min	45 min	45 min

[a] The GP runs were repeated 10 times to assess the variability in the accuracy of the estimates calculated by using the generated expressions. Four sets of input data were used: (1) the normalized ion counts (m/z from 51 to 200) from the PyMS spectra, a reduced set of 20 characteristic masses calculated using (2) the PMC, (3) MI (see text for details), or (4) the first 20 PCs derived from them. Two types of GP were run: a "nonlinear" GP, using all of the operator functions described in the text, and a "linear" GP, using only add, subtract, multiply, and divide. Entries in italic type are the best result from each set of 10 runs. [b] All these times were from using a Power Macintosh 7200/90 (601 processor, with 24 Mbytes RAM).

where the RMSEP was 1.74, and as good as that found using the linear regression technique of PLS[32] where the RMSEP was 1.26 (Table 1).

Genetic Programming Results on Reduced Mass Spectra. The training set for the above GPs contained only 33 spectra (11 samples in triplicate) described by 150 m/z intensities, and it is well-known that if the number of parameters in calibration models such as PLS and ANNs are significantly higher than the number of exemplars in the training set, then these methods have a tendency to overfit.[33,34] It is likely that the same trends will be true for GPs, athough the fact that these GPs were still able to converge shows that this was not necessarily a problem for this data set. However, convergence was slow, taking ~6 h, and to obey the parsimony principle as described by Seasholtz and Kowalski[33] the next stage was to reduce the number of inputs to the GPs. Therefore, product moment correlation and mutual information were used to select the 20 most well correlated (or significant) masses (from the training set only) to the percent adulteration; these were used as inputs to the linear and nonlinear GPs.

For the PMC- and MI-reduced data sets, 10 linear and 10 nonlinear GPs were calibrated for 2×10^2 generations, each with 5 populations each of 2×10^2 individuals, and these typically took between 1 and 1.5 h. The RMS errors for the test sets are shown in Table 2. It can be seen that for PMC-reduced data the linear and nonlinear GPs both give results very similar to one another and the average RMSEP was 1.94 and 1.97, respectively. It was interesting to observe that the linear GP gave a result very similar to the linear GP trained on the full spectra, whereas the nonlinear GPs estimates on the reduced data were worse than those when the full spectra were used; 1.97 compared to 1.51. This implies, for modeling the percentage adulteration of orange juice with sucrose, that the spectra were not purely linearly separable and that using the linear variable selection method PMC meant that valuable information was lost. Although in previous studies (see Table 1) PLS was better than nonlinear calibrations based on MLPs and RBFs, we would expect that if a nonlinear method, such as MI, was used to select the variables, then the nonlinear GP would improve. This was indeed found to be the case, and the average RMSEPs for the MI and PMC were 1.63 and 1.97, respectively, compared to a RMSEP of 1.51 for the full spectra as input to the GP (Table 2). However, the RMSEP for the linear GP calibrated on MI-selected variables also decreases. To investigate this phenomenon further, PLS was carried out on the PMC- and MI-reduced data and the RMSEPs were found to be 1.25 and 1.67, respectively, compared to the full spectral PLS approach where the RMSEP was 1.26 (Table 1). This indeed shows that MI has selected at least some nonlinear variables; however, and by contrast, MLP models showed that PMC was better than MI and the RMSEPs were 1.20 and 1.82, respectively, compared to an RMSEP of 1.64 from an MLP trained on the full spectra.

In many instances, the linear and nonlinear GPs trained with the 20 most characteristic using PMC or MI gave test set errors very similar to GPs trained on the full mass spectra (Table 2), so variable selection has not caused the modeling of the adulteration of orange juice to get any worse.

The above GPs all produced different expressions, and it was consequently impossible to write a single expression that could be used as a general equation for calculating the level of sucrose adulteration in orange juice. This shows that, by using the simplistic genetic search algorithm described, the GP runs were only able to find expressions corresponding to *local* optima in the search space, rather than the single *global* optimum expression. This is a common problem for heuristic or nondeterministic approaches of this kind.

However, by visually examining the expressions generated during the GP runs, it is possible to select m/z intensities that occur frequently in the derived expressions and to determine, albeit in a simplistic and qualitative way, how those intensities are proportionally related to the level of adulteration with sucrose. The term for m/z 119, P_{119}, occurs in almost all of the GP-derived

(30) Moody, J.; Darken, C. *Neural Comput.* **1989**, *1*, 281−294.
(31) Broomhead, D. S.; Lowe, D. *Complex Syst.* **1988**, *2*, 321−355.
(32) Martens, H.; Næs, T. *Multivariate calibration*; John Wiley: Chichester, UK, 1989.
(33) Seasholtz, M. B.; Kowalski, B. *Anal. Chim. Acta* **1993**, *277*, 165−177.
(34) Bishop, C. M. *Neural networks for pattern recognition*; Clarendon Press: Oxford, UK, 1995.

expressions and is almost always negated. This suggests that one of the most significant "rules" derivable from the GP results is that P_{119} is proportional to the level of adulteration, A, in a negative and linear way:

$$A \propto -P_{119}$$

This is supported by the fact that both the MI and PMC variable selection methods rank P_{119} very highly (third and first, respectively), with a strong negative correlation, and that this m/z is prominent in the subtraction spectrum of the pyrolysis mass spectra of pure orange juice from the same adulterated with 2% sucrose (Figure 1C). A similar negative linear relationship can be observed for P_{67} (ranked fifth and sixth) and P_{116} (ranked eighth and tenth). Note that P_{119} and P_{67} were both selected by visual inspection as being characteristic peaks for orange juice.

A term approximately equivalent to $\cos(4P_{68})$ also occurs with a high frequency in the expressions derived by GPs using the nonlinear function set. This suggests the rule

$$A \propto \cos(4P_{68})$$

The approximate factor of 4 in this rule is, presumably, correcting for the fact that the argument to the cosine function, which is assumed to be in radians, needs to be scaled appropriately to give the correct nonlinear relationship. Note that P_{68} is ranked first and third by MI and PMC, respectively, and is seen to be prominent in the subtraction spectrum (Figure 1C), supporting the significance of this peak.

Genetic Programming Results on Principal Component Scores. PCA is an excellent dimensionality reduction technique, and the use of PC scores as inputs to neural networks, without deterioration of the calibration model, has previously been applied to the analysis of UV/visible spectroscopic data,[35,36] for the identification of bacteria from their FT-IR spectra,[37] and for the quantification of binary mixtures of *Escherichia coli* from their PyMS spectra[38] and the adulteration of orange juice.[16] Therefore the first 20 PCs were extracted, which accounted for 99.58% of the total variance in the spectral data, and these were used as the inputs to other GPs (PC-GPs).

The best expression found using the first 20 principal components derived from the PyMS data was by a nonlinear PC-GP and was

$$\begin{aligned}\%\ \text{adulteration} = (((((((10.783312 + ((PC_1 - ((((PC_9 + \\ (PC_{13} - PC_{11})) + PC_{13}) + PC_{13}) + (PC_{13} - PC_5))) \times \\ 3.981746)) + (18.966914(-PC_3))) + (12.727644(- \\ PC_2))) + (22.313057(-PC_4))) + (4.652989 \tan(PC_2))) - \\ ((7.259247 + (19.076536PC_4))PC_4)) \exp(PC_{11}))\end{aligned}$$

where PC_n was the nth principal component score.

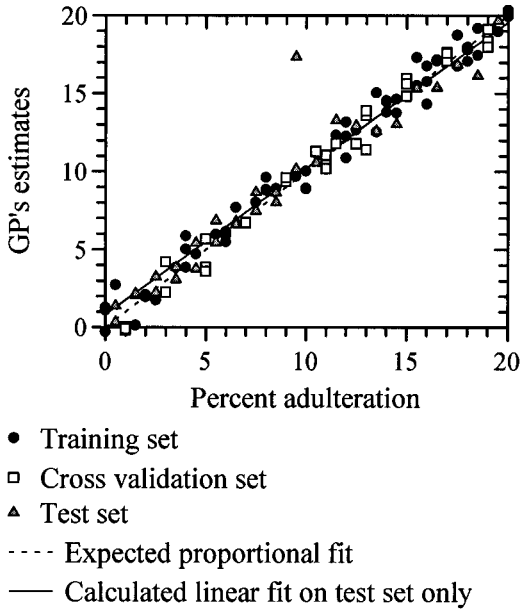

Figure 5. Estimated vs actual adulteration of orange juice with a 10% sucrose solution. The results shown are for the best expression generated using the PC data sets and a nonlinear GP. The final RMS errors in the GPs' estimates were 0.81 for the training set, 0.81 for the cross-validation set, and 0.92 for the test set. The estimate for the 9.5% adulteration sample in replicate 1 of the test set was a consistent outlier, so the estimates for this sample were discarded for the RMS error calculations.

This expression gave an RMS error in the estimations of 0.81 for the training set, 0.81 for the cross-validation set and 0.92 for the test set. The estimates of adulteration calculated by using this expression are shown in Figure 5, and the test set errors for this and the 10 linear and 10 nonlinear PC-GPs are shown in Table 2.

The expressions derived using the first 20 PCs gave substantially better RMS errors in their estimation of the degree of adulteration than the previous GPs trained on the full mass spectra or on a subset of this. Moreover, the RMSEPs were much lower than those seen previously when the adulteration was estimated by PLS, MLPs, or RBFs (Table 1).

That the nonlinear GPs reproducibly gave better estimates than linear GPs (Table 2) implies that the ability to perform a nonlinear map is still very important, and although PCA is a linear dimensionality reduction technique, it is worth stating that combinations of linear PCs will allow the emergence of the nonlinear properties. Unfortunately, the additional level of abstraction introduced by using PCs makes the rationalization of these expressions unfeasibly difficult, particularly when the GP expression includes nonlinear functions.

CONCLUSIONS

PyMS is a physicochemical spectroscopic method which produces a complex fingerprint of the sample under investigation. The spectral data recorded are in the form of the relative intensities of 150 masses. It is obvious that these data are of high dimensionality and necessarily difficult to interpret visually. We have previously shown that the neural cognition-based methods such as MLPs and RBFs and the linear regression technique of PLS could be employed successfully for the quantitative assessment of the adulteration of orange juice with sucrose.[16] However, the deconvolution of which masses were important, in a readily interpretable fashion, was not possible.

(35) Gemperline, P. J.; Long, J. R.; Gregoriou, V. G. *Anal. Chem.* **1991**, *63*, 2313−2323.
(36) Blanco, M.; Coello, J.; Iturriaga, H.; Maspoch, S.; Redon, M. *Anal. Chem.* **1995**, *67*, 4477−4483.
(37) Goodacre, R.; Timmins, É. M.; Rooney, P. J.; Rowland, J. J.; Kell, D. B. *FEMS Microbiol. Lett.* **1996**, *140*, 233−239.
(38) Timmins, É. M.; Goodacre, R. *J. Appl. Microbiol.* **1997**, *83*, 208−218.

Evolutionary programming is a new and exciting method for extracting meaningful correlations between inputs and outputs. We have previously shown that genetic algorithms can be used as a method for the selection of relevant variables from mass spectra prior to analysis by PLS.[11] An obvious extension of this approach is to use genetic programming to evolve a mathematical solution to the correlation of the relevant features of an *X*-matrix (e.g., mass spectra) to a *Y*-variable representing information of biological interest (e.g., percent adulteration).

In conclusion, this is the first study that has shown that even a relatively simplistic genetic programming approach can be applied successfully to the accurate quantification of orange juice adulteration by analyzing PyMS spectral data. This approach provided estimates for the adulteration that were at least as good as those from more widespread analytical methods such as ANNs and PLS, but with the additional benefit of enabling the rationalization of the correlation between the input data and the calculated outputs, something extremely difficult to do using the more conventional methods.

ACKNOWLEDGMENT

R.G. is indebted to the Wellcome Trust for financial support (Grant 042615/Z/94/Z) and R.I.G. D.B.K and A.M.W. to the UK EPSRC and the UK BBSRC for financial support.

Received for review May 5, 1997. Accepted for August 14, 1997.[⊗]

AC970460J

[⊗] Abstract published in *Advance ACS Abstracts*, October 1, 1997.