JOURNAL OF CHEMOMETRICS, VOL. 6, 267–281 (1992)

GENETIC ALGORITHMS AS A STRATEGY FOR FEATURE SELECTION

R. LEARDI, R. BOGGIA AND M. TERRILE

Istituto di Analisi e Tecnologie Farmaceutiche ed Alimentari, Via Brigata Salerno (Ponte), I-16147 Genova, Italy

SUMMARY

Genetic algorithms have been created as an optimization strategy to be used especially when complex response surfaces do not allow the use of better-known methods (simplex, experimental design techniques, etc.). This paper shows that these algorithms, conveniently modified, can also be a valuable tool in solving the feature selection problem. The subsets of variables selected by genetic algorithms are generally more efficient than those obtained by classical methods of feature selection, since they can produce a better result by using a lower number of features.

KEY WORDS Genetic algorithms Feature selection Multivariate analysis
Optimization methods

1. INTRODUCTION

One of the greatest problems in multivariate analysis is to select the combination of variables that produces the best result. This goal is attained through the elimination of those variables that produce noise or, though giving by themselves good information, are strictly correlated with other already selected variables.

Feature selection is very important both in studies of correlation (selection of the variables allowing one to build a mathematical model able to explain a response variable) and in studies of classification and modelling (selection of the variables allowing one to separate best between categories and to build a mathematical model able to describe the different categories with good specificity and sensitivity).

The goal of this paper is to show that genetic algorithms (GAs), though created as an alternative to optimization methods (experimental design, simplex, EVOP), can also be used in the completely different field of feature selection, where none of the previously cited methods can be applied.

Before showing the applications of GAs to feature selection, it is necessary to explain the theory of these algorithms and the field of applications in which they are normally used.

2. GENETIC ALGORITHMS

2.1. The optimization problem

Optimization is one of the most important steps during the set-up of a process: it makes it possible to obtain the best result from both a qualitative and quantitative point of view (e.g.

0886–9383/92/050267–15$12.50

Received 10 February 1992
Revised 29 July 1992

to obtain the highest possible yield of a chemical reaction) and/or from an economic viewpoint (e.g. to obtain a certain yield at the lowest cost).

When the number of factors involved in the process is very high, it is much more difficult to perform a good optimization.

With the univariate method each factor is studied independently by changing its value while keeping the values of the other factors constant. With this approach each factor is considered to behave independently of the others and so interactions cannot be studied. To do the latter, a multivariate technique must be used, since such methods allow one to change all the factors simultaneously.

Among the multivariate techniques, simplex, EVOP and the techniques of experimental design allow one to identify, after a relatively small number of experiments and to a good approximation, the experimental conditions leading to the desired response.

These techniques are perfectly suited to phenomena characterized by a response which can be well described by a simple mathematical model (Figure 1).

In contrast, they cannot be applied to phenomena whose response follows a very complex model or no model at all (Figure 2). In such cases the only way to be almost sure of finding the best response is to perform a grid search spanning the whole space of the variables. Of course, the more complex the model (if any) is, the thicker has to be the grid along which the search is done. This type of approach certainly gives a very good knowledge of the phenomenon, but the number of requirements very soon becomes unbearable as the number of factors grows.

A simplification of the grid search could be a random choice of some experiments among

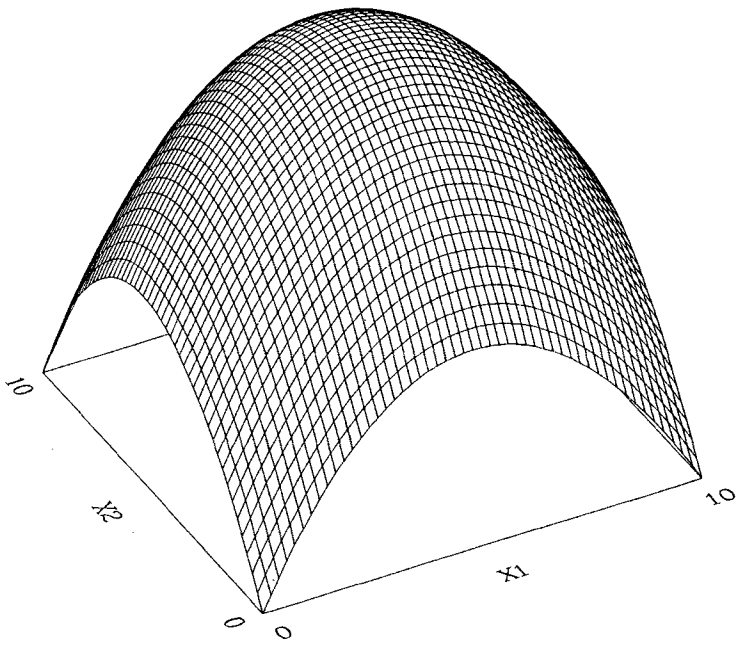

Figure 1. Response surface obtained by $Y = X_1 + X_2 - 0{\cdot}1X_1^2 - 0{\cdot}1X_2^2$. In such a case experimental design, simplex and EVOP are perfectly suited to find the maximum

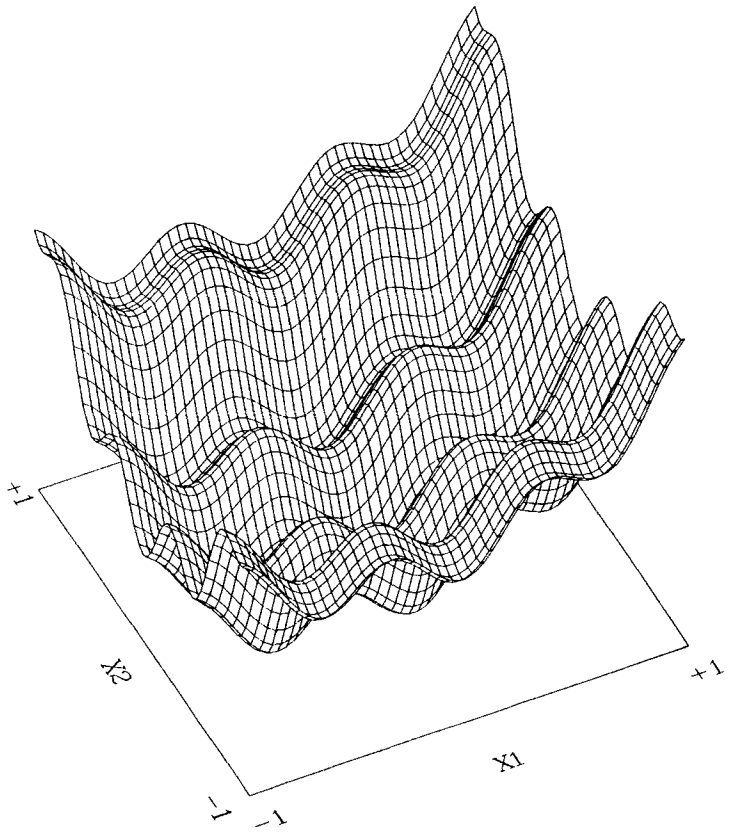

Figure 2. Response surface obtained by $Y = 0{\cdot}5X_1 + 0{\cdot}2X_2 + X_1^2 + 2X_2^2 - 0{\cdot}3 \cos(3\pi X_1) - 0{\cdot}4 \cos(4\pi X_2)$. It is evident that in such a case experimental design, simplex and EVOP cannot be applied

those proposed by the original grid, followed by some more experiments performed in the zones surrounding the best responses.

Such an approach is suggested by GAs.

2.2. Evolution theory in the biological world

GAs are inspired by evolution theory, according to which the evolution of a species is mainly ruled by the 'struggle for life'. Under this principle the 'best' individuals (i.e. those beings whose genetic material is best for the environmental conditions in which they live) have both the greatest probabilities of surviving and the greatest probabilities of winning the battles engaged for reproduction, thereby propagating their genome. Furthermore, when two good individuals mate, the combination of their genomes can generate offspring with even better genetic material. As a result, from this particular point of view a population evolves in such a way that the 'good' genome is more and more present in it.

Another source of variation is given by mutations. They are irregular changes with a very low probability of occurrence and they affect a single gene. They generally result in a pathological condition, but sometimes such a random change can produce a good result and thus contribute to the evolution of the population.

2.3. Evolution theory according to genetic algorithms

GAs take inspiration from evolution theory in the biological world, under the same assumption that the experimental conditions leading to better results will prevail over the worst ones and that an improvement can be obtained by some sort of recombination together with some random changes.

From this point of view the experimental conditions are considered as the genome (a single chromosome for the sake of simplicity) whose genes are the variables taking part in the process. Thus the fitness of each experimental condition is measured by a response which has to be optimized (e.g. the yield of a chemical reaction).

As in real life, this population will evolve towards the best through reproductions and mutations.

3. THEORY

3.1. The original algorithm[1-7]

GAs have five basic steps: 1, coding of variables; 2, initiation of population; 3, evaluation of the responses; 4, reproductions; 5, mutations. Steps 3–5 alternate until a termination criterion is reached; this criterion can be based on a lack of improvement in the response or simply on a maximum number of generations or on the total time allowed for the elaboration.

Let us now examine the five steps in more detail.

3.1.1. The coding

To allow an easy mathematical treatment of the 'chromosomes', a coding is needed. This problem is easily solved by the binary coding, so that each experimental condition is represented by a string of 0s and 1s.

As an example, let us suppose we are studying a process in which four variables are under investigation: temperature, reaction time, stirring and catalyst type.

Every experimental condition is considered to be a chromosome containing four genes, each gene containing the information about the corresponding variable.

Thus the experimental condition characterized by 30 °C, 60 min, no stirring and catalyst A will be coded as

0011110 0111100 0 0

with bits 1–7 (0011110) being the 'gene' temperature, bits 8–14 (0111100) the 'gene' reaction time, bit 15 (0) the 'gene' stirring and bit 16 (0) the 'gene' catalyst. ('Blanks' between genes have been used only for easier reading and are not present in the chromosome.)

3.1.2. Initiation of population

The original population is composed of a certain number N of chromosomes (normally between 50 and 500 depending on the dimension of the problem). After having decided the order of the genes in the chromosomes, the structure of each chromosome (i.e. the location of the experimental point in the space of the variables) is determined in a totally random way, by drawing the values assumed by the variables and coding the resulting experimental conditions (see Section 3.1.1).

3.1.3. Evaluation of the response

For each chromosome the response associated with the corresponding experimental conditions is evaluated. If the experimental conditions lie outside the experimental domain or correspond to an experiment impossible to perform, a null response can be given so that it will not spread to the following generation.

3.1.4. Reproduction

This step creates a new population of N chromosomes which can be considered as the next generation. It can be divided into two substeps: select–copy and cross-over.

3.1.4.1. Select–copy In this phase the new population of N chromosomes is produced by applying the 'select–copy' operator N times. This operator randomly selects a chromosome by the drawing of a random number. The probability of a particular chromosome being selected is a function of its associated response, so that the best ones have a greater probability of being picked up than the worst ones. A simple way to determine the probability is:

$$\text{prob}(i) = \text{response}(i)/\text{sum}(\text{responses})$$

Following this step, a new population is obtained in which the best chromosomes are copied more often; this leads to a better average response. This step only raises the percentage of good chromosomes relative to the parent population, i.e. no improvement at the level of the single chromosomes is attempted. This will take place in the next phase.

3.1.4.2. Cross-over The N chromosomes forming the new population are randomly paired to form $N/2$ pairs.

For each of them a random number is drawn to decide whether it has to undergo a cross-over. Generally, the probability of cross-over is set at a very high level (about 0·9), so that almost all the pairs undergo this operation. If a cross-over is to happen, then a cross-over point c is randomly selected, ranging from 1 to $g-1$, with g being the number of genes. After having selected the point c, two 'offspring' are generated: 'offspring' 1 takes genes 1 to c from 'parent' 1 and genes $c+1$ to g from 'parent' 2, while 'offspring' 2 takes genes 1 to c from 'parent' 2 and genes $c+1$ to g from 'parent' 1. The two offspring will then take the place of their parents in the new population.

According to this algorithm, in a cross-over, two consecutive genes in the parent chromosomes have a much higher probability of being found unchanged in the offspring than two genes at the extremes. With g genes there are $g-1$ possible cross-over sites and only one disrupts the sequence of two consecutive genes, which means a probability of $1/(g-1)$, i.e. 10% in the case of eleven genes, while all the cross-over sites will disrupt the two extremes. This also means that the order in which the variables are coded can be important.

The cross-over allows one to explore new experimental conditions by mixing values of variables already tested, though in different combinations.

Furthermore, if after the reproduction step all the chromosomes were to have the same gene with the same value, a deadlock situation would occur for that gene unless some random changes could take place. This is made possible by mutations.

3.1.5. Mutations

In this step some bits, randomly selected, change their status, becoming 0s if they were 1s and vice versa. To do this, for each bit of each chromosome a random number is drawn to decide whether it has to change. The probability of mutations is set to a rather small value, generally about 1%; e.g. for a population of 100 chromosomes, each composed of 50 bits, an average of $0{\cdot}01 \times 100 \times 50 = 50$ mutations occur in every cycle.

Following this step, each single variable can assume values never tried before; it is like performing big jumps in the experimental space.

After reproductions and mutations the new generation replaces the previous one and the algorithm continues from the evaluation of the response (see Section 3.1.3.)

3.2. Genetic algorithms applied to feature selection

With a data set in which each object is described by k variables, the problem is totally equivalent to the optimization of a process whose parameters are k nominal variables (yes/no). Each chromosome is then composed of k genes, each gene being formed by a single bit.

As an example, with ten variables, the combination taking into account variables 1, 5, 8 and 9 will be coded as 1000100110.

The response will be the cross-validated variance explained by the selected variable combination (in regression techniques) or the percentage of correct predictions (in classification and modelling techniques).

All the studies here reported have been performed with multiple linear regression (MLR), but the conclusions can easily be transferred to all other techniques, since the method used does not interfere with the GA.

The generally used feature selection methods belong to the 'stepwise' family:[8–18] they add or remove one variable at a time from a pool of variables. The 'forward' methods are rather fast, while the 'backward' methods require a very long computing time and cannot always be used (e.g. in MLR when the number of variables is higher than the number of objects).

Another method is based on decorrelation:[9,19] in it the best variable (i.e. the one having the highest correlation with the response in the case of regression analysis or having the highest discriminant power in the case of classification analysis) is selected and the other variables are decorrelated from it; the process continues on the decorrelated variables until no decorrelated variable has a correlation value (or a discriminating power) higher than the stop value.

The main problem of all these methods is connected with the algorithms themselves, since every step heavily conditions the subsequent choices, the possibility of exploring some variable combinations having been removed; it is like eliminating a very large part of the experimental domain.

The best strategy would be the evaluation of the results given by all possible combinations,[12–16] but as the number of variables increases, the number of combinations quickly becomes too high to be fully explored. With k variables there are in fact $2^k - 1$ possibilities.

3.3. Modifications to the original algorithm

The original algorithm is intended to work in a continuous space, while in our case the space under investigation is absolutely discontinuous: when working with k variables, it is like studying only the vertices of a k-dimensional hypercube. Several changes have thus been applied to the original algorithm to try to adapt it best to the specific purpose.

3.3.1. Initiation of population

The original algorithm says that the value of each bit is determined by the 'toss of a coin'. Under this hypothesis an average of 50% of the variables would be selected in every initial chromosome. This can lead to impossible combinations (e.g. when there are more variables than objects in MLR); furthermore, when only a few variables are selected, the elaboration time is much smaller and the information is much clearer.

An initial probability of selection has thus been added so that the average number of variables present in the initial population can be selected. Since it can happen that the user has constraints on the maximum number of variables to be used, the maximum number of variables present in each combination is asked for. This constraint will also be valuable in the subsequent phases. After the evaluation of the response the new chromosome is inserted into the ordered array of the previously created chromosomes. As a consequence chromosome $\#n$ is the nth best chromosome in the population.

3.3.2. Reproduction

The original algorithm, after having generated another population by copying elements of the previous one, randomly selects two parents and a crossing-over point; it then swaps between the two parents all the genes to the right of the selected point. This means that the order of coding is not indifferent.

The modified algorithm performs the choice of a couple of parents; immediately after that it draws the genes to swap. To avoid dependence on the coding, the drawing is repeated for each gene, so that the swap takes place for single genes and not for a series of contiguous genes.

The number of swaps is determined by a probability value: when it is 0·5, each gene has 50% probability of being swapped, so that the maximum 'fusion' between the two parents takes place; the lower this value, the more conservative will be the offspring, i.e. each of them will be more similar to one of the parents.

Immediately after the 'birth' of the two offsprings their response is evaluated, so that they can replace some members of the previous population (see Section 3.3.4.).

The number of reproductions in each cycle can be selected at the beginning of the programme.

3.3.3. Mutations

In the original algorithm, reproduction and mutations are strictly connected, since the evaluation of the response takes place only at the end of the mutations; here, in contrast, the two steps are separated and they continually alternate up to the end.

After all the mutations have been performed, the response of the chromosomes having undergone one or more mutations is evaluated, so that they can replace some members of the previous population (see Section 3.3.4.).

3.3.4. Generation overlapping and insertion of new chromosomes

According to the original algorithm, the chromosomes of generation $n + 1$ completely take the place of the chromosomes of generation n, so that no overlapping between generations is allowed; by doing this, very good chromosomes can be lost.

In the modified algorithm, complete generation overlapping is allowed, with the parents coexisting with their offspring and the mutants coexisting with the mutated.

As previously stated, the goodness of a subset of variables can be determined either by the response it gives or by the number of features it uses. Thus it is rather important to know the best result obtained by using a certain number of variables; this chromosome is highly informative and deserves to be saved, regardless of its absolute position in the scale of the responses. To do this, such a chromosome is 'protected' and cannot be eliminated from the population; its particular condition will end when another chromosome, using a number of features no greater than that it used, will give a better result.

After the evaluation of its response, for each chromosome it has to be decided whether to insert it into the population and, if so, what chromosome of the population to discard.

As stated before, a chromosome using k variables is protected when it gives the best response among all the chromosomes using at most k variables.

If the new chromosome is a protected one, then it will be a member of the population; it will enter the array of ordered chromosomes and the worst non-protected chromosome will be eliminated. If the new chromosome is non-protected, then it will be a member of the population only if its response is better than the response of the worst non-protected chromosome; in this case it will enter the array of ordered chromosomes and the worst non-protected chromosome will be eliminated.

By doing this, at each moment the population is composed of the best absolute chromosomes and of those chromosomes that are highly informative since they give the best result with a limited number of variables.

Techniques such as simulated annealing and Boltzmann jump have also an explicit probability of survival for non-optimal cases, so that the function is allowed to pass from one local minimum to another through a suboptimal path (with increased energy). This is not the case in GAs applied to feature selection, since they work in a discontinuous space.

Table 1 shows a simulated population ($N = 10$ chromosomes). In it the following chromosomes are protected: #1 (highest response with $v \leqslant 7$), #4 (for $v \leqslant 5$), #7 (for $v \leqslant 4$), #8 (for $v \leqslant 3$), #9 (for $v \leqslant 2$) and #10 (for $v = 1$). Chromosome #5 is non-protected since, though it is the best for $v = 6$, it is dominated by chromosome #4 which gives a better result with a lower number of variables.

Table 1. Simulated population of ten chromosomes. The protected chromosomes are marked by *

Chromosome #	Response	Number of variables selected (v)
1	80·35	7*
2	78·32	7
3	75·47	8
4	70·32	5*
5	68·43	6
6	65·13	5
7	60·65	4*
8	50·43	3*
9	40·71	2*
10	30·09	1*

Let us assume four cases.

(a) The new chromosome has a response of 55·13 with $v = 4$; it is non-protected and its response is lower than the lowest response of a non-protected chromosome (65·13); the chromosome is discarded.
(b) The new chromosome has a response of 68·19 with $v = 7$; it is non-protected and better than the worst non-protected chromosome (65·13); the chromosome enters the population and chromosome #6 is discarded).
(c) The new chromosome has a response of 72·14 with $v = 5$; it is protected and enters the population; chromosome #4 becomes non-protected and chromosome #6 is discarded.
(d) The new chromosome has a response of 42·11 with $v = 2$; it is protected and enters the population; chromosome #9 becomes non-protected and is discarded.

3.3.5. Control of replicates

The original algorithm does not check the presence of 'twins', so that the same chromosome can be present more than once. In the modified algorithm this is not allowed and every time a new chromosome is created (either in the original population or in the reproduction step or in the mutation step) it is immediately checked; if a 'twin' is found, then it is discarded and the creation of another chromosome takes place.

As a result of these modifications, when the termination criterion is reached, the actual population is formed by highly informative combinations. The expert of the problem then has a wide choice and the selection of the variable combination can take place on the basis of the result and of the variables to be used.

Very often, in fact, it can happen that it is more convenient to have a slightly worse result if it can be attained with a lower number of variables or without some variables whose measure is particularly difficult or expensive.

3.3.6. Influence of the different parameters

In the modified GA, five parameters (number of chromosomes, probability of initial selection, probability of cross-over, number of reproductions and probability of mutation) have to be defined; it is then very important to know if and how they affect the final results. To study this problem, several runs have been performed on different data sets.

It has to be noted that in this sort of application the 'evaluation of the response' is the step requiring by far the greatest computation time, since this subroutine is a true multivariate module (e.g. MLR or linear discriminant analysis). It is then very important that the algorithm is directed in the way of performing this step the lowest number of times and with the highest possible speed.

Number of chromosomes. A high number of chromosomes produces a loss in the final result, since the greater time required by a new generation leads to a lower number of generations in the same time; a population of about 30 elements, independent of the number of original variables, seems to be a good compromise between genetic variety and time.

Probability of initial selection. A low value of this parameter allows one to explore many more combinations in the same time, with information much easier to interpret. The algorithm itself will build more complex combinations through the reproduction phase. A good value seems to be one that selects an average of five variables per chromosome (e.g. 0·10 when working with 50 variables).

Probability of cross-over. A high mixing of the genomes of the parents improves the final result; the highest value of 0·5 seems to be the best.

Number of reproductions. The reproduction phase is the 'heart' of the algorithm; a number of reproductions equal to half the number of chromosomes (so that the number of offspring generated in each cycle equals the population dimension) gives a good emphasis to this part of the algorithm.

Probability of mutation. The mutation step allows one to avoid deadlock situations and to 'jump' to new zones of the space. A very high value of this parameter disrupts the configuration of the chromosomes too much, with a low probability of obtaining a good result, while a very low value does not give this step the importance it deserves. A good compromise is the probability producing on average one mutation per chromosome (e.g. 0·02 when working with 50 variables).

The several runs performed in trying to optimize these parameters have anyway shown that they have a wide range of validity, so that a previous study of the data set is not required and the same values can always be applied.

For a data set of 50 variables a suggested set-up is the following:

(i) number of chromosomes, 30
(ii) probability of initial selection, 0·10
(iii) probability of cross-over, 0·50
(iv) number of reproductions, 15
(v) probability of mutation, 0·02.

4. EXPERIMENTAL

4.1. Test data set

To verify if this technique can really lead to the choice of the best combination in a reasonable time, a synthetic data set formed by 16 objects described by eleven variables and a response has been used (Table 2). Owing to the relatively small dimension of the data matrix, all 2047 combinations have been evaluated.

This exhaustive exploration required 68 min (all the programmes have been written in QuickBasic and have been run on an Olivetti PcPro 486/33 mt having a 80486 coprocessor with a clock of 33 MHz); the best combination has been found to be the one retaining variables 1, 3, 4, 6, 7, 8, 10 and 11, with a cross-validated (CV) variance of 90·32% (16 deletion groups, leave-one-out method). Table 3 shows the ten best combinations.

On this data set the GA has then been applied ten times, the termination criterion being a computation time of 17 min, i.e. 25% of the time of the exhaustive exploration. The results are shown in Table 4.

It can easily be seen that the results are very encouraging, since the best combination has always been found, in an average time of just over 6 min and with a worst result of 11 min 50 s. Even more interesting is to note how this algorithm can find almost all the best combinations in a time much less than that required by an exhaustive search. Nine times out of ten it found the ten best combinations and on the worst occasion chromosome #20 corresponded to the 22nd best combination (i.e. it missed just two chromosomes among the 22 top-ranked ones); five times the algorithm scored a perfect 20 out of 20.

It is interesting to compare these results with those obtained by other techniques.

Table 2. Data set used to test the modified GA

Object	Variable number											
	1	2	3	4	5	6	7	8	9	10	11	*Y*
1	5	0	15	−1	0	75	−5	0	0	−15	25	123
2	5	4	15	1	20	75	5	60	4	15	25	406
3	15	4	15	−1	60	225	−15	60	−4	−15	225	1400
4	5	0	160	1	0	800	5	0	0	160	25	352
5	15	0	160	−1	0	2400	−15	0	0	−160	225	1740
6	5	4	160	−1	20	800	−5	640	−4	−160	25	600
7	15	4	160	1	60	2400	15	640	4	160	225	1820
8	25	0	15	−1	0	375	−25	0	0	−15	625	1300
9	25	4	15	1	100	375	25	60	4	15	625	2510
10	25	0	160	1	0	4000	25	0	0	160	625	1950
11	25	4	160	−1	100	4000	−25	640	−4	−160	625	1270
12	2	0	15	1	0	30	2	0	0	15	4	179
13	7	0	15	−1	0	105	−7	0	0	−15	49	420
14	7	0	15	−1	0	105	−7	0	0	−15	49	662
15	2	0	160	−1	0	320	−2	0	0	−160	4	166
16	7	0	160	1	0	1120	7	0	0	160	49	1110

Stepwise regression (programme STEPREG of PARVUS[9]), with F-to-enter = 2 and F-to-remove = 1, selects, in order, variables 1, 7, 11, 10 and 4 in 11 s (see Table 5).

Before comparing the two results, it has to be considered that the classical stepwise selection adds the variable causing the greatest increase in fitting without undergoing any cross-validation. Thus it can happen that the addition of a variable, though producing a best fit, causes a worst predictive ability. In contrast, the new algorithm always works on cross-validated variance.

From the analysis of the results (Table 3) one can see that the combination chosen by STEPREG explains 87·86% of the cross-validated variance and is placed only in sixth position with respect to the predictive ability. Furthermore, combination 3 explains more cross-validated variance (88·43%) with four variables only.

Regarding the comparison of computer time, it has to be stressed that the leave-one-out

Table 3. The best variable combinations of the test data set (16 deletion groups)

Combination	Number of variables	Selected variables	% CV variance
1	8	1 3 4 6 7 8 10 11	90·32
2	7	1 3 4 6 7 10 11	89·50
3	4	1 7 10 11	88·43
4	5	1 7 8 10 11	88·02
5	6	1 4 7 8 10 11	88·02
6	5	1 4 7 10 11	87·86
7	6	1 3 6 7 10 11	87·79
8	6	1 4 6 7 10 11	87·73
9	8	1 3 4 6 7 9 10 11	87·65
10	8	1 3 4 5 6 7 10 11	87·27

Table 4. Results obtained with the modified algorithm on the test data set (30 chromosomes; probability of initial selection, 0·45; probability of cross-over, 0·50; 15 reproductions per cycle; probability of mutation, 0·09; termination criterion 17 min; cross-validation with 16 deletion groups). The reported number corresponds to the position occupied by the chromosome in the ranking of the 2047 possible combinations. When a perfect score has been obtained, the time is also reported

Run	Chromosome #1	Chromosome #10	Chromosome #20
1	1 (7 min 01 s)	10 (13 min 58 s)	20 (13 min 58 s)
2	1 (6 min 34 s)	10 (13 min 44 s)	20 (13 min 44 s)
3	1 (5 min 42 s)	10 (11 min 11 s)	21
4	1 (1 min 01 s)	10 (7 min 27 s)	21
5	1 (6 min 04 s)	10 (15 min 39 s)	21
6	1 (5 min 26 s)	10 (12 min 58 s)	20 (15 min 45 s)
7	1 (9 min 43 s)	10 (14 min 51 s)	22
8	1 (2 min 17 s)	10 (10 min 45 s)	20 (15 min 19 s)
9	1 (6 min 37 s)	10 (12 min 20 s)	20 (13 min 55 s)
10	1 (11 min 50 s)	12	22

Table 5. Output of programme STEPREG on the test data set (F-to-enter = 2, F-to-remove = 1). F-to-enter for each variable at each cycle is reported. The variable entered is marked by*

	Cycle					
Variable	1	2	3	4	5	6
1	35·83*	—	—	—	—	—
2	2·08	0·77	0·70	0·16	0·16	1·37
3	0·44	0·77	0·47	0·43	0·22	0·09
4	0·80	2·74	0·11	0·05	2·10*	—
5	6·74	0·60	0·51	0·47	0·00	0·27
6	5·67	0·06	0·00	0·12	0·31	0·75
7	1·36	6·31*	—	—	—	—
8	0·49	0·02	0·16	0·01	0·21	0·69
9	0·63	2·37	0·03	0·01	0·00	0·09
10	0·54	1·15	0·23	5·02*	—	—
11	21·49	5·19	14·72*	—	—	—

method performs as many cycles as objects; with this data set each combination of variables has been tested 16 times.

When the technique based on decorrelation (programme SELECT of PARVUS) is applied, the same variables chosen by STEPREG are picked up in the same order.

4.2. Real data set ('CHEESE')

As an example on real data, the results obtained on the data set 'CHEESE' are reported. In this data set, 41 samples of 'provola' cheese, a typical cheese produced in the South of Italy, are described by 69 chemical variables (22 aminoacids, 17 fatty acids, humidity, fat, 28

glycerides); the response variable is the age of the sample (see Table 6). The goal of the study is to find a relationship between the chemical composition and the age.

First, STEPREG has been performed with F-to-enter = 2 and F-to-remove = 1; the following twelve variables have been selected: 3, 4, 5, 10, 12, 13, 40, 46, 47, 55, 60, 67 (time 4 min 10 s). To compare the efficiency of this choice with those produced by the GA, the cross-validated variance has been computed: with five deletion groups it was 83·03%.

SELECT (stop value 0·10) chose six variables (4, 40, 46, 55, 60, 64); with five deletion groups the cross-validated variance is 70·05%.

On this data set the GA has been run five times, with a termination criterion of 2 h. To make the results more easily comparable with those obtained by STEPREG, a maximum number of twelve chosen variables has been set.

From the analysis of the results (Table 7) it is evident that the cross-validated variance explained by the chosen combinations is almost always higher than that explained by the subset selected by STEPREG.

It is also evident that the choice operated by SELECT appears to be totally unsatisfactory, since the GA obtains the same variance by using just two or three variables.

Furthermore, the fact that very good results are associated with all the chromosomes of the population shows that at the end of the GA the user can choose among a very large number of suitable solutions.

Table 8 shows the variables chosen by STEPREG, SELECT and by each of the five runs of the GA. It is evident that the GA shows a certain degree of convergence, since eight variables (4, 5, 10, 12, 23, 34, 59, 69) have been selected three or more times.

Since particularly unlucky runs can produce not very good results (see run 3), it is always advisable to perform at least two runs on the same dataset.

Table 6. Data set 'CHEESE': list of variables

1) ASP	24) C10:0	47) C4:0 1,3-diglyceride
2) GLU	25) C12:0	48) C4:0 monoglyceride
3) ASN	26) C14:0	49) C4:0 triglyceride
4) SER	27) C16:0	50) C6:0 1,2-diglyceride
5) GLN	28) C18:0	51) C6:0 1,3-diglyceride
6) HIS	29) C18:1	52) C6:0 monoglyceride
7) GLY	30) C18:2	53) C6:0 triglyceride
8) THR	31) Other fatty acids	54) C14:0 1,2-diglyceride
9) CIT	32) Total non-volatiles	55) C14:0 1,3-diglyceride
10) ARG	33) C2:0	56) C14:0 monoglyceride
11) ALA	34) C3:0	57) C14:0 triglyceride
12) GABA	35) C4:0	58) C18:1 1,2-diglyceride
13) TYR	36) C5i:0	59) C18:1 1,3-diglyceride
14) AABA	37) C5:0	60) C18:1 monoglyceride
15) MET	38) C6:0	61) C18:1 triglyceride
16) VAL	39) Total volatiles	62) C18:0 1,2-diglyceride
17) PHE	40) Humidity	63) C18:0 1,3-diglyceride
18) ILE	41) Total fat	64) C18:0 monoglyceride
19) LEU	42) Triglycerides	65) C18:0 triglyceride
20) ORN	43) 1,3-Diglycerides	66) C16:0 1,2-diglyceride
21) LYS	44) 1,2-Diglycerides	67) C16:0 1,3-diglyceride
22) Total amino acids	45) Monoglycerides	68) C16:0 monoglyceride
23) C8:0	46) C4:0 1,2-diglyceride	69) C16:0 triglyceride

Table 7. Data set 'CHEESE': results obtained by the GA (30 chromosomes; probability of initial selection, 0·0725; probability of cross-over, 0·5; 15 reproductions per cycle; probability of mutation, 0·0145; five deletion groups; maximum number of variables, twelve; termination criterion, 2 h). For each run the table reports the percentage of cross-validated variance explained by the following chromosomes: #1, #20, the best one with six variables and the best one with three variables (the third and fourth values to be compared with the 70·05% explained by the six variables chosen by SELECT)

	% CV variance			
Run	chromosome #1	Chromosome #20	Six variables	Three variables
1	88·50	87·95	77·53	71·77
2	85·35	84·83	80·66	71·57
3	81·35	80·58	79·53	72·94
4	90·59	88·23	82·03	72·40
5	83·96	82·79	78·09	71·61

Table 8. Data set 'CHEESE': variables chosen by the different methods of feature selection

STEPREG	3	4	5	10	12	13	40	46	47	55	60	67
SELECT	4	40	46	55	60	64						
GA (run 1)	1	4	5	8	10	12	23	37	48	61	67	68
(run 2)	3	4	5	10	23	34	45	59	62	63	68	69
(run 3)	10	15	23	32	34	52	59	60	64	66	67	
(run 4)	3	5	6	9	12	18	34	59	61	62	64	68
(run 5)	4	7	8	12	21	23	32	36	58	59	60	63

Regarding the computer time, the total ratio with STEPREG is about 30:1; when taking into account the fact that the GA ran with five deletion groups, the real ratio is less than 6:1. Of course, longer runs would produce even better results, but the amelioration could be insignificant when compared with the longer time needed.

5. CONCLUSIONS

This paper shows that GAs can be a valuable approach in the problem of determining relevant variables. After some changes to the original algorithm, they can be effectively applied to problems of regression, classification and modelling, with the advantage of proposing, in a reasonable time, a wide range of solutions, among which the expert of the problem can choose the most suitable one.

The source code for the GA is available from the authors upon request.

ACKNOWLEDGEMENTS

This work received financial support from the Educational Department (MPI 40% and 60%) and from the National Council of Research.

REFERENCES

1. L. B. Booker, D. E. Goldberg and J. H. Holland, *Artific. Intell.* **40**, 235 (1989).
2. L. Davis, *Genetic Algorithms and Simulated Annealing*, Pitman, London (1987).
3. D. E. Goldberg, *Genetic Algorithms in Search, Optimization & Machine Learning*, Addison-Wesley, Reading, MA (1988).
4. D. E. Goldberg and J. H. Holland, *Machine Learning*, **3**, 95 (1988).
5. C. B. Lucasius and G. Kateman, *Trends in Analytical Chemistry*, **10**, (8), 254 (1991).
6. H. Muehlenbein, M. Gorges-Schleuter and O. Kraemer, *Parallel Comput.* **7**, 65 (1988).
7. C. T. Walbridge, *Technol. Rev.* **47**, 47–53 (January 1989).
8. N. Draper and H. Smith, *Applied Regression Analysis*, 2nd edn, Wiley Interscience, New York (1981).
9. M. Forina, R. Leardi, C. Armanino and S. Lanteri, *PARVUS: An Extendable Package of Programs for Data Exploration, Classification and Correlation*, Elsevier, Amsterdam (1988).
10. W. J. Kennedy and J. E. Gentle, *Statistical Computing*, Marcel Dekker, New York (1980).
11. *BMD—Biomedical Computer Programs*, University of California Press, Los Angeles, CA (1973).
12. D. A. Belsley, E. Kuh and R. Welsch, *Regression Diagnostics—Identifying Influential Data and Sources of Collinearity*, pp. 34–35, Wiley Series in Mathematics and Statistics, New York (1980).
13. F. Mosteller and J. W. Tukey, *Data Analysis and Regression—A Second Course in Statistics*, pp. 387–393, Addison-Wesley, Reading, MA (1977).
14. C. Daniel and F. S. Wood, *Fitting Equations to Data*, Wiley, New York (1971).
15. G. M. Furnival, *Technometrics*, **13**, 403 (1971).
16. G. M. Furnival and R. W. Wilson Jr., *Technometrics*, **16**, 499 (1974).
17. D. L. Massart, B. G. M. Vandeginste, S. N. Deming, Y. Michotte and L. Kaufman, *Chemometrics: A Textbook*, pp. 407–409, Elsevier, Amsterdam (1988).
18. W. R. Dillon and M. Goldstein, *Multivariate analysis. Methods and Applications*, pp. 375–379, Wiley Interscience, New York (1984).
19. M. A. Sharaf, D. L. Illman and B. R. Kowalski, *Chemometrics*, pp. 239–242, Wiley–Interscience, New York (1986).