9. Genetic Algorithms

9.1 Introduction

The concept of evolution is prevalent in most biological systems. Generation after generation, certain characteristics are encouraged if they fit the environment. A classic example of natural selection is the peppered moth [49]. Both light and dark moths were present during the industrial era. But the latter became the prominent variety because of changing environmental conditions. The trees darkened as a result of the pollution, so the white moths began to alter their pigmentation and evolve with a peppered exterior. Natural camouflage against predatory birds therefore increased their chances of survival. This system of natural selection can, in principle, be applied to computational optimisation methods using "genetic algorithms" [50].

9.2 Neural Networks and Genetic Algorithms

As discussed earlier, neural networks can be used to perform complex regression with a large number of inputs. The resulting non-linear models can capture their interactions, but we may wish to use the neural network backwards and identify sets of input variables resulting in a desired output value. However, the large numbers of variables and the non-linear character of the model render this difficult.

The problem is that of looking for the input set (x_i), which will give a desired output y, with:

$$y = f(x_1, x_2, x_3 \ldots\ldots x_n) \tag{9.1}$$

with the function f being non-linear. Genetic algorithms (GAs) is one possible method of solving such a problem. Goldberg described these algorithms as "search procedures based on the mechanics of natural selection and natural genetics" [50].

Once a target output value t is selected, the GA randomly generates sets of inputs called "chromosomes", which exist in populations:

$$\mathrm{X}_i = \left[x_{i1}, x_{i2}, x_{i3}, \ldots x_{iz}\right] \tag{9.2}$$

Each chromosome X_i is composed of a set of variables $\{x_{i1\ldots} x_{iz}\}$ called "genes". These sets are then ranked according to a fitness factor, which describes how well they perform. In the above example, a fitness factor F could be:

$$F = \frac{1}{(y_i - f_i)^2} \tag{9.3}$$

where y_i is the desired output for an i^{th} set of inputs. Those that are close to the desired output are allowed to "breed" in order to generate another z sets of inputs, whereas the others are eliminated. In this way, it should be possible to evolve towards the correct set of inputs, hence demonstrating that the GA process is based on the concept of "survival of the fittest".

9.2.1 Creating the next generation

Chromosomes with a high fitness have a good chance of being "reproduced" into the next generation. This therefore increases the number of fit chromosomes in subsequent generations.

Crossover is the next operator to follow. This process typically involves a pair of chromosomes, X_i and X_j, that can be randomly selected from any population of the new generation [51]:

$$\mathrm{X}_i = \left\lfloor x_{i1}, x_{i2}, x_{i3}, x_{ik}, x_{i(k+1)} \ldots x_{iz} \right\rfloor \tag{9.4a}$$

$$\mathrm{X}_j = \left\lfloor x_{j1}, x_{j2}, x_{j3}, x_{jk}, x_{j(k+1)} \ldots x_{jz} \right\rfloor \tag{9.4b}$$

where x_{i1} is the first gene of chromosome i and z is the number of genes.

Both X_i and X_j will crossover and exchange information to create two new chromosomes X_i' and X_j':

$$\mathrm{X}_i{}' = \left[x_{i1,}x_{i2},x_{i3},x_{ik},x_{j(k+1)}\ldots x_{jz}\right] \tag{9.5a}$$

$$\mathrm{X}_j{}' = \left[x_{j1,}x_{j2},x_{j3},x_{jk},x_{i(k+1)}\ldots x_{iz}\right] \tag{9.5b}$$

The final operator is mutation, where some genes are altered at random. This helps preserve the diversity of the GA and also to prevent early convergence within local minima. However, this raises the issue of a satisfactory mutation rate. It is important that mutations are large enough to escape local minima, but small enough to ensure that searching does not become stochastic. A possible compromise may be to have a multitude of populations but this may not be necessary if a large number of genes are being generated. The disadvantage then is the speed of convergence, so an investigation into the appropriate number of populations is necessary.

In parallel, there is also an introduction of new genes to populations during the optimisation process. These are again chosen at random, as with the initialisation of a new population. The same condition of non-negative values is also applied here, as well as the crossover and mutation processes.

In summary, each population contains chromosomes that begin with random values. Those that are fittest are reproduced to the next generation, where they mate and swap chromosome sections. Mutations also occur occasionally to randomly selected genes to preserve diversity.

After each generation, the values are continually fed into the neural network, where the score is calculated and ranking follows. This cyclical process continues until the user specifies the number of generations they require or until a desired accuracy, with respect to the target, is reached. The hope is that the optimal solution is found by the end of the cycle.

9.3 Application to Neural Networks

Figure 9.1 outlines the GA process in a flowchart, but the following sections outline important aspects of optimisation.

9.3.1 Normalised inputs for genes

The initial populations are given random input values for every gene of each chromosome, chosen by a random number generator. The range of each variable in the database used for training the neural network was normalised between ±0.5, as described previously. However, for the purpose of GAs, the values are allowed to exist beyond this range and search a wider range of input space. The error bars associated with these predictions may well be large, as seen in table 9.1, but knowledge of uncertainty is the most critical issue. Indeed, the level of uncertainty can be used as a constraint in setting the environment.

Population Number	***Chromosome Number***	***Cr* / wt %**	***Ni* / wt %**	***Mn* / wt %**	***Mo* / wt %**	***C* / wt %**	***Result* / MPa**	**± *Error* / MPa**
1	1	14.31	7.29	1.31	0.75	0.14	77.95	84.83
1	2	14.31	7.29	1.31	1.62	0.17	78.37	102.17
1	3	14.31	7.29	1.31	1.62	0.17	78.37	102.17
1	4	14.31	15.21	1.31	2.28	0.07	73.20	72.89
2	1	14.53	11.35	0.21	1.86	0.11	75.81	81.03
2	2	19.11	15.21	2.88	1.49	0.015	76.35	74.88
2	3	19.11	15.21	2.88	1.49	0.015	76. 35	74.88
2	4	19.11	15.21	2.88	1.49	0.015	76. 35	74.88

Table 9.1 An example of generated data from the GA of two populations with four chromosomes, each having five sets of genes, together with predictions and associated error, which accounts for test error and fitting uncertainty.

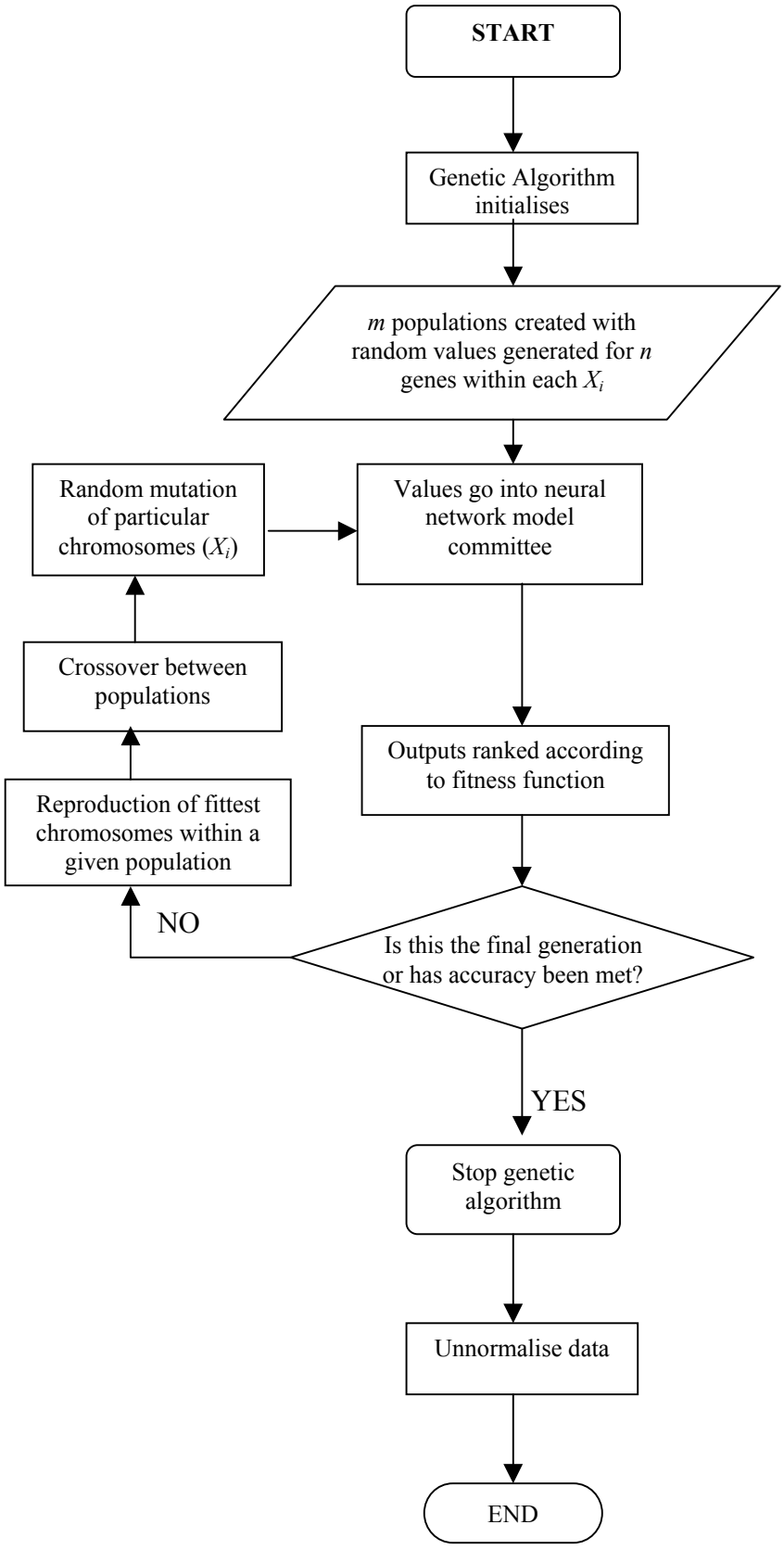

Figure 9.1 - Overview of the GA process.

9.3.2 Physically meaningful inputs

As the GA searches the input space, there is a distinct possibility that negative values are studied for variables, which cannot be less than zero. For these variables, the search was confined to regions where:

$$x_n \geq -\left[\left(\frac{x_{\min}}{x_{\max} - x_{\min}}\right) + 0.5\right] \tag{9.6}$$

9.3.3 Fitness function

The simplest method to rank chromosomes would be test error. However, since neural networks are being used, it would be especially useful to incorporate fitting uncertainty into the fitness function. Hence ranking was based upon the model committee error (as in equation 8.13):

$$FE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} {\sigma_{y_i}}^2 + \sum_{i-1}^{n} \frac{1}{n}(t - y_i)^2} \tag{9.7}$$

where n is the number of models in the committee that are used to make predictions on a single set of inputs (which give the lowest test error or highest LPE). The FE can define the fitness of each chromosome, so that a higher "score" gives a high ranking. However, the error needs to be inverted to guide the optimisation process correctly:

$$Score = \frac{1}{FE} \tag{9.8}$$

Genetic algorithms then use this information to speculate on new search points for the next generation. This is done using the three optimising operators; reproduction, crossover and mutation.

9.4 Test Simulations

A program was developed to use neural networks and GAs in tandem, using the attributes as described in this chapter. This allowed sets of inputs that lead to the same or improved strength to be identified.

9.4.1 Just a random search?

It was important to establish whether the optimisation process was more efficient than a simple random search. The YS model was used with all the variables from table 8.1 to examine this issue. Both simulations were set according to table 9.2 as follows:

Populations	4
Generations	100
Inputs allowed to vary	All variables
Boundary of generated random numbers	±1
Chromosomes per population	10
Crossover rate	Performed every 10 generations between all populations
Desired YS	0.4 (normalised); 310 MPa
Accuracy	To within 10%
Target score	25
Mutation rate	Nominal*

Table 9.2 Reference parameter settings

(* - Exact value is uncertain, but all variations of mutation rate are related to this nominal value)

The target score, T_s, is defined as:

$$T_s = \frac{1}{(\sqrt{T_n \times A})^2} \tag{9.9}$$

where T_n is the normalised target and A is the accuracy. The target score was set to 25 in this simulation, but the GA continued even if this was satisfied. This was because convergence would come at different times for each simulation. The aim was not to find the point of convergence, but rather the performance after a given number of generations.

For the random search, the best score was retained after each generation. However the GA search had its chromosomes ranked in the following way:

Chromosome number(s)	Action
0-3	Reproduced and kept at the top of the list
4-5	Swaps genes of chromosomes with others within a population
6-7	New individuals are created
8	Chromosome is mutated
9	Chromosome is used in crossover of genes across all populations

Table 9.3 Operations upon 10 chromosomes after ranking each generation

Note that the first chromosome is “0”, and the tenth is “9”.

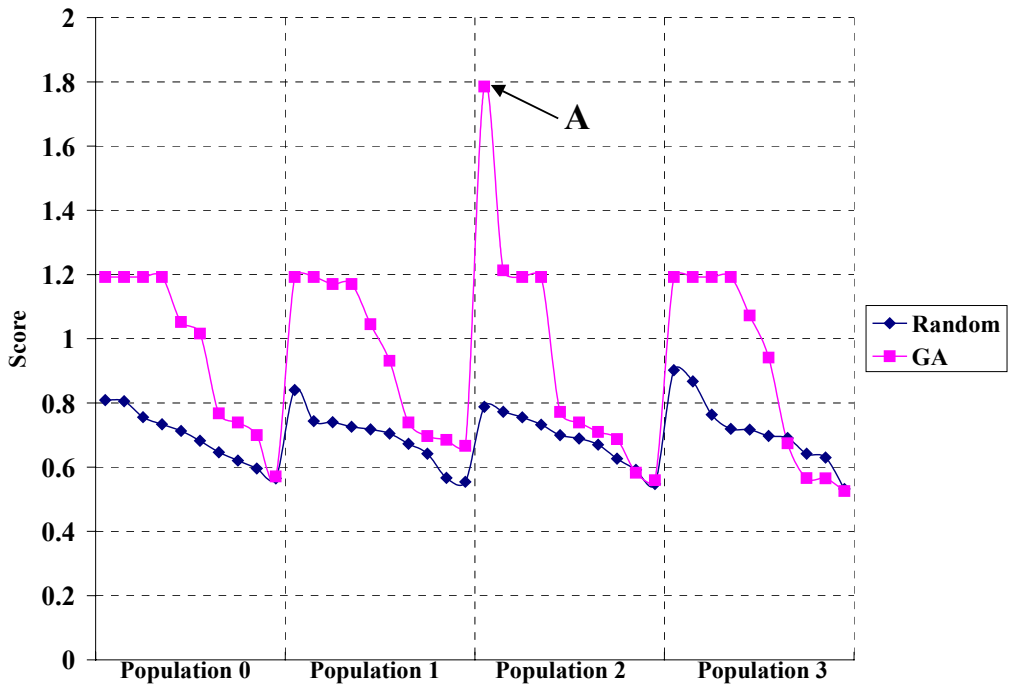

Figure 9.2 – Comparing a random search with the GA process, where each population is ranked in score order. Point "A" represents a chromosome with a unique high score in relation to the others.

The results in figure 9.2 show that the genetic algorithm is reaching higher fitness values than a random search. Note that four populations have been studied, the range over which the variables can vary is identical (table 9.2) for each population. Since all initial chromosomes are generated at random, the results in each population should not differ much, with the exception of point A (fig. 9.2), considering that each

population has only ten chromosomes. However the target score was 25 which means that the desired accuracy has not been achieved with either process.

In an attempt to increase the likelihood of reaching the target, the number of variables was reduced to only 2; chromium and nickel. The others were fixed to 0.47Mo-1.56Mn-0.62Si-0.01Nb-0.04Ti-0.17Cu-0.031N-0.062C-0.0007B-0.025P-0.013S-0.047Al wt%, whilst the ratio was calculated as 0.121. The heat treatment and test temperatures were 1400 K and 298 K respectively. The aim was to discover whether the number of variables was limiting the ability of the GA to converge.

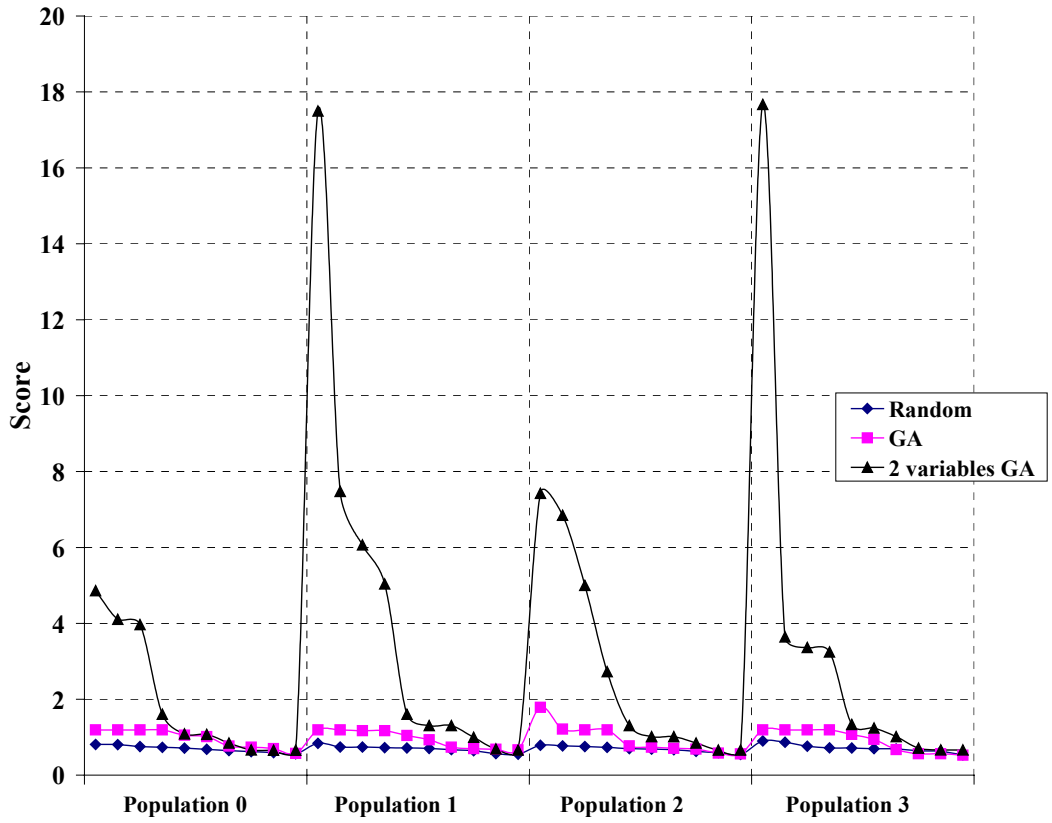

Figure 9.3 – Same as figure 9.2, but with an extra plot showing the effect of reducing the number of inputs that can vary to two.

From figure 9.3, it is clear that the number of variables does limit the speed of the process. However, it is first necessary to conduct test simulations to discover the effects of the different settings upon the performance of the GA.

The settings for the following tests are the same as in table 9.2. However, only chromium and nickel were allowed to vary, and the remaining compositions were set as mentioned above.

9.4.2 Generations

It is thought that as the search algorithm continues over time, the scores should tend to achieve, or exceed, the target. To investigate this, 50, 100 and 500 generations were tested, as shown in figure 9.4.

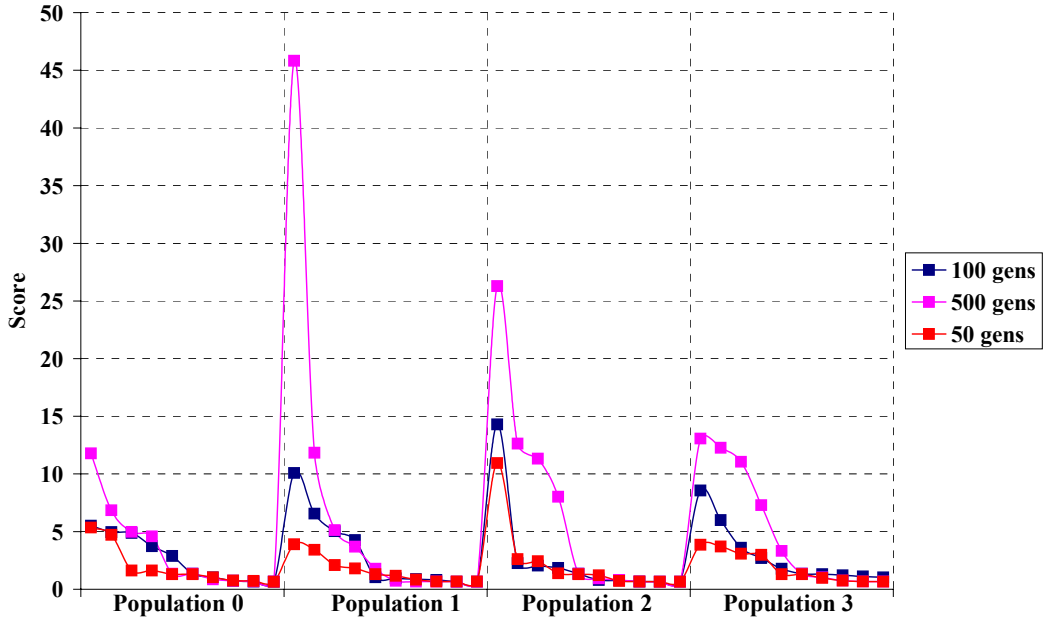

Figure 9.4 – Results of test simulation with 50, 100 and 500 generations. However, the benefits of running for 500 generations are exaggerated by the highly scored chromosome in population 1, thought to be a random case.

The premise that more generations equate to a higher score was confirmed, as the probability of covering the input space increases, at least until the highest possible score is achieved. This result reinforces the integrity of the GA process. However, it would be desirable to increase the speed of convergence on the target.

9.4.3 Populations

A multitude of populations, or a larger number of chromosomes in each population, is needed to survey the range of input space available. However, it cannot be so large that convergence is slow, so 2, 4, 8 and 20 populations were tested.

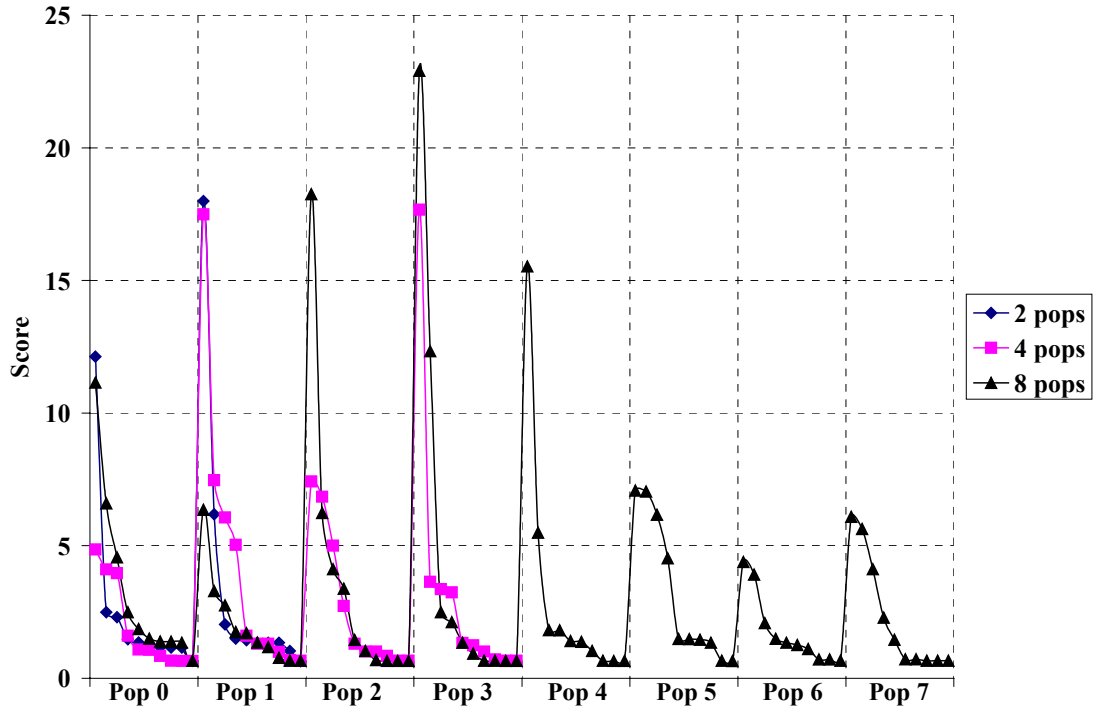

Figure 9.5 - Results of test simulation with 2, 4 and 8 populations

From figure 9.5, 8 populations performed the best in terms of producing relatively high scores in relation to the target score. The 20 population simulation did appear to achieve some high scores (not shown), but the majority of scores were relatively low. There is also a convergence speed problem, as the time to complete a generation also increases. From this, it was thought that enlarging the chromosome population might prove more fruitful.

Before this simulation was done, the settings in table 9.3 were also adjusted to cater for more chromosomes, table 9.4. If ignored, the final 10 chromosomes would be neglected of all but the crossover operator after each generation.

Chromosome number(s)	Action
0-7	Reproduced and kept at the top of the list
8-11	Swaps genes of chromosomes with others within a population
12-15	New individuals are created
16-17	Chromosome is mutated
18-19	Chromosome is used in crossover

Table 9.4 Revised operations upon 20 chromosomes after ranking each generation

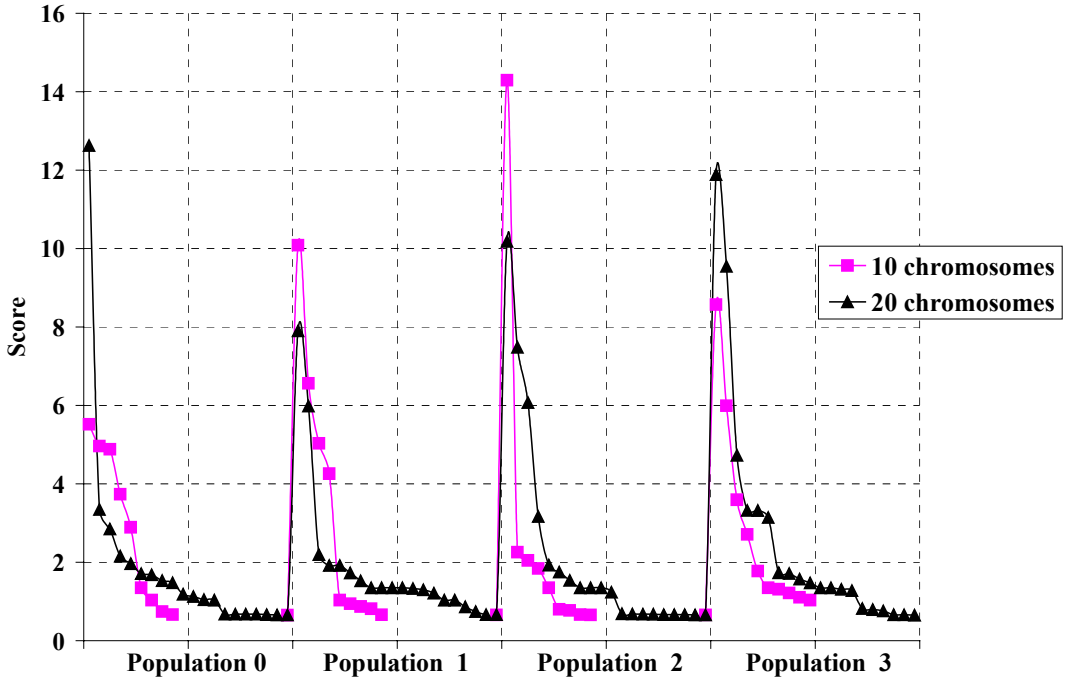

Figure 9.6 - Results of test simulation with 10 and 20 chromosomes. Note that the 10 chromosome population plot is not continuous, so that both models can fit on the same axis.

There was not much difference between the best performing chromosomes of each simulation, figure 9.6. However, increasing the number of chromosomes also increases the number of search points within input space, hence 20 chromosomes are seen to be more useful.

9.4.4 Crossover rate

From table 9.2, it is seen that crossovers occur after every 10 generations. This was adjusted to different settings to see the effect on the target score; 1, 5 and 20 were chosen to investigate.

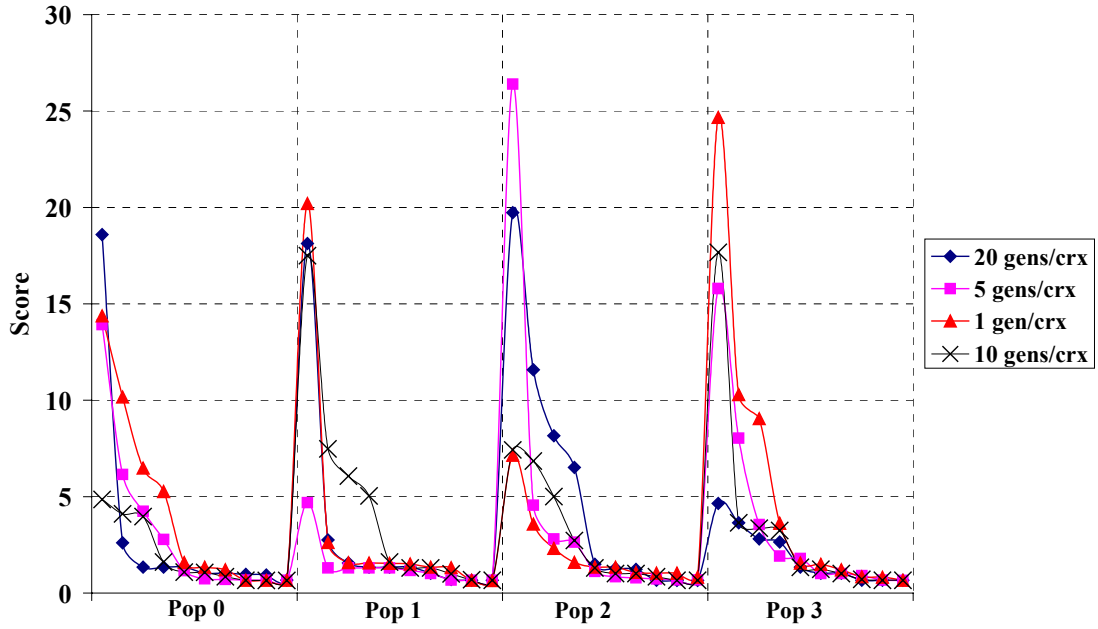

Figure 9.7 - Results of test simulation with 1, 5, 10 and 20 crossovers per generation

In general, crossovers after each generation gave the best performance. It was consistently amongst the highest scores in figure 9.7, compared to other simulations. The 20 generation model was also quite good. However, it was thought that the simulation that maximises the sharing of information between the population should be encouraged.

9.4.5 Mutations

This parameter was required to ensure that crucial information was not lost whilst advancing through the generations. Moreover, it is a safeguard against remaining within secondary minima. Nevertheless, it should not be too large so as to invoke a random search.

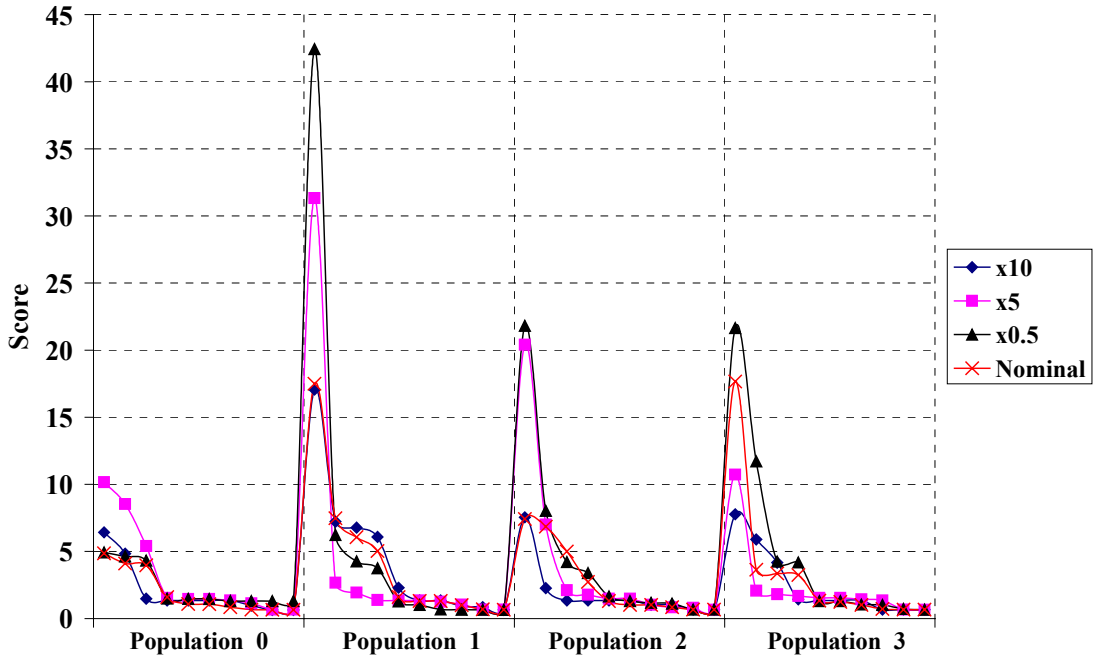

Figure 9.8 - Results of test simulation with 0.5x, 1.0x, 5.0x and 10.0x mutation rates

As seen in figure 9.8, the nominal value from the parameter settings (table 9.2) actually performed quite poorly, hindering the optimisation process. In fact, a mutation rate set to half of the nominal appeared to be the best performer.

9.4.6 Random number boundaries

Random values are used to supply gene values for every chromosome to initialise the GA process. This is also used to generate new chromosomes during optimisation. As shown in table 9.2, there is a boundary as to where these values can come from, hence ±0.7, ±3 and ±5 were examined.

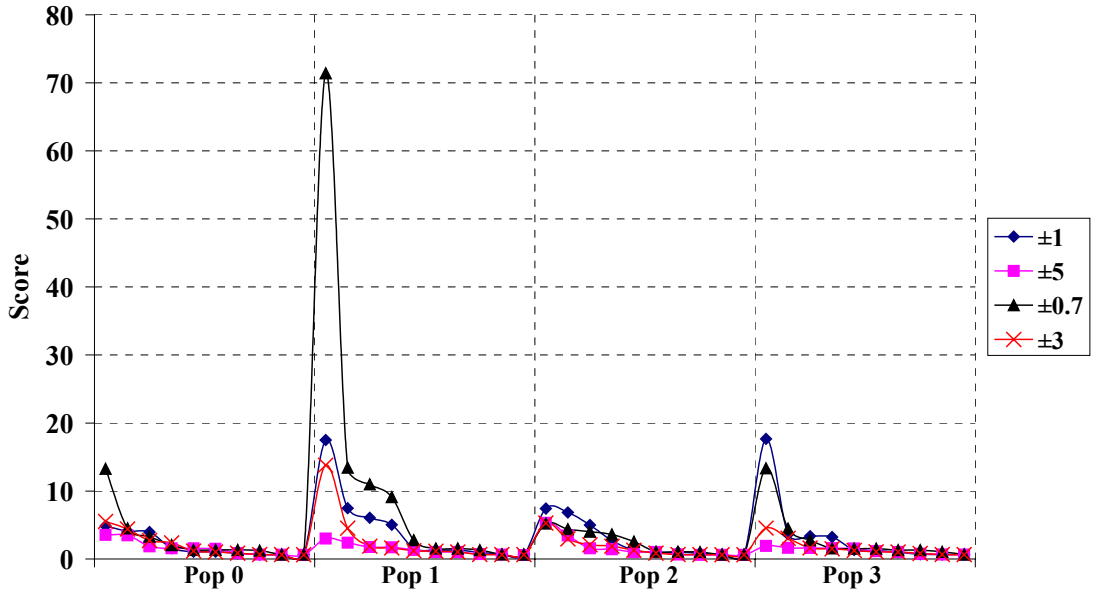

Figure 9.9 – Results of test simulation with ±0.7, ±1.0, ±3 and ±5 boundaries for random number generation

From figure 9.9, ±0.7 performed the best. However, this was probably because the error bars would get larger as the range increases beyond the neural network database. However, a wider domain is desired to search for a variety of solutions, hence ±5 may be desirable. Again, the convergence speed must be accounted for before selection, so a boundary of ±3 was taken as a compromise. Nevertheless, in the event that gene values became erroneous in future simulations, the boundary would be narrowed further to counter this effect.

9.4.7 Updated Settings

A final test simulation was conducted to compare the reference settings and the newly suggested settings, shown in table 9.5.

Populations	8
Generations	100
Inputs allowed to vary	Cr, Ni only
Boundary of generated random numbers	±3
Chromosomes per population	20
Crossover rate	Performed after every generation between all populations
Desired YS	0.4 (normalised); 310 MPa
Accuracy	To within 10%
Target score	25
Mutation rate	50% of nominal reference setting

Table 9.5 Revised parameter settings

Figure 9.10 shows that the new settings did not perform as well as the original after 100 generations. However, a larger range of input space is being searched from the initial random number generation. If this is taken into account, it is understood that convergence will be slower but for a purpose.

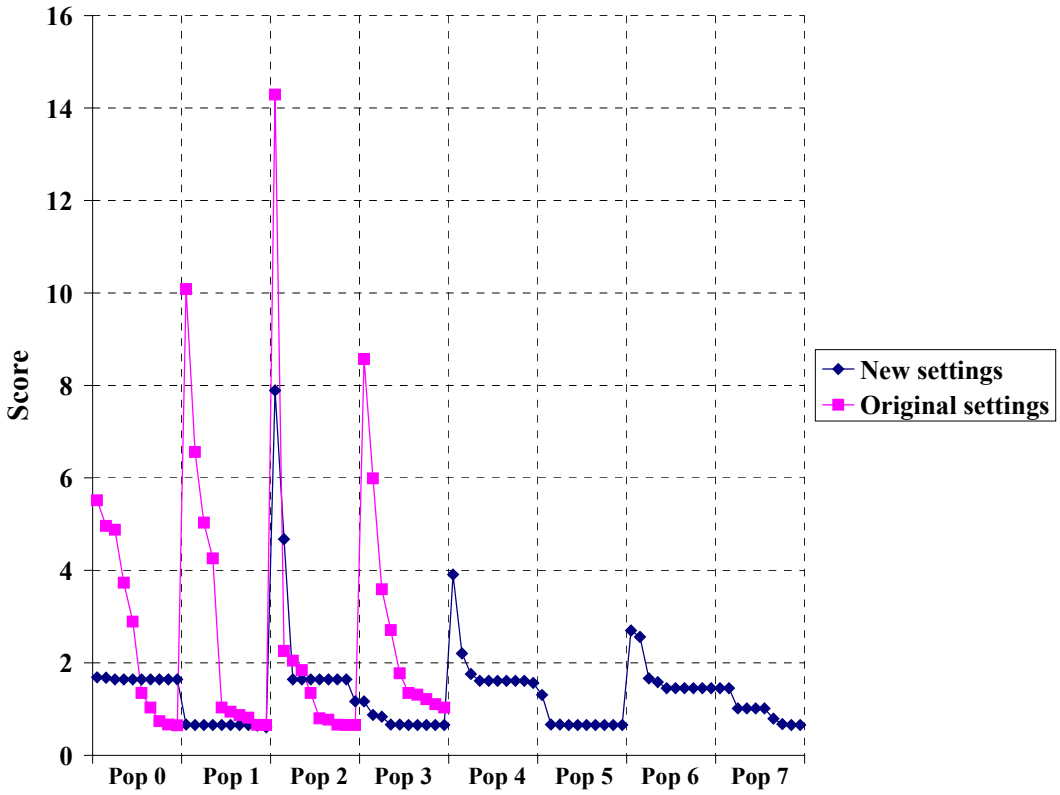

Figure 9.10 – Comparison of the original and new settings

9.4.8 Analysis summary

This analysis was useful as a guide for the best settings for optimisation. However, it is still difficult to ascertain whether convergence will be faster. For example, a high score below the target score after 100 generations may be performing well, but there is no guarantee that it will advance faster than a gene with an inferior score at that moment in time. Nevertheless, the examination highlighted that adjustments to parameter settings can have a marked difference on performance.

9.5 Predictions

Based on the settings suggested in section 9.4.7, predictions were made for a target outside the neural network database for temperatures 298 and 773 K, as shown in table 9.6. The only major differences with the previous section were increases in the number of input variables allowed to vary and the number of generations. The heat treatment temperature was fixed at 1400 K.

Populations	8
Generations	1000
Inputs allowed to vary	Cr, Ni, Mo, Mn, Si, B, C , N, ratio
Boundary of generated random numbers	±3
Chromosomes per population	20
Crossover rate	Performed after every generation between all populations
Desired YS	1.0 (normalised); 495 MPa
Accuracy	To within 10%
Target score	10
Mutation rate	50% of nominal reference setting

Table 9.6 Parameter settings for predictions

9.5.1 Reproduced chromosomes

A total of 5 simulations were run for both test temperatures, and an average score was taken. As previously stated, the highly scored chromosomes are reproduced and ranked at the top of the next generation. This ensures that the score does not degrade at any time during optimisation. In this case, 8 chromosomes are reproduced, but as seen in figure 9.11a, they all have the same score. This was a concern, as vital information is not seen to be well-distributed. To investigate this, identical simulations were run, but only 2 chromosomes were allowed to reproduce per generation. Hence the remainder were left to exchange information.

In figures 9.11a and 9.11b, it is clear that there is not much difference in top scoring chromosomes in either case. However with only 2 chromosomes reproduced, the majority are involved in sharing genes, hence increasing the likelihood of reaching the target score specified.

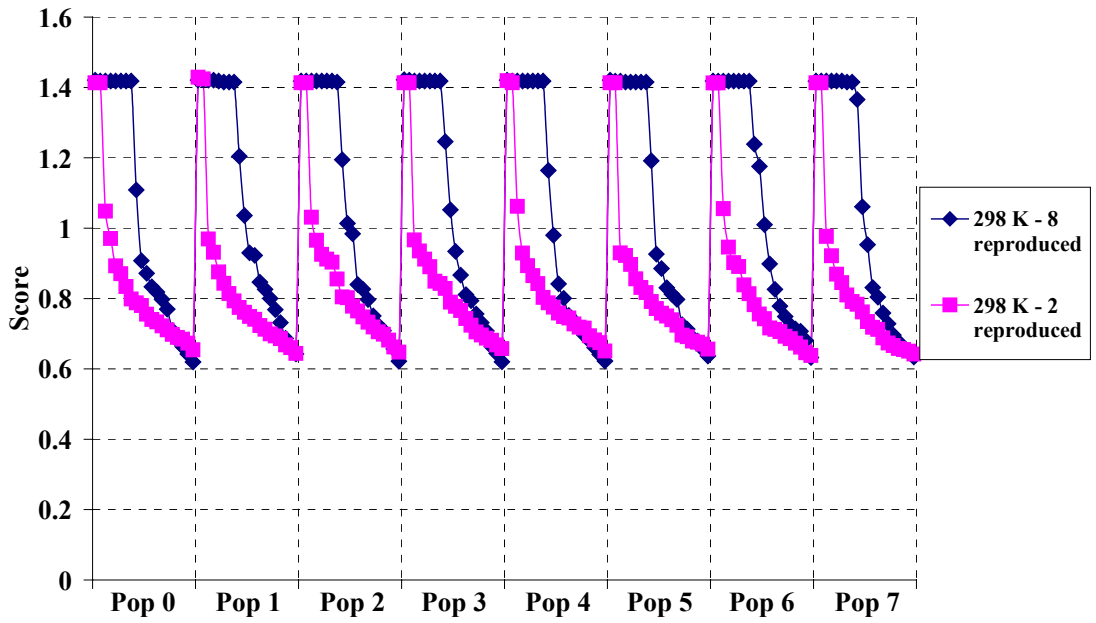

Figure 9.11a - Comparison of scores for varying chromosome reproduction at 298 K

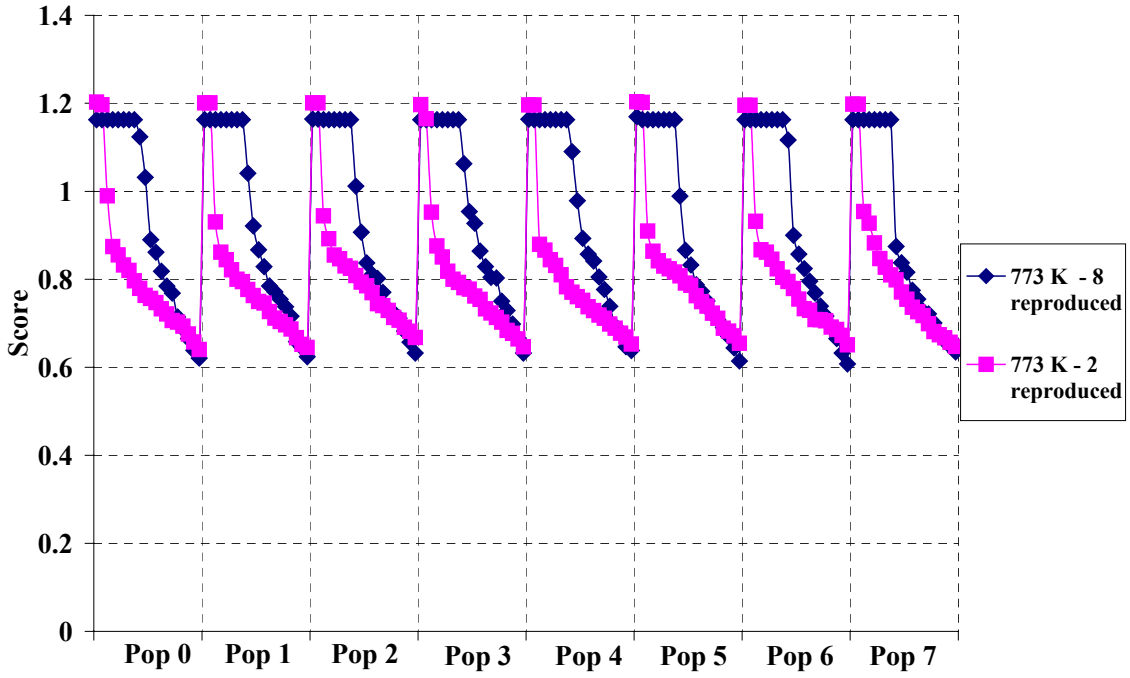

Figure 9.11b - Comparison of scores for varying chromosome reproduction at 773 K

9.5.2 Revised settings

The parameter settings are as defined in table 9.6, except for the random number boundary, which was limited to ±1 to speed up convergence. The number of generations was also increased to 2000 to allow the GA more time to find solutions.

Note that optimal searches were only conducted for yield strength. This was because the UTS neural network model committee was quite complex; hence the GA search would be very slow.

A total of three simulations were run for test temperatures 298 K and 773 K. This was done to achieve a range of compositions, which should begin from different areas of input space.

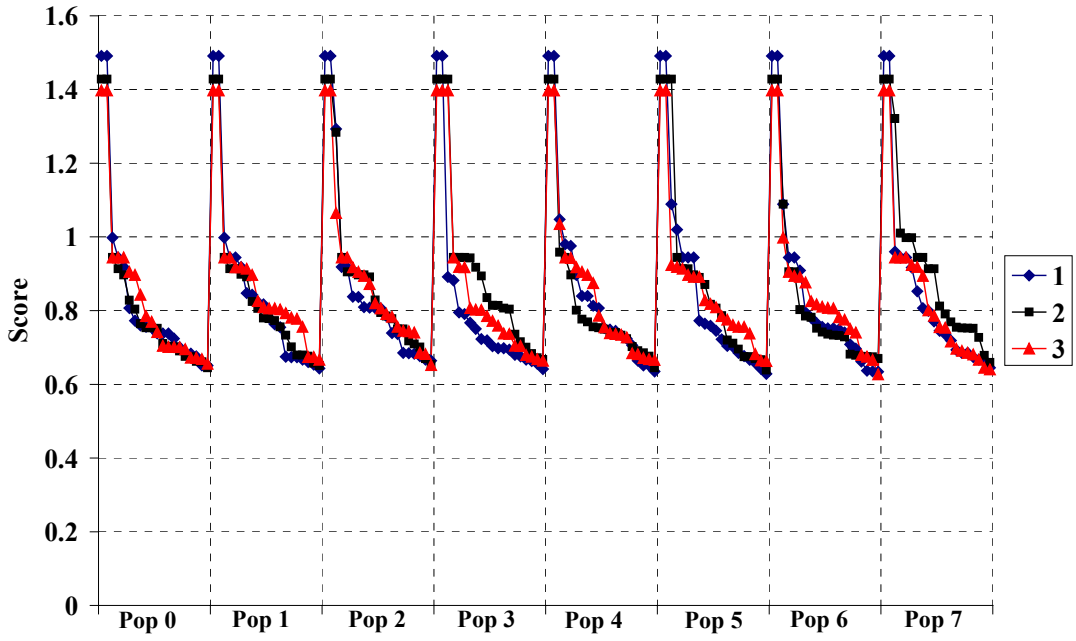

Figure 9.12a - Scores for simulation 1, 2 and 3 at 298 K

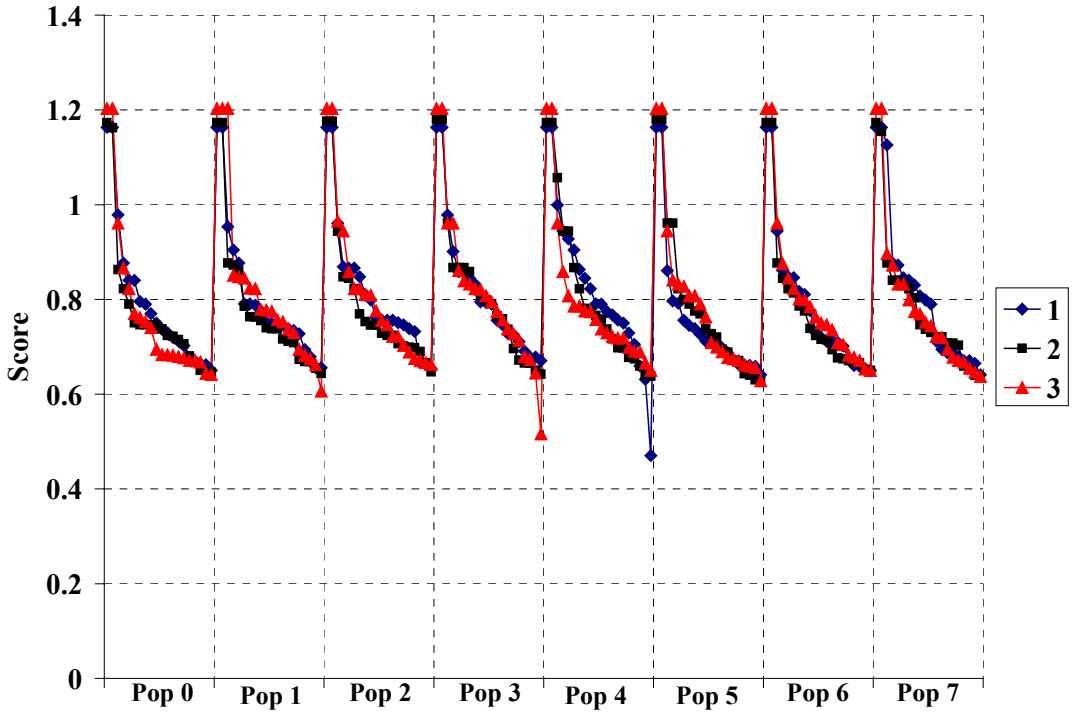

Figure 9.12b - Scores for simulation 1, 2 and 3 at 773 K

The target score was set to 10, but the maximum value achieved after the GA process was around 1.5 for both temperatures (figs. 9.12a and 9.12b). Nevertheless, analysis of the fittest chromosomes (top 2 of each population) revealed some interesting results, as shown in table 9.7.

In the 298 K simulation, the best chromosomes reached approximately 400 MPa, but the associated errors were quite high. However this was expected, as they had output values greater than the maximum YS reported in the neural network database. Moreover, the error bars are not so large as to be considered unrealistic and all the listed chromosomes listed appear to have sensible gene values.

Cr / wt%	Ni / wt%	Mo / wt%	Mn / wt%	Si / wt%	N / wt%	C / wt%	B / wt%	Ratio	Result / MPa	Error / MPa
18.689	9.799	1.737	2.425	0.575	0.0405	0.103	0.005	0.238	396.440	171.352
18.480	12.142	4.365	2.425	0.575	0.0405	0.091	0.004	1.054	422.055	178.843
17.434	12.378	4.365	2.425	0.575	0.0405	0.068	0.006	0.703	394.063	168.524

Table 9.7 - Best compositions from GA after 2000 generations for 298 K for Fe-0.01Nb-0.04Ti-0.17Cu-0.025P-0.013S steel

Cr / wt%	Ni / wt%	Mo / wt%	Mn / wt%	Si / wt%	N / wt%	C / wt%	B / wt%	Ratio	Result / MPa	Error / MPa
18.673	11.446	1.761	2.425	0.575	0.064	0.107	0.002	1.531	399.559	302.067
20.488	18.286	4.365	2.069	0.245	0.014	0.124	0.010	0.425	363.839	216.094
18.673	11.446	1.761	2.425	0.575	0.064	0.107	0.002	1.319	387.744	288.738
18.673	11.446	1.455	2.425	0.575	0.064	0.107	0.002	1.319	385.426	293.789
20.488	18.286	4.365	2.425	0.245	0.014	0.124	0.010	0.425	364.222	219.021
18.102	10.123	4.365	2.425	0.575	0.064	0.128	0.005	0.261	381.350	218.256

Table 9.8 - Best compositions from GA after 2000 generations for 773 K for Fe-0.01Nb-0.04Ti-0.17Cu-0.025P-0.013S steel

(error in tables 9.7 and 9.8 includes $\pm 1\sigma$ and fitting uncertainty)

The search conditions were quite different for 773 K. The maximum YS reported in the database for steel at 773 K was 265 MPa, hence requesting 495 MPa would be quite a challenge. Nevertheless, the GA process still predicted quite well, as with 298 K, managing to predict around 400 MPa (table 9.8). However, the associated errors were larger in this case.

The predictions made here do require further investigation. Batch production in lab conditions may be a possibility to see how well the GA has predicted. However, this is beyond the scope of this project.

To improve upon these findings, an increased number of populations, chromosomes and generations could be used to help the GA achieve its objective. The convergence speed may be slower, but time is relative in any situation.

9.6 Summary

A program incorporating the theory of GAs and neural networks was developed and a new fitness function was devised for optimisation. The aim was to search for sets of inputs that led to the same or improved strength. The software for optimising tensile properties can be obtained from the web:
http://www.msm.cam.ac.uk/map/map.html

Initially, analysis showed that the GA process was superior to a random search. Different parameter settings were explored to aid optimisation and help speed up convergence. For example, increasing the number of chromosomes and populations helped the GA by occupying more of the input space. Allowing the crossover of information between populations after each generation also allowed more data to be available to the system.

Predictions were made using these optimal parameters to find a target YS at different temperatures. However, the desired target was deliberately set outside the original neural network database, to see how the GA would cope with such uncertainty. Results showed that reasonable compositions were suggested, even though the target YS was not met. Error bars associated with predictions were understandably large, but not excessive.

The next step would be to conduct a number of repeats to obtain a range of compositions, and for longer periods, to see whether set targets could be met.

In general, GAs are not intended to directly translate theoretical compositions of a given strength into reality. However, the aim of this method is to narrow the focus of research into new and existing areas which have not been previously explored.