NAME: MICHAEL BLACK

STUDENT NUMBER: 221402063

WRCV ASSIGNMENT 4:

TRAINING OF A FEEDFORWARD NEURAL NETWORK USING EVOLUTIONARY ALGORITHMS

Contents

GENETIC ALGORITHM:

The performance of the genetic algorithm was analysed by varying the parameters namely: population size, mutation rate, and mutation magnitude. The SSE over iterations of the best individual per generation was used as a metric to judge how well the algorithm was performing. The performance parameters that are of interest is convergence time, accuracy, and speed of the algorithm.

Test parameters:

Mutation Rate Test			
Fixed parameters	mutation magnitude = 5	iterations = 1000	population = 300
Mutation Rate	Training Accuracy	Test Accuracy	Final SSE
0.005	82.75	82.55	1671
0.01	94.123	93.56	901
0.015	79.59	79.23	1953
0.02	94	92.86	798
0.04	95	94.27	739
0.1	95	94.92	763
0.15	99.56	99.39	155
0.2	85.93	85.4	1520

Mutation Magnitude Test			
Fixed Parameters	mutation rate = 0.15	iterations = 1000	population = 300
Mutation Magnitude	Training Accuracy	Test Accuracy	Final SSE
1	64.3	62.5	3732
2	89	89	1199
5	95.21	94.8	643
7	95	94.44	785
10	99.56	99.39	155

Population Test			
Fixed Parameters	mutation rate = 0.15	iterations = 1000	mutation magnitude = 5
Population	Training Accuracy	Test Accuracy	Final SSE
100	84.22	83.11	1708
200	95.33	94.64	715
300	99.56	99.39	155

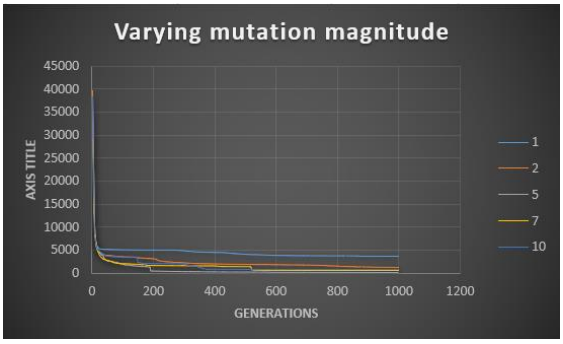

Figure 2 Mutation Magnitude

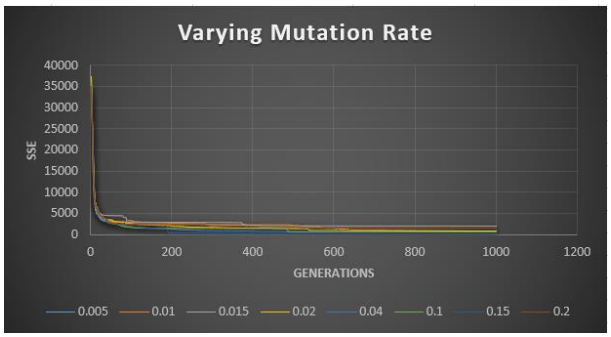

Figure 1 Mutation Rate

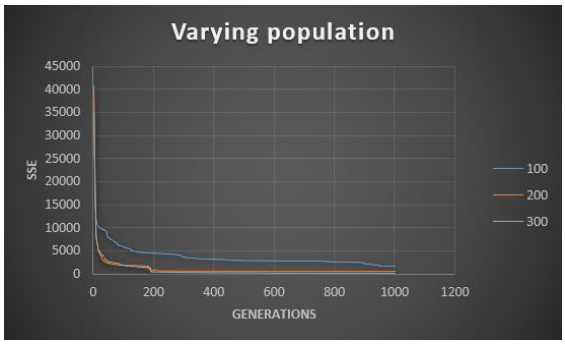

Figure 3 Population

Observations:

The algorithm used rank-based selection, elitism, and uniform crossover. This combination had the better performance when compared to roulette wheel selection with two-point crossover. Rank based selection and elitism allowed only the best individuals to process to the next generation and to mate. A value of 10% for elitism and 50% for rank-based selection was found to yield a sufficient accuracy of almost 99%. When a lower percentage of the fittest population was used, the population lost diversity and proceeded to find solutions in non-optimal local minima.

The convergence rate of the algorithm was mainly influenced by the size of the population, with higher population values causing quicker convergence. The accuracy of the algorithm also improved with an increased population, but it cost more computing power and is slower. Starting at a low mutation rate of 5% and increasing to 20% showed the effect of this parameter quite clearly, with the solution favoring higher mutation rates. Higher mutation rates allowed the training cycle to converge later and hence it kept learning even after many iterations. An increased mutation magnitude also allowed the algorithm to make more meaningful changes with the higher mutation rates, as can be seen with the larger dips in the Figure 1. Even with high values of mutation magnitude the algorithm still provided a high accuracy of 95%. The algorithm was however the second slowest to process.

Best Results	
Population	300
Mutation Magnitude	5
Mutation Rate	0.15
Training Accuracy	99.56
Test Accuracy	99.39
Best SSE	155

Genetic Algorithm Bean Classifications:

1 : DERMASON	11 : BARBUNYA	21 : BARBUNYA	31 : BARBUNYA	41 : HOROZ
2 : SIRA	12 : CALI	22 : HOROZ	32 : CALI	42 : HOROZ
3 : BOMBAY	13 : DERMASON	23 : CALI	33 : SIRA	43 : BOMBAY
4 : BOMBAY	14 : SIRA	24 : SIRA	34 : HOROZ	44 : BOMBAY
5 : HOROZ	15 : BARBUNYA	25 : SIRA	35 : SIRA	45 : CALI
6 : HOROZ	16 : BARBUNYA	26 : BOMBAY	36 : HOROZ	46 : DERMASON
7 : HOROZ	17 : SIRA	27 : BOMBAY	37 : CALI	47 : CALI
8 : BARBUNYA	18 : SEKER	28 : SIRA	38 : BOMBAY	48 : SIRA
9 : SIRA	19 : BARBUNYA	29 : BOMBAY	39 : SIRA	49 : CALI
10 : DERMASON	20 : SEKER	30 : BOMBAY	40 : SIRA	50 : SIRA

Figure 4 Genetic Algorithm Classifications

DIFFERENTIAL EVOLUTION

A Differential algorithm using binomial crossover was implemented for evaluation. The population, Scaling factor, and recombination probability was of interest when analyzing the performance of the differential evolution algorithm. The SSE of the best individual was used as a metric to determine how well the algorithm was learning over time. The same performance parameters namely, convergence, accuracy, and speed were observed.

Test Parameters:

Population Test			
Fixed parameters	F = 1	Cr = 0.1	
Population	Training Accuracy	Test Accuracy	Final SSE
100	70	72	3358
200	75.8	75.65	3448
300	67.05	66.87	3433
300	80.31	78.95	2798

F Test			
Fixed parameters	Population=300	Cr = 0.1	
F	Training Accuracy	Test Accuracy	Final SSE
2	70.51	69.61	3642
5	74.27	74.11	3649
10	72.87	71.44	3623

C Test			
Fixed parameters	Population=300	F=1	
Cr	Training Accuracy	Test Accuracy	Final SSE
0.2	65.55	64.83	4282
0.5	92.66	91.92	5141
0.7	62.92	61.65	4272

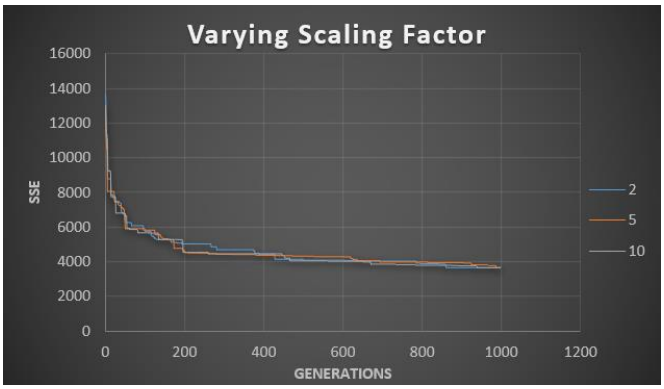

Figure 5 Scaling Factor

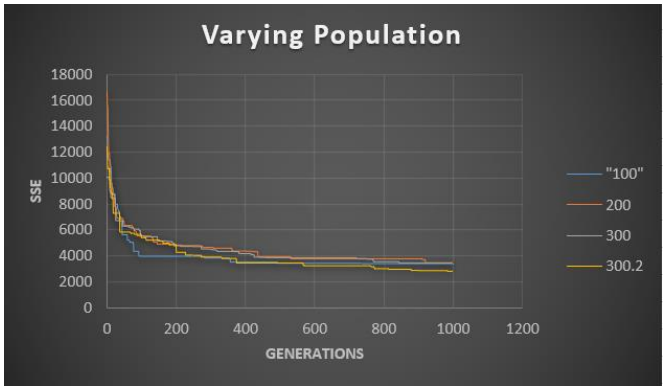

Figure 6 Population

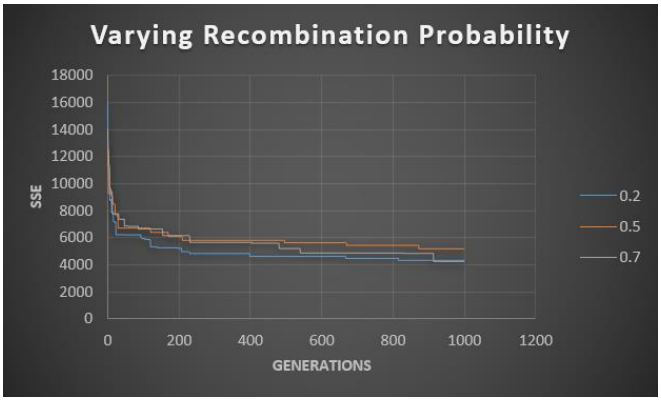

Figure 7 Recombination

Observations:

The differential evolution algorithm was the slowest, requiring a high amount of processing power for each iteration. The algorithm also learns in steps as can be clearly seen in Figure 6. This means that the algorithm will hang on a certain value of fitness for a long time before improving. The algorithm will keep improving but will require many hours of running to produce a 99% accurate result. The Scaling factor did not have much impact on the accuracy in the range that it was tested in. Recombination and population had the largest impact on the performance of the algorithm. Results were inconsistent as can be seen with in the population test table. For the same parameters the accuracy went from 80 to 67 percent, this was largely due to the initialization of the individuals as it was different each time, and the space between solutions has an impact on the algorithm. The SSE also does not correlate well with the accuracy of the algorithm as can be seen in the C Test table with a high SSE of 5141 giving a high accuracy of 93 percent.

When doing addition iterative testing, the solution did seem to favor lower scaling factors and higher recombination probabilities, however there is still no clear correlation between these values and the accuracy since the results are not highly reproducible.

Optimization				
Fixed Parameters	population = 300	Iterations = 1000		
F	Cr	Training Accuracy	Test Accuracy	Best SSE
0.5	0.7	86.5	86	2517
0.1	0.5	68.92	67.8	2826
0.05	0.9	78.34	78	2797
0.1	0.2	69.85	68.97	3669

Best Results	
Population	300
Scaling Factor	0.5
Recombination	0.7
Training Accuracy	86.5
Test Accuracy	86
Best SSE	2517

DIFFERENTIAL EVOLUTION BEAN PREDICTIONS:

1 : BOMBAY	11 : BARBUNYA	21 : BARBUNYA	31 : BOMBAY	41 : BOMBAY
2 : SIRA	12 : CALI	22 : BOMBAY	32 : CALI	42 : BOMBAY
3 : BARBUNYA	13 : BOMBAY	23 : CALI	33 : SIRA	43 : BARBUNYA
4 : BOMBAY	14 : SIRA	24 : SIRA	34 : BOMBAY	44 : BOMBAY
5 : BARBUNYA	15 : BARBUNYA	25 : SIRA	35 : SIRA	45 : CALI
6 : BARBUNYA	16 : BOMBAY	26 : BOMBAY	36 : BOMBAY	46 : BOMBAY
7 : BOMBAY	17 : SIRA	27 : BOMBAY	37 : CALI	47 : CALI
8 : BARBUNYA	18 : BOMBAY	28 : SIRA	38 : BOMBAY	48 : SIRA
9 : SIRA	19 : BARBUNYA	29 : BOMBAY	39 : SIRA	49 : CALI
10 : BOMBAY	20 : BOMBAY	30 : BOMBAY	40 : SIRA	50 : SIRA

Figure 8 Differential Evolution Classifications

PARTICLE SWARM OPTIMIZATION

A global best particle swarm algorithm was implemented with an added inertia weight to increase the accuracy of the algorithm. The SSE was of the fittest individual was used to observe the performance of this algorithm. The same performance parameters namely, convergence, accuracy, and speed were observed. The variable parameters that were tested consisted of the following, particle amount, inertia weight, cognitive component, and the social component. The purpose of testing was to determine the effects of exploration and exploitation by setting either the cognitive or social component to 0 and observe the outcome.

Test Parameters:

Particle Variation			
Fixed Parameters	omega = 0.4	cognitive = 1	social = 1
Particles	Training Accuracy	Test Accuracy	Best SSE
10	11.47	11.6	8385
100	14.44	14.37	8227
200	49.65	47.38	6289
250	30.93	28.72	7683
300	46.04	45.88	6936
300	21.76	21.26	7618

Inertia			
Fixed Parameters			
Omega	Training Accuracy	Test Accuracy	Best SSE
0.4	30.6	30.02	3739
0.9	55.94	54.4	4780
0.9	42.51	42.03	5931
1.5	9.27	8.6	6234

Local Attraction			
Fixed Parameters	particles = 200	social = 0	omega = 0.9
Cognitive	Training Accuracy	Test Accuracy	Best SSE
1	28.74	27.45	7502
1.5	30.45	28.88	7476

Global Attraction			
Fixed Parameters	particles = 200	cognitive = 0	omega = 0.9
Social	Training Accuracy	Test Accuracy	Best SSE
1	17.82	14.48	31055
1.5	18.17	15.23	13759

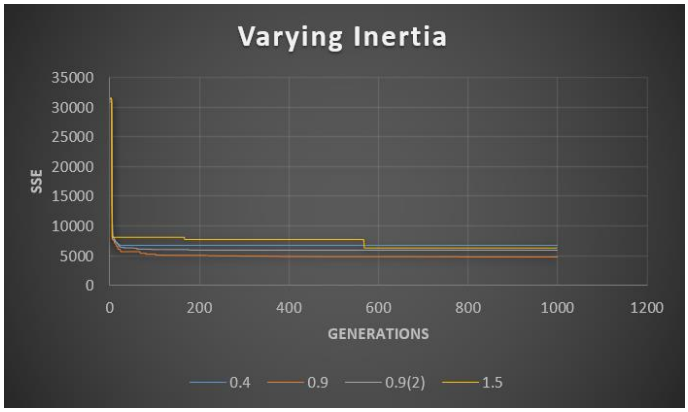

Figure 10 Inertia

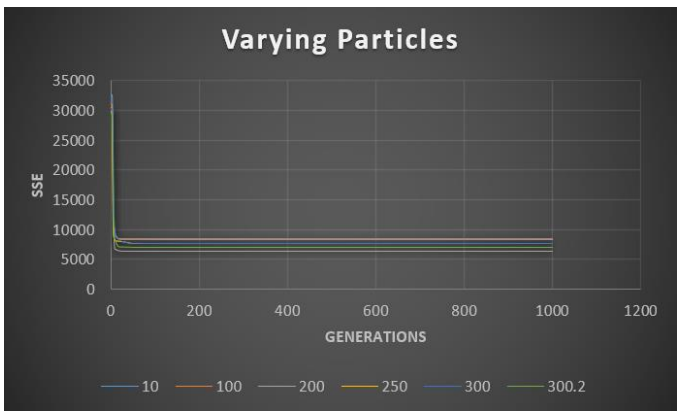

Figure 9 Particles

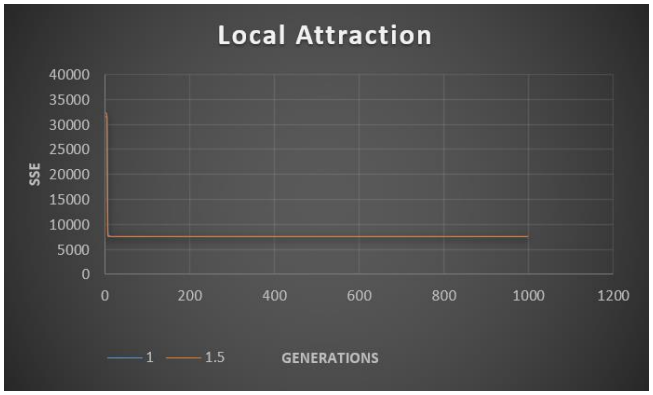

Figure 11 Local

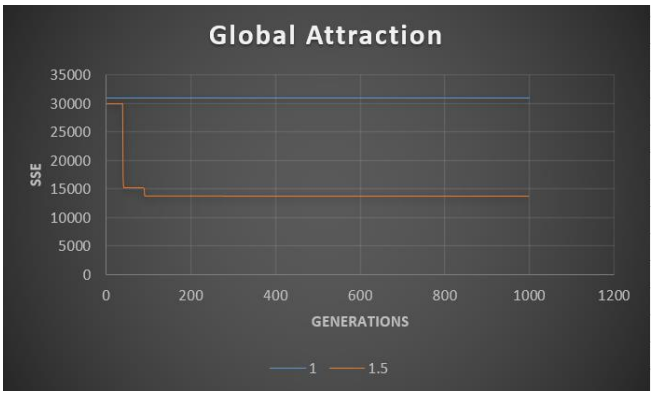

Figure 12 Global

Observations:

The particle swarm optimization algorithm had the quickest convergence rate but also the lowest best accuracy when compared to the other algorithm. When observing the number of particles, 200 – 300 seemed yield the best solutions, although this did vary when test was repeated. Values below 100 particles did not yield any results. The algorithm does not need to be run for 1000 iterations since it converges quickly and almost never improves after even as little as 100 iterations at time. A high inertia component did seem to help training further than 100 iterations when looking at Figure 8. The algorithm became slow when values for social and cognitive were above 2. Per iteration it was the slowest of all the algorithm with high social and cognitive values but required less iterations to converge. A higher social component was favored when finding solutions as can be seen in Figure 9.

Results were inconclusive since no consistent results were found for specific parameters. The results varied at random, but after iterative testing the most appropriate values were found that yielded a fair accuracy of between 60% to 70% consistently.

Best Results	
Particles	200
Inertia	0.5
Cognitive	1.5
Social	1.9
Training Accuracy	72.5
Test Accuracy	70

Particle Swarm Optimization Bean Predictions:

1 : SEKER	11 : BOMBAY	21 : SIRA	31 : BOMBAY	41 : HOROZ
2 : SIRA	12 : CALI	22 : HOROZ	32 : CALI	42 : HOROZ
3 : SIRA	13 : HOROZ	23 : CALI	33 : SIRA	43 : BOMBAY
4 : BOMBAY	14 : SIRA	24 : SIRA	34 : HOROZ	44 : BOMBAY
5 : HOROZ	15 : BOMBAY	25 : SIRA	35 : SIRA	45 : CALI
6 : HOROZ	16 : SIRA	26 : BOMBAY	36 : HOROZ	46 : SEKER
7 : HOROZ	17 : SIRA	27 : BOMBAY	37 : CALI	47 : CALI
8 : CALI	18 : HOROZ	28 : SIRA	38 : BOMBAY	48 : SIRA
9 : SIRA	19 : SIRA	29 : BOMBAY	39 : SIRA	49 : CALI
10 : HOROZ	20 : HOROZ	30 : BOMBAY	40 : SIRA	50 : SIRA

Figure 13 Particle Swarm Optimization Classifications

Conclusion:

The genetic algorithm yielded the highest accuracy as well as the highest speed per iteration. The particle swarm optimization algorithm has the fastest convergence rate but was the least accurate of the algorithms. The differential evolution algorithm was the slowest algorithm, but it was the second most accurate algorithm.

The results could be improved by collecting more data of more different values and repeating in order to find the deviations of the solutions. The accuracy could also have been improved by using a single unaltered set of weights instead of randomly generating each time the program runs.