The program created and used during this investigation is used to demonstrate the influence of parameter changes and operator methods on the ability of a Genetic Algorithm (GA) to work towards the global optima of a search space, it is an evolutionary algorithm inspired by Darwin’s natural selection built around the principles of selection and reproduction, with the initial population randomly generated.

The GA unless stated otherwise takes a population (P) of 200 individuals with a chromosome length (N) of 20, encoded as floats ran for 200 generations, with results averaged over two complete runs allowing the most consistent parameter combinations to be uninfluenced by outlying results. Results will be rounded to 3 decimal places for clarity and the only stopping criteria is fixed generations reached.

The program initially evaluates the fitness score of all individuals before performing selection on the population, with only superior individuals being chosen. The newly formed population is then subject to crossover, producing a new offspring population. The offspring are then altered with the mutation operator, as crossover continually recombines two of the best solutions in the population there is a risk the lack of diversity may result in convergence at a local minima, the introduction of ‘random’ values into the chromosome allows individuals the chance to escape the local minima across the search space.

**Styblinski-Tang**

The Styblinski-Tang function has a global minimum of -39.16599\*N (Surjanovic & Bingham, 2013)

resulting in an approximate minimum of -783.3198 where N=20 and each value within N (x) = between -5 and 5. Following population generation and evaluation the program runs a wide parameter sweep of both chance of mutation (mutrate) and maximum size of mutation (mutstep), mutrate ranges from 0 to 0.1 (0.1 = 10% chance) in 0.005 increments and mutstep ranges from 0 to 2 in 0.1 increments, all combinations of mutrate and mutstep within the ranges are tested, sorted by fitness and recorded, with the top five performers separated and displayed as shown in figure 1.

Table

Description automatically generated

*Figure 1 – Top 5 performing combinations with the best fitness achieved and the average fitness in P.*

Tournament selection is used to improve the populations average fitness initially, with two random individuals being drawn from the population P times and having fitness’s compared, with the greater being passed forward into the offspring. This is a simple selection method that can be scaled to a larger tournament where required, in this instance the small number of ‘competitors’ allows genetic diversity to be maintained. If two below average individuals are selected, regardless of their relative weakness to the population one shall be passed along allowing for potentially useful parts of the chromosome to be utilised due to crossover. Other selection methods are widely used such as Roulette Wheel Selection however due to the fitness function handling both positive and negative values it would not be suitable for use, in other minimization functions it could be used if data was normalised to allow for correct proportionalities to be applied (Abd Rahman et al., 2016).

Crossover mimics mating in nature, combining two individuals (chosen as promising during selection) to create offspring that hopefully inherit the best parts of both parents to create a superior individual. Using single-point crossover the best achieved fitness score was -783.146, here a random point is chosen between 0 and N-1 to be used as the crosspoint, all values before the crosspoint in an individual are swapped with the corresponding value from the next member of the population. Multi-point crossover is very similar, however an additional crosspoint is selected between the initial crosspoint and N-1, all values beyond the new crosspoint are once again switched, this can take place any number of times, multi-point here achieved -781.673.

When ran with both single-point crossover and multi-point crossover for the full parameter sweep, whilst results are similar amongst the best achieving solutions it can be observed that especially with a higher population size a better individual is likely to be found with multi-point crossover, it is possible this is due to the higher likelihood an individual’s values will be altered during crossover, increasing the probability weaker individuals will inherit stronger genes from the other parent. On initial sweep single-point crossover achieves only nine individuals with a score lower than -775, multi-point achieves 26, likewise the average population fitness is generally improved by an average of 12 points across the top 30 performers.

Offspring are then mutated to introduce genetic variation, the effect mutrate has on the GA’s ability to search the space and obtain the best solutions is evident when tweaked to extremes. Setting mutrate to 1% results in early convergence due to the lack of genetic variation resulting in the population becoming trapped in local optimums. Setting mutrate to 50% however mutates so regularly that promising solutions are destroyed instead of being developed, with the average population fitness being pseudo-random akin to random search (see figure 2). Multiple runs of wide parameter sweeps suggest that a mutrate of up to 10% provides the best balance.

Chart

Description automatically generated

*Figure 2 – High mutrate inhibits population from settling, hence causing an unpredictable average.*

The program also implements elitism, once mutation is complete the best individual is sought out as well as the worst, the worst individual is then replaced with a duplicate of the best individual eliminating a relatively ‘bad’ combination of genes with a strong performer, whilst this reduces diversity slightly, it increases the chance that another poor individual will combine with a strong individual to create something useful.

The program then runs the five best mutrate/mutstep combinations discovered through the GA and plots the results to a graph (shown in Figure 3), the non-deterministic nature means that results vary, but from experimentation it is rare for outlying results to be produced by the selected combinations.

Graphical user interface

Description automatically generated

*Figure 3 – Best performers graphed across a GA run.*

Figure 4 shows the distribution of gene values in random individuals scattered across the graph, as opposed to the black squares representing the best found individual tightly centred around the global minima at -2.903534 (Surjanovic & Bingham, 2013). Figure 5 shows three runs of the best combination, with best fitness, average population fitness and the average of both figures, the graph shows all runs moving in a consistent fashion with scores coming together at approximately 50 generations.

Chart, scatter chart

Description automatically generated

*Figure 4 – Scatter showing random individuals vs best performer avoiding the local optimum.*

Graphical user interface

Description automatically generated

*Figure 5 – 3 runs of the best combination plotted.*

To investigate the effects of population size and number of generations the program runs the best combination with populations of 200-400 and generation numbers of 100-500. Unsurprisingly the higher the number of both the stronger the results typically, with the highest combination achieving -783.22 with single-point crossover. The greater the population size the higher the probability of finding a good individual at initiation and there are more opportunities for a single individual to mutate into a good solution. More generations ran also allows more chances at a beneficial mutation, likewise selection and crossover take place more times allowing the population to consistently fine tune. Figure 6 below shows the results of the generation/population experiment on the single-point crossover GA.

*Table

Description automatically generated*

*Figure 6 – Generation and Population experiment.*

**Dixon-Price Function**

This section of the report investigates the Dixon-Price test function and will focus on the results achieved with less explanation of the programs functionality, the only changes unless stated are the test function itself, elitism, single point crossover and tournament selection are used unless stated otherwise. The test function is a minimisation function with a global optimum fitness score of zero (Surjanovic & Bingham, 2013).

Running the wide parameter sweep produced scores ranging from 1.019 through to 34,616.730. Of the top five performers in figure 7 three have a mutrate of 3% and two have 2% suggesting this range is desirable. Mutstep is more varied, however the value for four of the top achievers sits between 0.8-1, it is possible that due to slight value changes creating exponential fitness changes, mutrate should be relatively low to allow crossover to hone solutions, opposed to mutation jumping across the search space. However, the 19 worst performers of the sweep all have a mutstep of 0.1 or 0.2 regardless of mutrate, suggesting that when mutation does take place a low mutstep has a negative impact.

Table

Description automatically generated

*Figure 7 – Top performers.*

Upon rerunning the above combinations, index 1 was found to be the best achieving 0.884 and was chosen as the combination to investigate. Figure 8 shows 3 further runs, narrowly missing 0 on only 200 generations, whereas poor mutrate/mutstep combinations achieve scores in the 30,000 range.

Graphical user interface

Description automatically generated

*Figure 8 – Multiple runs of best combination.*

Single point cross over appears to create better individuals, this may be due to the exponential change in fitness score with only small changes to an individual’s values, meaning that extra disruption to the chromosome causes more harm than good once suitable individuals have been found, multi-point crossover achieves 1.860 as the best score in the wide sweep but fails to achieve a sub-one score even when ran for P=400 and 500 generations.

When population and generations are increased on the single-point individual however 0 is achieved multiple times, unsurprisingly the higher both P and runs the better outcome. When looking at a population of 200 for example in figure 9 results consistently improve with an increased run size. When looking at all runs of 200 with varying population sizes however there is less consistent improvement with population size, suggesting that for this problem generations ran is paramount to the population size.

Table

Description automatically generated

*Figure 9 – Generation and population experimentation.*