CS3910 Bachelor's Report in Computer Science

Experimental Study on Novel Intelligent Matrix Optimisation Methodology

School of Engineering & Applied Sciences, Aston University, Birmingham

Author

Matthew Ahearn

Word Count

1,142[1]

Pages

6[1]

November 2021

[1]Excluding front page, tables and captions, hypotheses, references and appendix.

1. Introduction

The problem proposed by this report is the optimisation of an n length array of double values – the lower the resulting value, the more accurate the optimisation. The novel solution was designed around an evolutionary algorithm, with additional local searches performed on each child array as it is generated - "It is commonly experienced that the combination of an evolutionary and a heuristic model performs better than either of its 'parent' algorithms alone"[1]. An assumption is made that there is access to an evaluation function.

2. Proposal of Solution

2.1. Baseline

The baseline solution is a simple evolutionary algorithm utilising One Point Crossover and Swap Operators on a generational model.

2.2. Novel Solution

2.2.1. Explanation

The novel solution is a modification layered atop the baseline. As such it is a duplicate, with each child having a local search applied to it after recombination and mutation. Two local searches will be used - Random Optimisation and Golden-Section Searches. The local search applied to each child will be randomly selected via weighted value.
The Golden-Section Search, a mutation of the Fibonacci Search[2], considers each node by adding an increasingly small float to the node value and submitting the replacement value if any optimisation is made – exploring the local area. The Random Optimisation selects random replacements for each node and submits any which provide an optimisation. The first time an optimisation is made, the search is closed.

2.2.2. Justification of Novel Solution

By applying a local search algorithm to each child array after it has undergone recombination and mutation, we can explore each node of the array for the global minima while avoiding getting stuck in a local minimum. The Random Optimisation gives the generation an opportunity to find its way out from any accidental local minima to either a better local minimum, or the global minima, which will prove mostly useful in the initial phases of the algorithm as we attempt to find the global minima to optimise around but will always provide a chance to escape a local minimum in later generations. We choose Random Optimisation over a gradient based heuristic optimisation method "Because the gradient information indicates only the 'localized' feature of the objective function, rather than the global optimum, only the local optimum can be found"[3]. Conversely, the Golden-Section Search homes in on a local minimum specifically, drastically cutting down the results to achieve the local minima. This Local Search method will come into play more in the later phases of the generations.

3. Experimental Methodology

3.1. Experimental Procedure

Each algorithm will be run 30 times. Each algorithm will have a population and initial size of 50, run for 10 generations, and have a mutation chance of 0.7. The test data set will be used.

As the data is not expected to achieve normality, the One Sample Z Test will be used for statistical analysis.

All raw data collected in this study is available in the appendix.

3.1.1. Optima Discovery Testing

We will examine the differences between the best evaluation results of both methods. The best evaluation found will be output for each run, and the given data examined. The hypotheses of this test follow:

H0: On average, the novel solution is no better than the baseline solution at optimising arrays in terms of accuracy of estimation.
H1: On average, the novel solution is better than the baseline solution at optimising arrays in terms of accuracy of estimation.

3.1.2. Generational Improvement Testing

We will examine the number of iterations required to reach a minima. We will examine the rate of percentage change of evaluations between generations from each algorithm and be able to deem a local or global minima found once the percentage change between generations has plateaued - for this study, fallen below 1%. To examine this in greater detail, this test will change the number of generations parameter to 30. The hypotheses of this test follow:

H0: On average, the novel solution requires a greater or equal number of generations before the percentage increase in optimisation between generations is less than 1%.
H1: On average, the novel solution requires a lesser number of generations before the percentage increase in optimisation between generations is less than 1%.

3.1.3. Time Testing

A final test will be run to compare the time each algorithm takes to complete one run. This test will use milliseconds as the metric, as the number of evaluation iterations used should be static in both algorithms and would display nothing of interest. As such, this testing will be judging specifically the implementation of both methods used in this study. The hypotheses of this test follow:

H0: On average, the novel solution will take more time to perform the optimisation task on the given problem.
H1: On average, the novel solution will take less time to perform the optimisation task on the given problem.

3.2. Experimental Results

3.2.1. Optima Discovery Testing

The results from each test give a mean value of 7.6043 for the baseline solution, and 0.16335 for the novel. This establishes that the hypothesis H1 has merit.

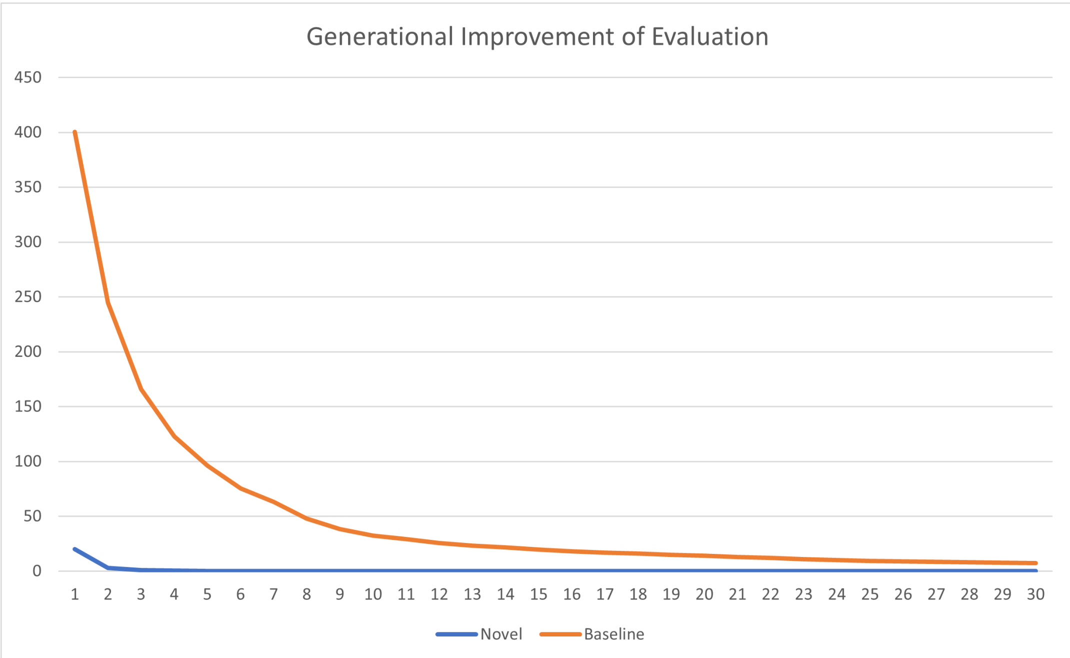

Figure 1: Graph charting the best evaluation produced by each generation, for all 30 generations, for the novel solution in blue, against the baseline solution in orange. The X axis displays the generation number and the Y axis displays the current best evaluation result.

As the data collected fails the Central Limit Theorem, we must use the One Sample Z Test.

Performed on the data we get a result of t=4.08, while the t-value on the table[4] at 49 Degrees of Freedom and p=0.05 is 1.671. As our calculated t-value is larger than the table value, we can reject the null hypothesis.

3.2.2. Generational Improvement Testing

The percentage change between generation optimisation is shown in the following figures:

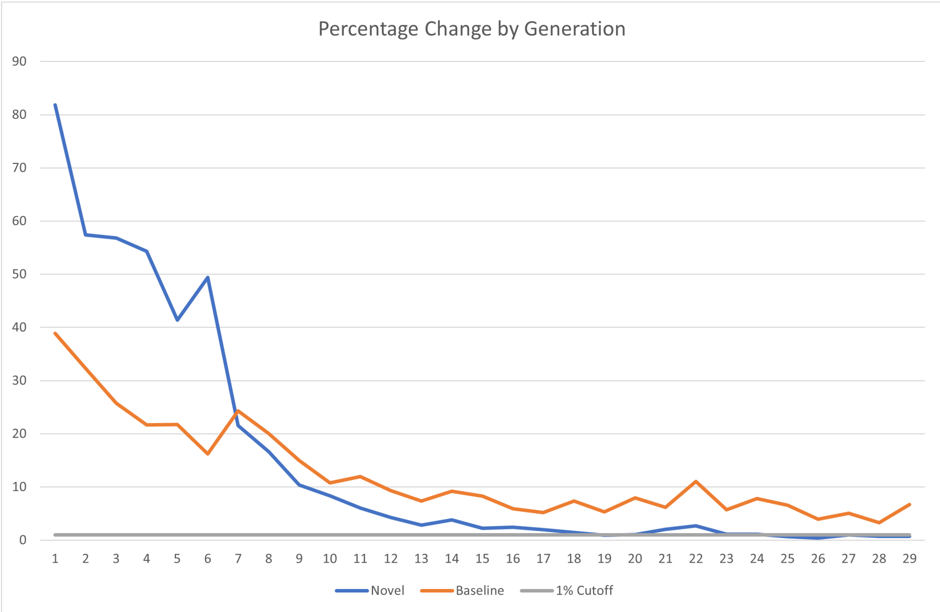

Figure 2: Graph charting percentage change between each generation, for all 30 generations, for the novel solution in blue, against the baseline solution in orange. The 1% Cutoff line, in grey, represents the 1% mark below which we consider the algorithms to have settled in their minima positions. The X axis displays the generation number and the Y axis displays the percentage change from the last optimal value.

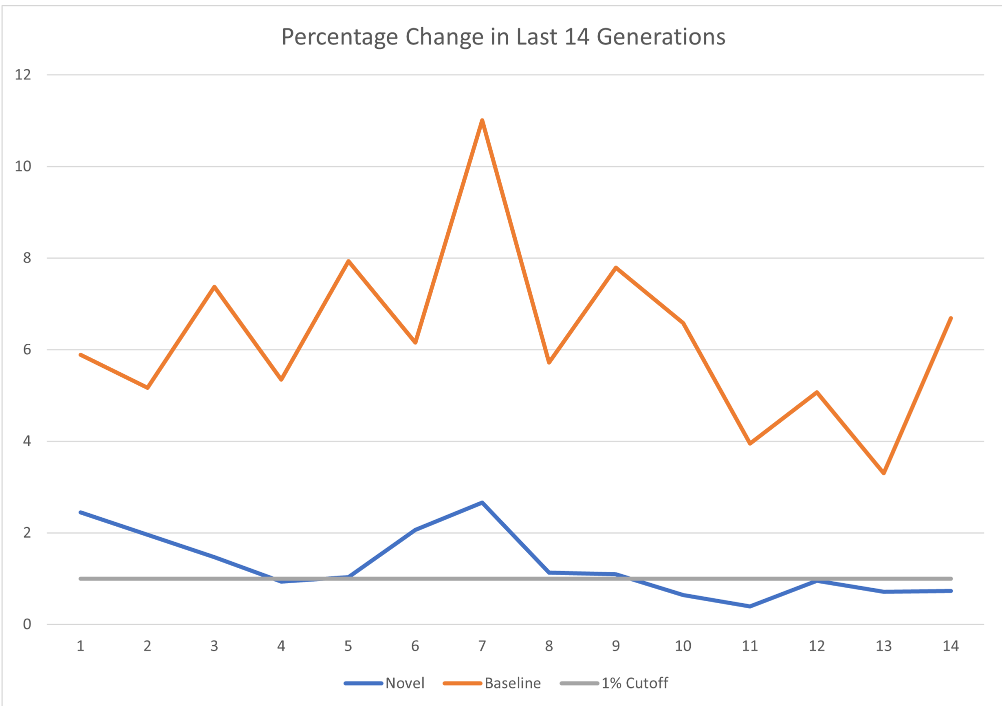

Figure 3: As the previous chart could not display enough detail, fig.3 displays only the last 14 generations, where the percentage change falls below the 1% guideline. The novel solution is plotted in blue, while the baseline solution is plotted in orange. The 1% Cutoff line is visible in grey. The X axis displays the generation number and the Y axis displays the percentage change from the last optimal value.

The data used for this analysis was already proven statistically significant in the previous test, and so no analysis will be performed here - the trend is already proven.

The baseline solution fails to reach the 1% change mark, and therefore cannot be determined to have settled within the allocated population size. However, as shown in a higher degree of detail in fig3, the novel solution succeeds in falling below a 1% change on generation 21, with a percentage change of 0.986. Given these results, the null hypothesis is rejected.

3.2.3. Time Testing

The baseline takes a mean time of 84.47ms, and the novel solution with a mean time taken of 3560ms. The two closest points of data were 109 for the baseline, and 3694 for the novel, showing the data sets at no point gave close results. As such, H1 is rejected.

4. Analysis of Experimental Results

4.1. Interpretation & Comparison of Results

By looking at the rejected hypotheses, we can conclude two points: the novel solution consistently achieves superior optimisations and does so in fewer generations - but lacks time efficiency.

The mean results showed the novel solution had a 97.85% reduction in evaluation results. This gain displays a clear victory over the baseline.
In addition, the novel solution achieved its first plateau of optimisation on the 21st generation, with any further optimisations providing an insignificant (less than 1%) bonus. On the other hand, the baseline solution failed to plateau at any point within the 30 generations allocated - and showed no trend towards a plateau based off the collated data.

However, the novel solution performed 4,015% slower than the baseline. This time is directly tied to the implementation of the solution created for this study, meaning that more efficient implementations could cut down on this time. The tests performed on generational improvement show a more abstract view of number of necessary evaluations – the novel requiring only 21 generations versus the baseline's ¿30.

4.2. Justification of Results

Given the preceding interpretation, the novel solution cannot be justified as the objectively superior solution. Instead, where quality of optimisation is of paramount importance, the novel solution provides a strong - 97.85% reduction - improvement. However, in cases where speed is necessary, the results support the baseline solution as a winner. With further testing, it is expected that a partial implementation of the novel solution may provide a high quality of optimisation without the huge decrease in time performance. Additionally, it is important to note again that the time measurements are specific to the implementation in this study.

Therefore, while the novel solution provides strong results where time is of no concern, additional time must be given to optimising the solution and creating a partial implementation whereby the increase in time requirements can be minimised, while still providing improvements to the optimisation.

References

[1] Smith J. E. Agoston E. "Introduction to Evolutionary Computing". In: Springer, 2015. Chap. 10.

[2] Mordecai Avriel and Douglass J. Wilde. "Optimality proof for the symmetric Fibonacci search technique". In: *The Fibonacci Quarterly. Official Organ of the Fibonacci Association* 4 (1966), pp. 265–269. ISSN: 0015-0517.

[3] Junyi Li and R. Russell Rhinehart. "Heuristic random optimization". In: *Computers & Chemical Engineering* 22.3 (1998), pp. 427–444. ISSN: 0098-1354. DOI: https://doi.org/10.1016/S0098-1354(97)00005-7. URL: https://www.sciencedirect.com/science/article/pii/S0098135497000057.

[4] Statistics How To. *T-Distribution Table (One Tail and Two-Tails)*. URL: https://www.statisticshowto.com/tables/t-distribution-table/. (accessed: 13.12.2021).

Appendices

Data displaying best evaluation results across 30 runs of the baseline algorithm:
https://drive.google.com/file/d/1DSOxbOHw1kuaRr5vWozvgkNV1XOZZ0vI/view?usp=sharing

Data displaying time taken for each run, across 30 runs of the baseline algorithm:
https://drive.google.com/file/d/1YNritVkVJp2qs3TL5lccdhS2OlkNqStu/view?usp=sharing

Data displaying best evaluation results across 30 runs of the novel algorithm:
https://drive.google.com/file/d/1Qsj0SNo4LVpj7Y3wxUIVh6aeVG2tukNx/view?usp=sharing

Data displaying time taken for each run, across 30 runs of the baseline algorithm:
https://drive.google.com/file/d/1_lxtZHV7SpezQ3WyMfNUUkIMZsbVVy6x/view?usp=sharing