CSE 848 Project Report
Extending Applicability of Groundwater Flow Algorithm through Algorithmic Equivalence

Submitted by: Ritam Guha (MSU ID: guharita)

Date: March 26, 2021

1 Abstract

According to *No Free Lunch* theorem, there exists no algorithm which works better than all the other algorithms over all possible problem domains. However, the notion of this theorem breaks down when only a specific problem domain is considered. This means there exists some algorithms or some operators which may work great for specific sets of problems. Algorithmic Equivalence makes outstanding use of this interpretation. The main idea behind algorithmic equivalence is to import operators suitable for a problem domain to an algorithm to make it applicable to the problem domain. This report explores the effectiveness of algorithmic equivalence using Groundwater Flow Algorithm (GWFA) as the underlying optimization algorithm over various unimodal and multimodal problems. From the results and corresponding discussion, it can be observed how algorithmic equivalence can become a powerful tool for extending the applicability of GWFA.

2 Introduction

Algorithmic Equivalence can be defined as the process of finding *connections* between two different algorithms and representing them in a common framework. The purpose of finding this common framework is that if two algorithms are found equivalent, it is possible to borrow useful operators from one algorithm to the other to improve their individual performances. Inpsired from the success achieved in [1] to improve the performance of Particle Swarm Optimization (PSO), it was interesting to extend the concept of algorithmic equivalence for other algorithms in varying problem settings. For this purpose, a recently proposed meta-heuristic algorithm named Groundwater Flow Algorithm (GWFA) [2] has been selected as the underlying optimization algorithm. The goal of the project is to improve the performance of GWFA over multiple Unimodal and Multimodal problems using algorithmic equivalence with different algorithms.

In statistics, Unimodal problems are defined as problems where objective functions are having a single peak in their distributions. On the other hand, Multimodal Problems may have several peaks in their objective distributions. The goals for these two types of problem settings are often considered to be different. In most of the case, the goal of unimodal problems is to get to the best solution in the objective space as fast as possible. On the other hand, in 1987, Goldberg et. al formulated the goal of multimodal problems differently from unimodal problems. In [3], they argued that if the goal of multimodal problems still remain to find the best solution, the algorithms may run into some trouble when small initial population may allow some sampling error. Then the algorithms may start over estimating some schemata and finally they may reach the wrong optima. For this reason, the goal of the multimodal problems was further shifted to find all the different peaks at once. The later literature [4] have followed this definition and kept conceiving multimodal problems using this objective.

From the previous discussion, it is clear that the same algorithm cannot work efficiently for both types of problems because different procedures are needed to satisfy these different (somewhat contradicting) goals or objectives. For this reason, these two problems are suitable for testing out the power of algorithmic equivalence. There are several advantages of doing algorithmic equivlence in different problem settings.

- First of all, different programmers may be comfortable using separate algorithms. According to the *No Free Lunch* theorem, there is no algorithm which is better than all the other algorithms for the entire set of problems. So, if programmers want to extend the applicability of the preferred algorithm over different problem domains where the algorithm does not work well, algorithmic equivalence is a great way to do that. The goal then becomes to make it algorithmically equivalent to another algorithm which works well for the destination problem, if possible.

- Algorithmic Equivalence is a way to discover strengths and weaknesses of a particular algorithm with the help of some other algorithms in different problem settings.

In this project, GWFA is used as the index algorithm which is modified with the help of algorithmic equivalence over the two sets of problems. GWFA is a nature-inspired optimization algorithm proposed in 2020 based on the flow of groundwater from one place to another. At the final stage of water cycle, i.e. precipitation, the water precipitated in the form of hail, rain, snow etc. recharges the water beds present undergorund. These areas are known as Recharge Areas (RAs) while the places where groundwater gets exposed to the surface of the earth (like streams, rivers etc.) are known as Discharge Areas (DAs). The flow of groundwater from RAs to DAs is known as groundwater flow and French mathematician Henry Darcy was the first person to mathematically describe this flow. GWFA uses this concept for optimization by considering the best solutions as DAs and the other solutions as RAs where the goal for the RAs is to move towards the DAs (the better solutions found by the algorithm) with an expectation to improve their own solutions and explore the space in the process.

A careful observation at the GWFA algorithm will illustrate the fact that some of its features are similar to Evolutionary Algorithms (EAs) in many aspects. For example, the position update procedure for any RA in GWFA involves three candidate solutions (One DA, local average of all the better RAs and the RA under consideration). Although the equation for combining these three candidate solutions involve terms like hydraulic gradient and velocities, it is ultimately a way of recombination which is a very important operator in the domain of EA. Some other concepts like Mutation, Parent-Child Comparison are also present in GWFA which makes it a great candidate to perform algorithmic equivalence with EAs.

The rest of the report is divided into 2 sections. Section 3 describes the project in detail explaining the problems, the gradual progress with algorithmic equivalence, experimental settings and metrics along with the experimental outcomes and detailed analysis. Finally, the project is concluded in the 4^{th} section.

3 Project Description

In this section, the overall structure of the project, the goals and the gradual progress has been described in detail. The final objective of the project is to explore the prowess of algortihmic equivalence using GWFA as the underlying test algorithm and applying it over unimodal and multimodal problems. In order to successfully perform GWFA's algorithmic equivalence with other EAs, the first process is to represent GWFA in a common framework as other EAs. After careful observation of GWFA's inner operations, it has been observed that GWFA can be represented as the sketch shown in Figure 1. The following points can be considered as a short summary for the Figure:

- The population is sorted according to the fitness scores and the best solutions are called DAs, while the worse solutions are regarded as RAs.
- In generation t, an RA is updated using a recombination operator among three different sets of parents: the RA itself, a collection of fitter RAs and a randomly selected DA. After recombination, the child solution is preserved for the next generation and is not used in the same generation anymore. In standard GWFA, the collection of the fitter RAs is represented by the average of the fitter RAs. So, a combination of 3 parents recombine to produce a child solution corresponding to an RA.
- The DAs are only mutated and if the mutated solution is better than the current DA, it is preserved for the next generation, else the current DA is passed on to the next generation unchanged.
- The Intermediate generation is the collection of the updated solutions which are not yet sorted according to the fitness scores. After sorting this popula-

tion according to the fitness scores, it provides the population for generation $t+1$.

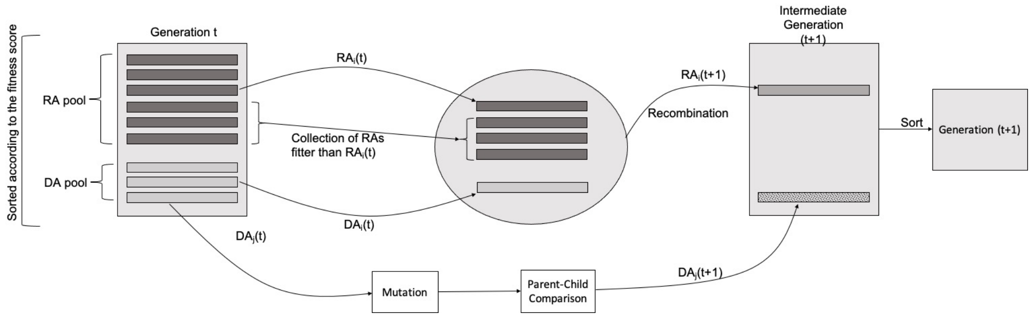

Figure 1: Representation of GWFA in an Evolutionary Algorithm Framework

In the next part, the two types of problems are discussed one after the other.

3.1 Unimodal Problems

As a part of this project, GWFA and its variants are applied over three popular unimodal problems: Axis-Parallel Hyper-Ellipsoid Function (F_{elp}), Rotated Hyper-Ellipsoid Function (F_{sch}) and Rosenbrock's function (F_{ros}). The three function descriptions are provided in Table 1.

Function Alias	Function Expression
F_{elp}	$f(x) = \sum_{i=1}^{n} i x_i^2$
F_{sch}	$f(x) = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2$
F_{ros}	$f(x) = \sum_{i=1}^{n-1} 100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2$

Table 1: Mathematical Expressions of the three unimodal functions

In 2002, Dr. Deb proposed G3-PCX [5], short for Generalized Generation Gap based Genetic Algorithm with Parent Centric Crossover. It is an algorithm which is specialized for unimodal problems. The goal of this algorithmic equivalence is to check how GWFA's performance over these problems can be improved by importing different operators from G3-PCX. As it works great over these problems, it can be assumed that the operators in G3-PCX are well-suited for unimodal problems and they should improve GWFA's performance. Please note that the objective of this project is not to discuss the superiority of one algorithm over the other. The target is to improve GWFA's performance and in the process, learn how one algorithm's inner operations can help another algorithm to grow.

3.1.1 Experimental Setting

The goal for the unimodal problems is to make the algorithm converge as fast as possible for the three functions. The convergence is defined as finiding a solution having functional value within ≤ 0.1 (S_1 criterion) or $\leq 10^{-20}$ (S_2 criterion) radius of the global optimum value. Every variant of GWFA is run 50 times. If all the runs converge, the variant is tested for quality based on the number of function evaluations it required. If some of the runs did not converge, then the number of times it converged is mentioned in brackets. If the variant did not converge at all for any run, the best values found by the runs are recorded. The outcome of the experimentation is provided in Table 2.

3.1.2 Order of Operator Imports

At first, we need to analyze G3-PCX and identify the operators which are suitable for unimodal problems. GWFA's basic operations are summarized below:

- The best solution of the population is always used as one of the parents for recombination. The rest of the $\mu - 1$ parents are solutions randomly chosen from the population.
- Using parent centric crossover (PCX), generate λ child solutions from the ($\mu - 1$ randomly selcted parents + 1 global best solution).
- Two parents are then randomly selected from the population and the top two solutions from the set of (λ child solutions + two randomly selected parents) replace these two parents.

From the summary, it is evident that the G3-PCX uses the global best solution as the index parent to improve the quality of the entire population under consideration. Also, the global best solution is always used at any point of the algorithm. So, a good solution surpassing the current global best can readily participate in the child creation process and does not need to wait for the iteration to end. That's why it is a steady state algorithm which can be extremely helpful for unimodal functions. Some of these operators along with some other common EA operators can help GWFA to improve its performance over unimodal problems.

At the beginning, only the S_1 criterion was considered for the GWFA variants. The order in which different operators of G3-PCX are included to the standard GWFA is explained below:

- **Standard GWFA:** At first, the standard version of GWFA was applied over the three unimodal problems mentioned before. For F_{elp} function, it was able to find a convergence for all the independent runs, but for F_{sch} function, it converged only 4 times and for F_{ros}, it did not converge at all.
- **Adding Mutation:** Mutation is a very common operation used in EAs which provides random perturbations to the candidate solutions. From the

earlier discussion, it can be noted that in the standard variant pf GWFA, mutation is only applied to the DAs. RAs don't participate in mutation after the recombination process. So, mutation was a natural choice for the first operator for importing. But, from the experimenation, it can be seen that Mutation worsens the solutions for all the three cases. So, Mutation was never used after this point.

- **Adding SSU:** Here the goal was to import the steady state operation used in G3-PCX. As evident from the inner workings of GWFA, it uses multiple global bests (DAs) as parents. So, the steady state update (SSU) procedure was implemented by moving an RA to the DA pool if its fitness exceeded the worst DA after position update. The replaced DA was then added to the RA pool. This process made a huge difference in the performance of standard GWFA for all the three cases.

- **Adding Selection:** Getting inspired from the improvement in GWFA, in the next variant of GWFA, Tournament Selection was added to increase the selection pressure for the elite candidates. In GWFA, elite solutions are called DAs and they guide the RAs towards a better direction. Every RA randomly chooses a DA to follow. But this variant of GWFA selects a DA for each RA through Tournament Selection. From the results, it can be seen that Selection also helped the algorithm improve further.

- **Adding Parent Child Comparison:** Previously GWFA replaced the childern created from the parents directly without comparing their fitness values. Parent Child Comparison (PCC) preserves the children only if they are able surpass their parents in terms of fitness. But after adding PCC to GWFA, the quality of the solutionss degraded. So, the use of PCC was discarded for these problems.

- **Adding Parent Centric Crossover:** The next intuition was that converting the position update equation in GWFA with some other method may help the algorithm. So, the central recombination process of G3-PCX known as the PCX operator was used in place of the position update procedure. PCX uses three parents for each RA, namely: the RA itself, a selected DA and the average of all the better RAs than the present RA (known as LA in GWFA). The DA serves as the index parent and finally PCX returns a recombination of these three parents. PCX helped the algorithm to a large extent and finally this version of GWFA was able to get convergences for the Rosenbrock's function.

- **Using Single DA:** G3-PCX always uses the global best solution to update the population. It was interesting to observe how GWFA would perform if a single DA was used. From common intuition, for unimodal problems, it is a good idea to focus too much on the global best solution. From the experimentation also, it was observed that using single DA helped in faster conervgence of the algorithm.

- **Random LA Selection:** GWFA uses the average of all the better RAs for an RA as a source of guidance. But the problem with this approach is that the better RAs are **known** to have more fitness than the RA under consideration but the quality of their avergae is **uncertain**. So, the next modification of GWFA was to use a randomly selected RA (as LA) from the pool of better RAs, instead of taking their average as the guiding factor. It worked better than the previous approach.

A quick highlight of the operator import procedure can found in Figure 2.

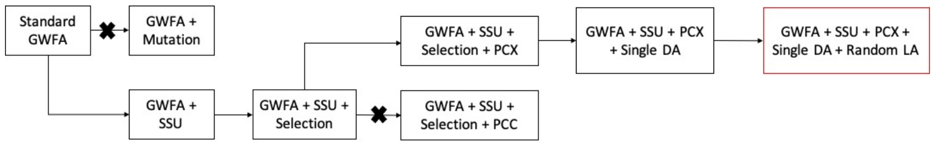
Standard GWFA
GWFA + Mutation
GWFA + SSU
GWFA + SSU + Selection
GWFA + SSU + Selection + PCC
GWFA + SSU + Selection + PCX
GWFA + SSU + PCX + Single DA
GWFA + SSU + PCX + Single DA + Random LA

Figure 2: Incremental Process of Importing EA operators to GWFA for Unimodal Problems

The entire operator import strategy is presented in Figure 3. The operators which helped GWFA improve are highlighted in red, while the operators which did not help are grayed out.

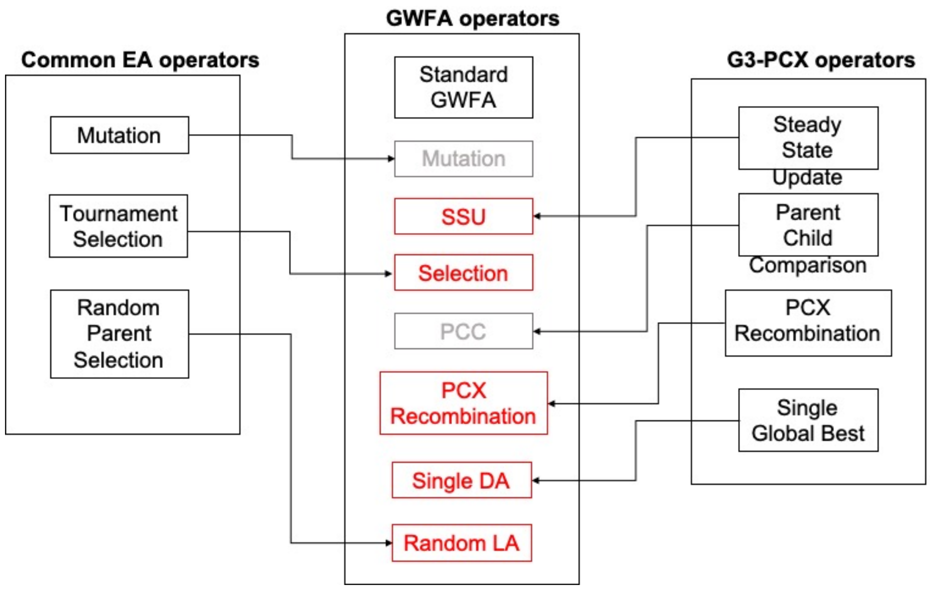

Figure 3: Entire list of Imports through Algorithmic Equivalence for Unimodal Problems

3.1.3 Experimental Outcome

The results obtained for all the variants of GWFA is displayed in Table 2. It can be observed how the number of functions needed to get convergence decreased as different operators were adeded to GWFA. Specially the F_{ros} function is really complex to be optimized. GWFA did not get any converegence for the function before using PCX operator. The final variant of GWFA was also tested for the S_2 criterion and it has been compared with G3-PCX. From the results, it can be observed that GWFA is able to achieve performance comparable to G3-PCX except for Rosenbrock's function which means that borrowing operators from G3-PCX has made GWFA somewhat algorithmically equivalent to G3-PCX and at the core, these two algorithms have become very similar in nature.

Criteria	$\mathbf{F_{elp}}$			$\mathbf{F_{sch}}$			$\mathbf{F_{ros}}$		
	Best	**Mean**	**Worst**	**Best**	**Mean**	**Worst**	**Best**	**Mean**	**Worst**
	Standard GWFA								
$\mathbf{S_1}$	78480	264011.2	469880	29940 (4)	171300	898620	15.63071727	17.68300715	18.75319299
	GWFA + Mutation								
$\mathbf{S_1}$	147640	294700	548920	31080 (1)	31080	31080	15.43421954	17.89707144	18.79622757
	GWFA + SSU								
$\mathbf{S_1}$	20440	48371.2	246760	29280 (29)	43080	879360	15.68396041	17.74399581	18.39445204
	GWFA + SSU + Selection								
$\mathbf{S_1}$	16760	20900	25040	26520 (47)	40320	818640	14.38492712	18.78694878	68.24623085
	GWFA + SSU + Selection + PCC								
$\mathbf{S_1}$	20720	25040	36920	34320 (35)	55380	956100	12.92479599	17.73948282	71.36594349
	GWFA + SSU + Selection + PCX								
$\mathbf{S_1}$	12160	17680	29640	33420	43080	59640	619720	697230	846960
	GWFA + SSU + Selection + PCX + Single DA								
$\mathbf{S_1}$	1832	2648	3056	3948	5164	5772	609500	733360	903970
	GWFA + SSU + Selection + PCX + Single DA + Random LA								
$\mathbf{S_1}$	1040	1460	1880	2160	2470	3090	19790	48170	735380
$\mathbf{S_2}$	6920	7550	8180	15180	16730	17970	609500 (47)	733360	903970
	G3-PCX								
$\mathbf{S_2}$	5744	6624	7372	14643	16326	17712	14847 (38)	22368	25797

Table 2: Experimental Outcomes for the incremental additions of EA operators to GWFA

3.2 Multimodal Problems

In order to make GWFA applicable over multimodal functions, different sets of operators are needed to satisfy the goal for multimodal problems. The main objective is to distribute the population over the different peaks present in the fitness landscape. The operators used for unimodal functions cannot be used for multimodal functions. Two different test functions mentioned in Table 3 are used to validate the performance of GWFA and the variants: a sinusoidal function with equal peaks (F_{sineq}) and another sinusoidal function with distinct peaks ($F_{sindist}$). There are 5 peaks in total for each of the functions.

Function Alias	**Function Expression**
F_{sineq}	$f(x) = -\sin^6(5\pi x)$
$F_{sindist}$	$f(x) = -e^{2x}\sin^6(5\pi x)$

Table 3: Mathematical Expressions of the two multimodal functions

Instead of focusing on a single algorithm for multimodal function, the common concepts used to tackle this objective are used to improve GWFA's performance over multimodal problems.

3.2.1 Experimental Setting

As the goal is to distribute the population over the different peaks of the function, a great indicator of the quality of the solution set would be to check the number of solutions placed close to different optima. For this purpose a radius of 0.1 is used for the problems. So, solution with functional value within 0.1 distance from the nearest peak's functional value would be considered to be included in the count. The goal of the improvement should be to distribute these counts properly over the 5 different optima.

3.2.2 Process of Operation Import

The way to import different operations for multimodal functions was to observe GWFA properly and identifying the spots which were restricting the distribution of the population. Different operations were then added to counter that. So, the process of importing these operations were:

- **Restricted Mutation on DAs:** The first thing to note over here is that, the DAs should not move too much in the search space and should remain in their own basins trying to find the best point in the basin. So, it was natural to restrict the movement of the DAs. In order to implement that, the mutation on the DAs were restricted and only 1% change in every dimension was allowed thorugh the mutation process.

- **Nearest DA:** For every recombination in GWFA, three parents participates in the process: a randomly selected DA, an LA and the RA itself. In order to make the RAs move towards their closest optimal points, it was necessary to consider the closest DA, instead of randomly selecting it.

- **Nearest LA:** Just like nearest DAs, it was also necessary to select the nearest LA (nearest RA fitter than the current RA) as one of the parents.

- **Clearing:** Finally, the main operator which is used heavily to satisfy the goal of multimodal problems, i.e., clearing [6] was used in GWFA. The point of the clearing algorithm is to reduce the fitness of a solution if it is close to multiple other solutions. The way this idea was implemented in GWFA was to distribute the DAs properly over the search space. For this variant, the DAs are not the fitness-wise best solutions of the population. They are the most properly distributed set of best points. The global best is still selected for the DA pool. The next DA is the next fittest solution which is at some distace from the global best solution. The third DA in the pool is the next fittest solution which is at some distance from both the solutions present in the current DA pool. This is how the DA pool is populated incrementally by adding a DA each time till the entire pool is filled.

A quick summary of the Operation Import procedure for multimodal problems is shown in Figure 4.

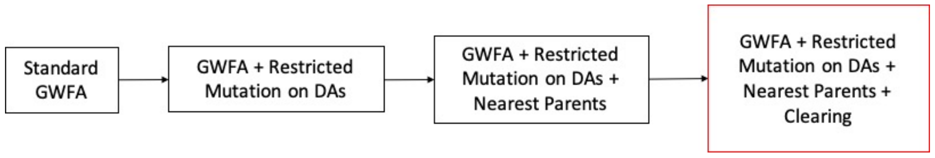
Standard GWFA
GWFA + Restricted Mutation on DAs
GWFA + Restricted Mutation on DAs + Nearest Parents
GWFA + Restricted Mutation on DAs + Nearest Parents + Clearing

Figure 4: Incremental Process of Importing EA operators to GWFA for Multimodal Problems

3.2.3 Experimental Outcome

The number of final points placed close to the 5 optima of the problem for the standard GWFA and the final clearing GWFA are displayed in Table 4. It can be observed that for F_{sin}, the standard GWFA distributes the solutions mostly around the 4^{th} optimum, while the final version of GWFA containing the clearing operator is able to properly distribute its population around all the optima. The same trend can be observed for the other function with different peaks as well. For $F_{sindist}$, the standard GWFA implementation has very few number of population which are placed close to the optima (16 around the 2^{nd} optimum and 5 around the 1^{st} one) whereas clearing GWFA can distribute the population around all the optima for different peaks.

Function Name	**Proximity Index**	**Standard GWFA**	**Clearing GWFA**
F_{sin}	1st Optimum	0	13
	2nd Optimum	0	20
	3rd Optimum	1	21
	4th Optimum	91	27
	5th Optimum	1	19
$F_{sindist}$	1st Optimum	5	12
	2nd Optimum	16	20
	3rd Optimum	0	18
	4th Optimum	0	22
	5th Optimum	0	28

Table 4: Experimental Outcome of the standard and clearing versions of GWFA over Multimodal Problems

The final convergence of both the approaches are presented in Figure 5. For more visual demosntration and iteration-wise changes of the population, please refer to: Convergence Demonstration.

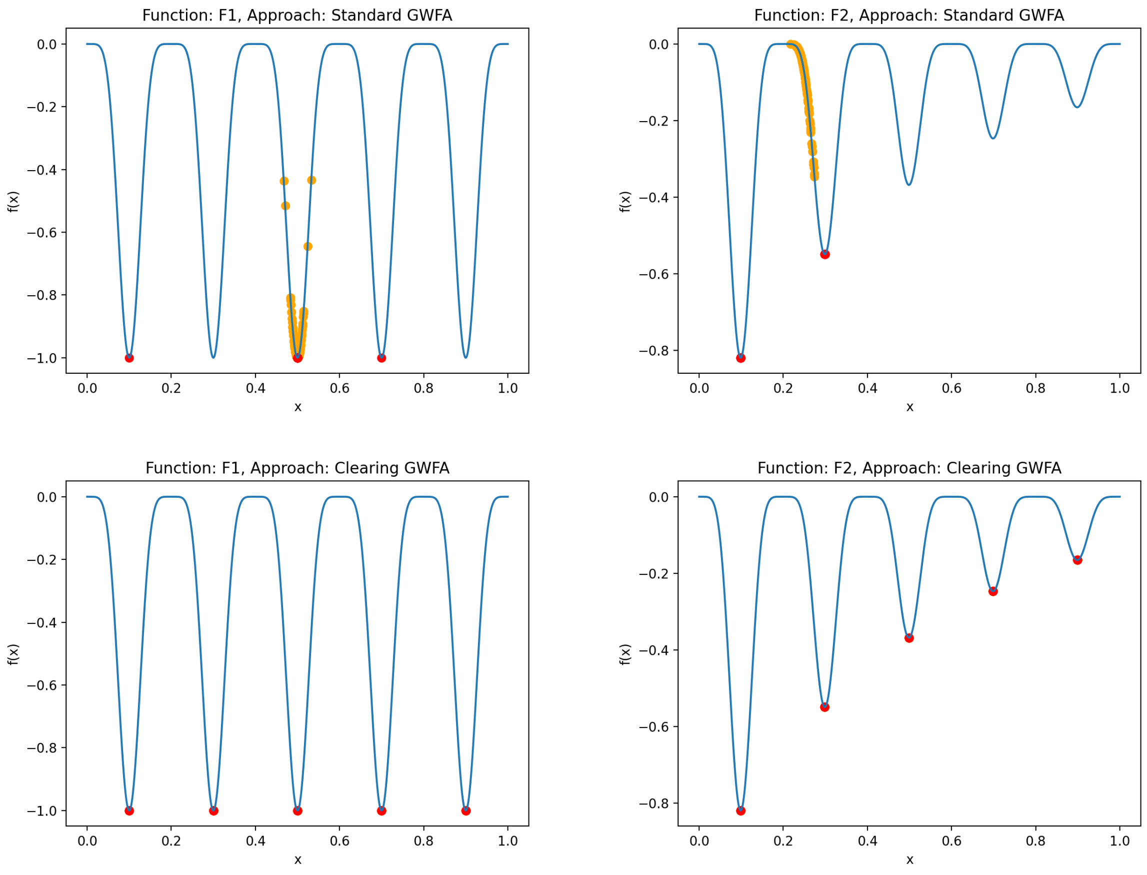

Figure 5: Final Convergence of the Population of Standard and Clearing GWFA for the 2 Functions

4 Conclusion and Future Plan

In conclusion, it can be stated that starting from the standard version of GWFA till the end version, it has gone thorugh a lot of improvement by borrowing operators from several EA concepts for both unimodal and multimodal problems. Converting GWFA to an EA framework made it easier to identify different EA operators for importing. This has helped to detect different strengths and weaknesses of GWFA in varying problem settings. This work can be further extended by applying GWFA in other problem domains like multi-objective optimization problems through algorithmic equivalence. This experimentation has proved that although there are different algorithms working great for different problem domains, at the core, some algorithms may be structurally similar to other algorithms. Researchers can take advantage of this structural similarity through algorithmic equivalence to extend applicability of one algorithm over different problem domains.

References

[1] Kalyanmoy Deb and Nikhil Padhye. Improving a particle swarm optimization algorithm using an evolutionary algorithm framework. *KanGAL report*, 2010:003, 2010.

[2] Ritam Guha, Soulib Ghosh, Kushal Kanti Ghosh, and Ram Sarkar. Groundwater flow algorithm: A novel hydro-geology based optimization algorithm. 2020.

[3] David E Goldberg, Jon Richardson, et al. Genetic algorithms with sharing for multimodal function optimization. In *Genetic algorithms and their applications: Proceedings of the Second International Conference on Genetic Algorithms*, pages 41–49. Hillsdale, NJ: Lawrence Erlbaum, 1987.

[4] Xinjie Yu and Mitsuo Gen. *Introduction to evolutionary algorithms*. Springer Science & Business Media, 2010.

[5] Kalyanmoy Deb, Ashish Anand, and Dhiraj Joshi. A computationally efficient evolutionary algorithm for real-parameter optimization. *Evolutionary computation*, 10(4):371–395, 2002.

[6] Alain Pétrowski. A clearing procedure as a niching method for genetic algorithms. In *Proceedings of IEEE international conference on evolutionary computation*, pages 798–803. IEEE, 1996.

[7] Alan Pétrowski. An efficient hierarchical clustering technique for speciation. *Evolution. Technical report, Institute National des Telecommunications, Evry, France, Technique Report*, 1997.