Groundwater Flow Algorithm: A Novel Hydro-geology based Optimization Algorithm

Ritam Guha[a], Soulib Ghosh[a], Kushal Kanti Ghosh[a], Ram Sarkar[a]

[a]*Dept. of Computer Science and Engineering, Jadavpur University, Kolkata, India - 700032*

Abstract

A novel meta-heuristic nature-inspired optimization algorithm known as Groundwater Flow Algorithm (GWFA) is proposed in this paper. GWFA is inspired from the movement of groundwater from recharge areas to discharge areas. It follows a position update procedure guided by Darcy's law which provides a mathematical framework of groundwater flow. The proposed optimization algorithm has been evaluated on 23 benchmark functions. The significance of the results is statistically validated using Wilcoxon rank-sum, Friedman and Kruskal Walis tests. To prove the robustness of the algorithm, it has been further applied on several standard engineering problems. From these exhaustive experiments, it has been observed that the proposed GWFA can outperform many state-of-the-art optimization algorithms.

Keywords: Groundwater Flow Algorithm, Optimization, Hydro-geology, Engineering application, Meta-heuristic, Uni-modal, Multi-modal

1. Introduction

Optimization deals with the process of searching for the best solution in a given scenario. An algorithm employed to find such a feasible solution is called an optimization algorithm [1, 2, 3]. The set of optimization algorithms can

☆This document is a collaborative effort.

Email addresses: ritamguha16@gmail.com (Ritam Guha), ghoshsoulib@gmail.com (Soulib Ghosh), kushalkanti1999@gmail.com (Kushal Kanti Ghosh), ramjucse@gmail.com (Ram Sarkar)

be broadly classified into two different categories: deterministic and stochastic. Deterministic optimization algorithms [4, 5] utilize analytical properties of data to search for the global optimum solution with theoretical guarantees. These procedures are used when it is absolutely necessary to find the global optimum solution. On the other hand, stochastic algorithms try to find near-optimal results through some approximations. A class of stochastic algorithms is called heuristic [6, 7]. Although heuristics do not guarantee to find the best feasible solution, they surely find a near-optimal solution within a reasonable time duration. In case of NP-hard problems, it is almost impossible to find the best solution within a finite amount of time. That is why heuristic algorithms have become a very popular and widely-explored research topic in recent times to solve various time constrained optimization problems.

Heuristic algorithms consider a particular problem as a black box with a set of inputs and outputs. The inputs are the variables of the problem and outputs are the optimum solutions, and their corresponding values. A heuristic search starts with creating a set of random inputs as the candidate solutions to the problem. The search is continued by evaluating each solution, noting the objectives values, and modifying the solutions based on their current outputs. These steps are repeated until a termination criterion is met, which can be either a threshold on the maximum number of iterations or maximum number of function evaluations. Regardless of the specific structure, at some (rather, each) stage, such algorithms require to compare two solutions to decide which one is better. An objective function or fitness function is used to evaluate the merit of each solution.

Although heuristics are very effective in solving real-life challenging problems, there are many difficulties when solving optimization problems [8]. Besides, optimization problems have diverse characteristics which are not similar at all. Therefore, it is a challenge to the researchers to design optimization procedures which are problem-independent in nature. Heuristic algorithms typically use problem specific properties to find better approximations. To overcome such difficulties, a new class of heuristics has been created, which is known as meta-

heuristics. The word 'meta-heuristic' has been first coined by Fred Glover [9]. It is a high-level problem-independent framework with some specific strategies used to devise optimization algorithms. In the last two decades, meta-heuristic algorithms have become quite popular in the literature due to [10] their (i) simple concepts and structures, (ii) almost derivation free mechanisms, (iii) ability to handle local optima, (iv) flexibility, and (v) easy and effective hardware implementation.

The most important characteristic [11] of a good meta-heuristic algorithm is to find a good balance in its exploration and exploitation capabilities. With exploration or diversification property, the algorithm aims to find out the 'promising' areas where the global optima may lie, whereas the exploitation or intensification property is responsible for searching the areas already discovered in a more precise manner in order to pinpoint the solution. Exploration preserves the diversity, and stops the algorithm to converge prematurely to some local optima. Exploitation helps the algorithm converge. Interestingly, many of such meta-heuristic algorithms are intuitively inspired by nature. Simple natural elements have proved their skills to perform optimization in the most trivial way possible. As a result of this, researchers have developed various nature-inspired meta-heuristic optimization algorithms such as Genetic Algorithm (GA) [12] inspired from genetic crossover and mutation, Particle Swarm Optimization (PSO) [13] based on flocking of birds, Ant Colony Optimization (ACO) [14] inspired from foraging behavior of ants etc. This has motivated us to carefully observe natural occurrences and propose a novel meta-heuristic developed based on the flow of groundwater from recharge areas to discharge areas.

The contributions of this paper are:

1. Developed a novel meta-heuristic nature-inspired optimization algorithm called Groundwater Flow Algorithm (GWFA) inspired from flow of groundwater.
2. Evaluated GWFA against 23 standard benchmark functions consisting of 7 -uni-modal, 6 multi-modal and 10 fixed dimensional multi-modal func-

tions.

3. Compared GWFA with some classic and state-of-the-art optimization algorithms used in the literature.
4. Explained statistical significance, convergence and overall superiority of the algorithm in terms of obtained results.
5. Applied the algorithm over 5 engineering application problems to prove its robustness.

The rest of the paper is organized as follows: Section 2 provides a brief overview of the research work being carried out in the domain of nature-inspired optimization, Section 3 presents the proposed algorithm in detail with proper explanation of its ability to handle exploration-exploitation balance, Section 4 provides various test results obtained for the proposed algorithm and their comparison with some classic as well as recently proposed optimization algorithms, Section 5 finally concludes the manuscript and provides future scope of improvement.

2. Related Work

As a modern trend in optimization, researchers across the globe are proposing various meta-heuristic algorithms to tackle different avenues of optimization problems. Meta-heuristic algorithms can be divided into different categories based on varied criteria: single solution based and population based [15], nature inspired and non-nature inspired [16], metaphor based and non-metaphor based [17]. From the 'inspiration' point of view, these algorithms can roughly be divided into four categories [18]: Evolutionary, Swarm inspired, Physics based, and Human related.

- **Evolutionary algorithms** are basically inspired from biology. They utilize crossover and mutation operators for the evolution of the initial population, usually selected in a random fashion, over the iterations and eliminates the poor solutions, thereby ensuring the improved solution. GA

is a well-known method of this category which follows the Darwin's theory of evolution. Co-evolving algorithm [19], Cultural algorithm [20], Genetic programming [21], Grammatical evolution [22], Bio-geography based optimizer [23], Stochastic fractal search [24] etc. are some well-known evolutionary algorithms.

- **Swarm inspired algorithms**, on the other hand, imitate individual and social behavior of swarms, herds, schools, teams or any group of animals. Every individual has its own specific behavior, but the behavior of the collection of individuals gives us a way to solve complex optimization problems. A popular algorithm belonging to this category is PSO, devised by following the behavior of flock of birds. Another important algorithm of this category is ACO, designed by mimicking the foraging process of the ant species. Several popular and some recent algorithms of this category are: Shuffled frog leaping algorithm [25], Bacterial foraging [26], Artificial bee colony [27], Firefly algorithm [28], Grey Wolf optimizer (GWO) [29], Ant Lion optimizer (ALO) [30], Whale optimization algorithm [31], Grasshopper optimization algorithm (GOA) [32], Squirrel search algorithm [33], Harris Hawks optimization (HHO) [34] etc.
- **Physics based algorithms** are inspired following the rules used to govern a physical process. Some inspiring physical processes are based on metallurgy, physics, chemistry, complex dynamic systems and even music. An oldest one belonging to this category is Simulated Annealing (SA) [35], formulated by following the annealing [36] process of the metals in metallurgy and materials sciences. Another popular method of this category is Gravitational search algorithm (GSA) [37], designed by following gravity and mass interaction. Besides, there are other methods belonging to this category which include Self propelled particles [38], Harmony search (HS) algorithm [39], Black hole optimization [40], Sine Cosine algorithm [41], Multi-verse optimizer [42], Find-Fix-Finish-Exploit-Analyze [43] etc.
- **Human related algorithms** look for the global optima by following

human behavior. Teaching-Learning-Based optimization [44] is a popular one belonging to this category which is formulated by following the improving procedure of class grade. Apart form this, some other important methods of this category include Society and civilization [45], League championship algorithm [46], Fireworks algorithm [47], Tug of war optimization [48], Volleyball Premier League algorithm [49].

Presence of such a significant number of meta-heuristic algorithms, clearly raises the question about the need for another meta-heuristic algorithm. However, as indicated by *No Free Lunch* [50] theorem for optimization, there cannot be any single algorithm which will be equally applicable for all the optimization problems desiring optimal solutions. With each new algorithm following any regular or natural phenomenon, researchers primarily aim to provide some new facet to the algorithm where both exploration and exploitation will have a superior trade-off, thereby trying to get away from the local optima and eventually compass to the global optima. Nevertheless, accomplishing these objectives are not straightforward, hence motivating researchers to propose new algorithm that can be applicable to different problem domains.

3. Groundwater Flow Algorithm

In this section, a detailed outline of the proposed algorithm is provided which is followed by the mathematical description of the optimization procedure.

3.1. Hydro-geological Perspective/Inspiration

Three-fourth of the Earth's surface is covered with water. Most of this water can be seen with bare eyes when people go to beaches or stand at the side of rivers or ponds. But still there is water which cannot be seen unless a hole is dug up in the ground deep enough to reach the water tables. This water is known as groundwater which accounts for 1.7% of all of Earth's water and 30.1% of the freshwater [51]. The depth of ground water varies from places to places. In some places (like marsh), water may occur very close to the ground

surface whereas in some other places, it may appear at hundreds of feet below the ground level (as in deserts). Water nearing the ground surface may be few hours old, at moderate depth, it may be few hundreds years old and the water nearing the earth's core maybe some thousand years old flowing from one place to another underground. The level of water underground is known as water table.

This water occupy the pores in the soil and move from one place to another. But not 100% of the pores are occupied by water, some space may be occupied by air as well. Specially the pores of the soil closer to the Earth's surface are significantly occupied by air. This part of the soil is called unsaturated region whereas the part where all the pores are occupied by water is known as saturated region. In the final stage of water cycle, i.e. precipitation, water falls on the Earth's surface as rain, snow, hail etc. The precipitated water either flows over the surface following the slope and reaches a stream, or it infiltrates the ground and enters into the unsaturated region. Some of this water gets used by plants or evaporates due to heat but rest of the water goes deeper into the ground and recharges the ground water. The area where the precipitated water flows past the unsaturated zone to the water table is called the Recharge area (RA) while the areas where the groundwater flows to (streams, lakes etc.) are called Discharge Areas (DAs). Groundwater flow is known as the flow of groundwater from RAs to DAs. The concept can be made clear by looking at Figure 1.

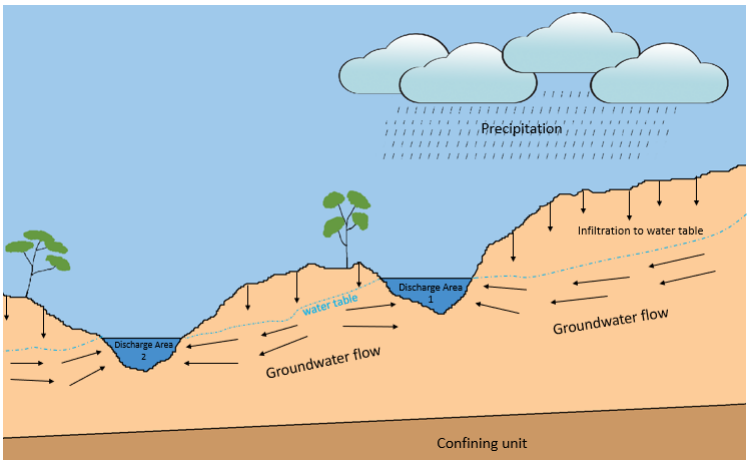

Figure 1: Image describing the concept of groundwater flow.

The reason of this flow is typically the gravitational force. Water at RAs has higher potential energy due to their high elevations. So, they move to DAs with lower elevation by overcoming the internal frictions (determined by viscosity). Due to the gravitational pull, groundwater slowly flows through layers of rocks, soil and sand which are known as aquifers (originated from two Latin words 'aqua' or water and 'ferre' or carrier). The flow of groundwater has always been of great interest to the mankind apparent from many Biblical references as well. Finally in 1855 and 1856, a French engineer named Henry Darcy performed various experiments [52] and was able to mathematically define the flow of groundwater which has been named as the Darcy's Law. Darcy defined the flow of water through sand beds during his experimentation but later it has been generalized for other fluids and porous mediums as well.

In this work, the flow of groundwater is mathematically modelled to solve optimization problems. Darcy's law helps in the movement of searching agents efficiently while performing optimization.

3.2. Mathematical model

The most important aspect of the proposed optimization approach is the position update policy which is guided through Darcy's law. In this section, the

mathematical definitions of Darcy's law are discussed.

During experimentation, Darcy had discovered that the velocity with which groundwater flows is dependent on two major factors: height difference and gap in positions. Darcy's experimental setup has been presented in Figure 2. Consider that groundwater is flowing from point A to point B as shown in Figure 2. The velocity of this flow is directly proportional to the height difference (Δh) between the two points, but inversely proportional to the length of the gap (L) between them. The ratio of this two factors (i) is termed as the Hydraulic Gradient (HG).

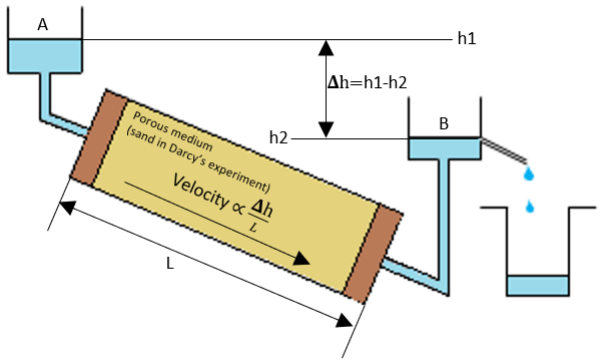

Figure 2: Setup used by Darcy to conduct experimentation on groundwater flow. Darcy used sand beds as the medium between the water sources but later it got generalized for any porous medium.

$$i = \frac{\Delta h}{L} \tag{1}$$

HG clearly suggests that when the height gap between the points is more and the length between them is shorter, the water flows more rapidly. The velocity is directly proportional to the HG. So, the velocity term can be expressed as:

$$v_d = k \times i \tag{2}$$

where k is the constant of proportionality also known as coefficient of permeability. This velocity is known as the discharge velocity. It is an idealistic term which considers water flows through the entire soil between the two points but it is not realistic as water only flows through the pores. So, there is another term used for defining velocity, called the seepage velocity, which is the velocity of the water flowing through the pores of the soil. It can be represented as:

$$v_s = \frac{v_d}{\phi} \tag{3}$$

where ϕ is called the porosity of the soil. Porosity is the ratio of combined volume of pores to the entire volume of soil. Seepage velocity is ultimately the velocity with which the groundwater flows from one place to another. Depending on the value of the porosity (property of the soil), the seepage velocity may vary.

The proposed model uses seepage velocity to update the position of the candidate solutions.

3.3. Optimization approach

In any population-based meta-heuristic optimization algorithm, it starts with some candidate solutions which act as the search agents. These search agents traverse various paths in the search space in pursuit of better solutions. GWFA uses some GWSs as its search agents from where water flows in different directions to search for the global optimum solution. The optimum solutions are represented by the DAs where the GWSs are guided. Hence, the objective here is to look for the most optimal DA by flowing groundwater from the RAs. It is to be noted that in this section GWSs, agents and candidate solutions are used to represent the same thing. In a broader sense, we can say that each GWS is a search agent which carries a candidate solution. The algorithm consists of three separate phases: initialization, position update and exploitation of DAs. A flowchart describing the entire algorithm is presented in Figure 3.

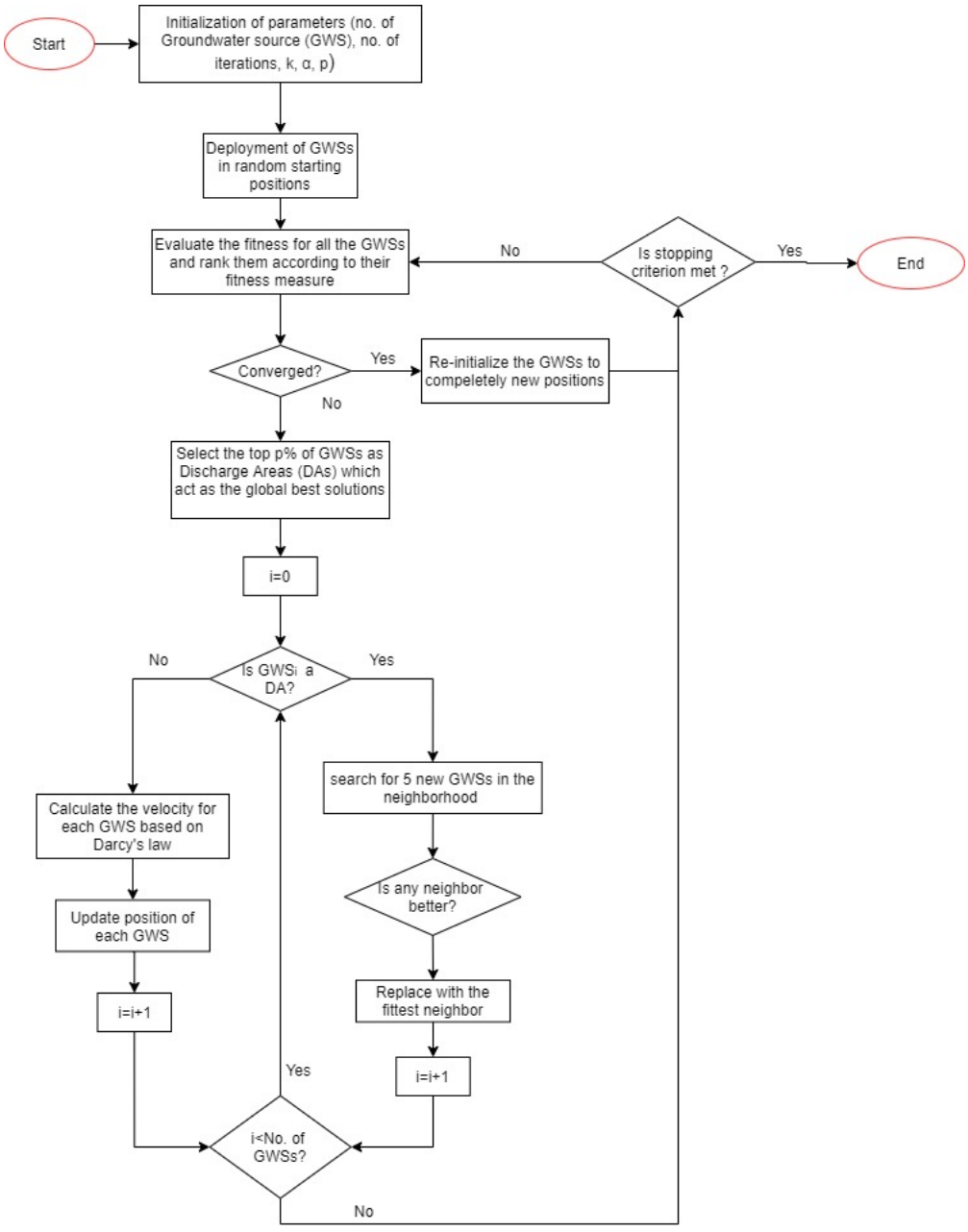

Figure 3: Flowchart describing the optimization approach adapted in GWFA.

3.3.1. Initialization

The GWSs act as the search agents in the proposed optimization approach traversing various paths in search of a solution to the problem. At any stage of the algorithm, the solution represented by any GWS is called a candidate solution. At the first stage of the algorithm, a certain number of GWSs are initialized with random candidate solutions. Suppose n GWSs are initialized and the function under consideration describing the optimization objective (say f) has D dimensions. Then the GWSs will be represented as:

$$x_{ij} = lb_j(f) + rand_j(1) \times (ub_j(f) - lb_j(f)) \quad \begin{matrix} (i = 1, 2, \ldots, n) \\ (j = 1, 2, \ldots, D) \end{matrix} \tag{4}$$

where x_{ij} represents the value of i^{th} candidate solution in j^{th} dimension, $ub_j(f)$ and $lb_j(f)$ are the upper bound and lower bound of function f respectively and $rand_j(1)$ is a random number restricted in $[0, 1]$. In this way, n GWS positions are initialized which act as the initial solutions to the optimization problem. For rest of the discussion, $\vec{x_i}(t)$ represents the i^{th} GWS in the population in t^{th} iteration of the algorithm. Each candidate solution is guided by a velocity term. The velocity of every individual GWS is also initialized with 0.

$$\vec{v_i}(0) = 0 \tag{5}$$

Here $\vec{v_i}(0)$ represents the initial velocity of i^{th} GWS.

3.3.2. Position update

This is the most crucial part of the proposed algorithm. After getting the initial GWSs, their positions are updated based on groundwater flow rules and abiding by Darcy's law. At the beginning of the position update policy, the quality of all the solutions are calculated using the objective function. It is to be noted that objective function in an optimization problem is the function whose value needs to be minimized (minimization problems) or maximized (maximization problem). Objective function provides a fitness measure for the

solutions. The lower (or higher) the function value is for minimization (or maximization) problem, the fitter the solution is. After getting quality measure for each and every GWS, the fittest $p\%$ of them are assigned as DAs. The rest of the GWSs are then considered to form RA. The objective for the GWSs in RA is to approach one of these appropriate DAs, thereby improving the fitness of the candidate solutions. Apart from the DAs, if there are GWSs in the RA which are fitter than the others then the less fit GWSs will also get guided towards the fitter GWSs. Intuitively it can be considered that fitter GWSs in the RA are at lower elevations in RA. Hence, water from higher GWSs keep flowing towards these GWSs. The movement of water will be guided by three important factors:

1. Previous flow of the water.
2. Flow towards selected DA.
3. Flow towards the fitter GWSs in the RA.

For each of the GWSs in the RA, one DA is selected randomly as shown in Equation 6. But instead of defining a flow to every fitter GWS in RA, an average of the lower GWSs is constructed to reduce computation. The solution representing GWS is then guided towards the selected DA and lower averaged GWS (LA) using Darcy's law of groundwater flow. From the discussion provided in Section 3.2, it can be observed that groundwater is mainly guided by two important terms: height difference (Δh) and length of gap (L). In the proposed optimization approach, Δh is calculated as the difference in the current candidate solutions and L represents the difference in their current functional values. The mathematical representations of these terms are shown below:

$$\vec{DA_i}(t) = random_i(\vec{DA_1}(t), \vec{DA_1}(t), \ldots, \vec{DA_g}(t)) \; where \; g = floor(\frac{p \times n}{100}) \quad (6)$$

$$\vec{LA_i}(t) = mean(\vec{x_j}(t)) \; \forall j : f(\vec{x_j}(t)) < f(\vec{x_i}(t)) \quad (7)$$

where $\vec{DA_i}(t)$ and $\vec{LA_i}(t)$ represent the selected DA and the lower averaged

GWS for i^{th} GWS.

$$\Delta \vec{h_{id}}(t) = \vec{DA_i}(t) - \vec{x_i}(t) \tag{8}$$

$$\Delta \vec{h_{il}}(t) = \vec{LA_i}(t) - \vec{x_i}(t) \tag{9}$$

$$L_{id}(t) = f(\vec{DA_i}(t)) - f(\vec{x_i}(t)) \tag{10}$$

$$L_{il}(t) = f(\vec{LA_i}(t)) - f(\vec{x_i}(t)) \tag{11}$$

Here the subscripts d and l correspond to the DA and LA counterparts respectively.

HG for each GWS is then calculated as the ratio of Δh and L terms. Darcy's law states that the discharge velocity is directly proportional to the HG. Therefore, by multiplying the HG with a constant of proportionality (k), discharge velocity (vd) for every individual is computed. k is also known as the coefficient of permeability.

$$\vec{HG_{id}}(t) = \frac{\Delta \vec{h_{id}}(t)}{L_{id}(t)} \tag{12}$$

$$\vec{HG_{il}}(t) = \frac{\Delta \vec{h_{il}}(t)}{L_{il}(t)} \tag{13}$$

$$\vec{vd_{id}}(t) = k \times \vec{HG_{id}}(t) \tag{14}$$

$$\vec{vd_{il}}(t) = k \times \vec{HG_{il}}(t) \tag{15}$$

The discharge velocities directed towards the DA and LA are combined using Equation 16. The weightage provided to $\vec{vd_{id}}(t)$ is α while $\vec{vd_{il}}(t)$ gets the weightage of $1 - \alpha$ where α is in $[0, 1]$. The value of α is determined through

experimentation provided in later sections. α controls the level of importance of guidance provided by DA and LA.

$$\vec{vd_i}(t) = \alpha \times \vec{vd_{id}}(t) + (1 - \alpha) \times \vec{vd_{il}}(t) \tag{16}$$

Discharge velocity is velocity of the water when there is no obstruction along the flow. In general, there are porous mediums known as aquifers in the path of the groundwater. So, the water only flows through the pores of the aquifers. Hence, a new term is introduced which is known as the seepage velocity (sv). Seepage velocity is determined by dividing the discharge velocity by porosity of the intermediate medium. The expression for seepage velocity can be represented as:

$$\vec{vs_i}(t) = \frac{\vec{vd_i}(t)}{\phi_i} \tag{17}$$

where ϕ is the porosity of the medium. ϕ is a very special term in the algorithm because it helps to control the exploration-exploitation balance in the process. When position of a GWS gets updated, all the dimensions of the solution are not updated. The number of dimensions to be updated (d) is guided by a factor called the control factor (CF) which is changed over the iterations as:

$$CF(t) = (1 - \frac{t}{T}) \times D \tag{18}$$

where T is the maximum number of iterations. The number of dimensions to be updated is selected by randomly choosing an integer d in the range $[CF(t), D]$. As evident from expression 18, the value of $CF(t)$ decreases as the iterations progress. Therefore, at the beginning, there are more chances of a high number of dimensions getting updated (i.e. ensuring exploration) compared to the later stages (i.e. ensuring exploitation). The value of ϕ can be expressed as:

$$\phi_i = \frac{d_i}{D} \tag{19}$$

Finally the velocity of any individual GWS can be calculated by taking the combination of its previous velocity (learning from past experience) and its seepage velocity (learning from recent experience). The contributions of both the factors to the velocity is the same. The velocity can be represented as:

$$\vec{v_i}(t) = \beta \times v_i(\vec{t-1}) + \gamma \times \vec{vs_i}(t) \tag{20}$$

Here $\beta = \gamma = 0.5$. Value of the velocity is used to update the positions of the GWSs (i.e. candidate solutions). For each individual, d random dimensions are selected from the entire set. The values of these positions are modified according to the velocity as:

$$\vec{list}_i = (random_i[CF(t), D])_{1\times d} \tag{21}$$

$$\vec{x}_{ij}(t+1) = \begin{cases} \vec{x}_{ij}(t) + \vec{v_{ij}}(t) \text{ if } j \in list_i \\ \vec{x}_{ij}(t) \text{ otherwise} \end{cases} \tag{22}$$

After the update, it may so happen that the GWSs get converged. At this situation, every GWS will represent the same candidate solution. It can be identified when all the GWSs are having same fitness measure. If such a situation occurs, the algorithm stores the best GWS found till now, discards all the solutions and re-initialize every GWS in the same way as discussed in the Initialization section.

3.3.3. DA exploitation

In this procedure, the GWSs flow towards DAs. So, if the DAs are not changed over iterations, the solutions will automatically get converged. In order to avoid that scenario, the DAs perform exploitation by checking their neighborhoods. Some dimensions of every DA are selected and their values are changed to find a neighborhood solution. In this way, for each DA, 5 neighborhood solutions are constructed and the one with the best fitness measure is selected to replace the older and less fit DA. If there is no fitter neighbor, DA is not changed. If at a certain point, the solutions get converged, the DAs does not

participate in exploitation and they also get re-initialized.

Exploration-exploitation balance: One of the major concerns in any optimization procedure is to maintain a balance between the exploration and exploitation of the searching. It is a dilemma requiring the process to make a choice between searching a new unexplored area of the search space and searching the same area with deeper insights. So, one needs to make an appropriate decision at this point to proceed. GWFA manages to control the trade-off in a very effective way. GWFA uses not one but multiple DAs which are treated as the global best of the current iteration which are followed by the other GWSs. For each GWS, a DA is selected randomly. It gives rise to the exploration of the search space because more avenues are to be looked for. Had it been only one DA, all the other GWSs would have been followed its path and finally the searching procedure would have found limited or only a part of the search space. The next important term which helps in achieving the balance is CF. As discussed before, CF supervises how many dimensions of the GWSs get updated by the velocity. At the beginning of the algorithm, there are chances of many dimensions getting changed but during later stages the number of dimension updates is restricted to a lower value. More updates of dimensions lead to exploration because it helps to inspect new areas of the search space. On the other hand, when less number of dimensions are updated, only nearby neighborhoods are checked, thereby enhancing the exploitation capability of the algorithm. In the later stages of the algorithm, DAs participate in exhaustive exploitation to intensify the exploitation in the search space. As evident from the discussion, GWFA has the ability to maintain the trade-off between exploration and exploitation.

Computational Complexity: The complexity of the algorithm depends on number of GWSs (n), number of iterations (I) and number of dimensions in the objective function (d). The computational complexity is described in terms of big-O notation. From the pseudo code of GWFA presented in Algorithm 3.3.3,

it is clear that the functional values are being evaluated only in two occasions: computation of fitness values and DA exploitation. Time complexity of each function evaluation is $O(d)$. During computation of fitness values, each GWS is evaluated once. So, for I iterations and n GWSs, there will be $(n \times I)$ function evaluations. On the other hand, DA exploitation is performed for only $(\frac{n \times p}{100})$ DAs. For each DA, 5 neighbors are evaluated. So, for I iterations, there will be $(5 \times \frac{n \times p \times I}{100})$ function evaluations. Time requirement in case of all the other computations is $O(1)$. Therefore, overall time complexity (TC) all the model can be calculated as:

Fitness computation:	$O(n \times I \times d)$
DA exploitation:	$O(5 \times d \times \frac{n \times p \times I}{100})$
Overall TC:	$O(n \times I \times d) + O(5 \times d \times \frac{n \times p \times I}{100})$
	$= O(max(ndI, \frac{ndpI}{20}))$
	$= O(ndI)$ [as p is a constant and does not depend on n]

From the discussion, it can be seen that GWFA requires polynomial time for computation. This proves that it is a time-efficient algorithm and is applicable to optimization.

Algorithm 1 Pseudo code of Groundwater Flow Algorithm

Require: n, Max_Iter, k, α, f

Initialize n GWSs with their random positions $x_i(0)$ where $(i = 1, \ldots, n)$.
$t \leftarrow 0$
while $t < Max\_Iter$ **do**
  for i=1 to n **do**
    Calculate fitness value of each GWS as $fit(i) = f(x_i(t))$.
  end for
  Sort the GWSs according to increasing (or decreasing) order of fitness values for minimization (or maximization) problem.
  if $fit(1) == fit(n)$ **then**
    Store the best GWS, re-initialize all GWSs.
    // need to re-initialize to circumvent premature convergence
    $t \leftarrow t + 1$.
    Goto Step 3.
  end if
  Select the top p% (or $j = floor(\frac{n \times p}{100})$) of the GWSs as DAs.
  DA=$\{DA_1, DA_2, \ldots\ DA_j$
  $CF(t) = 1 - \frac{t}{Max\_Iter}$.
  for i=1 to n **do**
    if $x_i(t) \notin$ DA **then**
      Compute $\vec{DA}_i$ and $\vec{LA}_i$. (Equations-6,7)
      Determine the HGs.(Equations-12,13)
      Compute discharge velocities of the flow towards DA and LA. (Equations-14,15)
      Combine the two discharge velocities.(Equation-16)
      Calculate the seepage velocity $v_i\vec{(t)}$.(Equation-17)
      Find the overall velocity of the flow.(Equation-20)
      Create the list of dimensions to be updated $\vec{list}_i$.
      $\vec{x}_{ij}(t+1) \leftarrow \begin{cases} \vec{x}_{ij}(t) + v_{ij}\vec{(t)} \text{ if } j \in list_i \\ \vec{x}_{ij}(t) \text{ otherwise} \end{cases}$
    else
      Create 5 neighbors from $x_i(t)$ through perturbation
      if Fitter neighbor found **then**
        Replace the fittest neighbor with $x_i(t)$
      end if
    end if
  end for
end while

4. Experimentation

This section includes the performance of the proposed optimization algorithm called GWFA over some standard bechmark functions and a comparative study with some other state-of-the-art methods. The section is further divided into six sub-sections. The first sub-section contains the demonstration of the benchmark functions. The second sub-section comprises of parameter tuning experimentation for GWFA. The third sub-section reports the obtained the results and a detailed comparative study with other optimization methods. Next two sub-sections consist of testing outcomes for convergence and statistical significance respectively. The final sub-section includes the outcomes of the proposed algorithm when evaluated on five standard engineering applications.

4.1. Benchmark function description

In order to evaluate the proposed approach, 23 benchmark functions stated in [53] are selected. The 23 benchmark functions can be broadly classified into three categories: 7 uni-modal functions (F_1 to F_7), 6 multi-modal functions (F_8 to F_{13}) and 10 multi-modal functions with fixed dimensions (F_{14} to F_{23}). The description of the uni-modal, multi-modal, and multi-modal functions with fixed dimensions are given in Tables 1, 2 and 3 respectively. The tables contain the function description, domain and optimum value. The uni-modal functions have only one optimal value, and obtaining this value helps estimating the exploitation ability of any algorithm. The multi-modal functions have more than one optimal value, which essentially test the exploration ability of the algorithm. The final category is similar to the multi-modal functions but with small dimensions.

Table 1: Description of uni-modal functions used in present work

Function	**Domain**	**Optimum**	**Position**
$F_1 = \sum_{i=1}^{n} x_i^2$	$[-100, 100]^{30}$	0	$(0)^{30}$
$F_2 = \sum_{i=1}^{n} x_i + \prod_{i=1}^{n} x_i$	$[-10, 10]^{30}$	0	$(0)^{30}$
$F_3 = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j\right)^2$	$[-100, 100]^{30}$	0	$(0)^{30}$
$F_4 = \max\{\lvert x_i \rvert, 1 \leq i \leq n\}$	$[-100, 100]^{30}$	0	$(0)^{30}$
$F_5 = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	$[-30, 30]^{30}$	0	$(1)^{30}$
$F_6 = \sum_{i=1}^{n} (x_i + 0.5)^2$	$[-100, 100]^{30}$	0	$(0)^{30}$
$F_7 = \sum_{i=1}^{n} i x_i^4 + random[0, 1]$	$[-1.28, 1.28]^{30}$	0	$(0)^{30}$

Table 2: Description of multi-modal functions used in present work

Function	**Domain**	**Optimum**	**Position**
$F_8 = \sum_{i=1}^{n} -x_i \sin\sqrt{\lvert x_i \rvert}$	$[-500, 500]^{30}$	-12569.5	$(420.96)^{30}$
$F_9 = \sum_{i=1}^{n} [x_i^2 - 10cos(2\pi x_i) + 10]$	$[-5.12, 5.12]^{30}$	0	$(0)^{30}$
$F_{10} = -20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - \exp(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i) + 20 + e$ e is Eular constant	$[-32, 32]^{30}$	0	$(0)^{30}$
$F_{11} = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]^{30}$	0	$(0)^{30}$
$F_{12} = \frac{\pi}{n}\{10\sin^2\pi y_i + \sum_{i=1}^{n-1}[1 + 10\sin^2\pi y_{i+1}] + (y_n - 1)^2\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{x_i + 1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, x_i > a \\ 0, -a \le x_i \le a \\ k(-x_i - a)^m, x_i < a \end{cases}$	$[-50, 50]^{30}$	0	$(1)^{30}$
$F_{13} = 0.1\{\sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2[1 + \sin^2(3\pi x_i)] + (x_n - 1)^2[1 + \sin^2(2\pi x_n)]\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	$[-50, 50]^{30}$	0	$(1)^{30}$

Table 3: Description of the multi-modal functions with fixed dimension used in present work

Function	**Domain**	**Optimum**	**Position**
$F_{14} = (\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^2})^{-1}$	$[-65.53, 65.53]^2$	1	(-32,32)
$F_{15} = \sum_{i=1}^{11}\Big[a_i - \frac{x_1(b_1^2 + b_1 x_2)}{b_1^2 + b_1 x_3 + x_4}\Big]^2$	$[-5, 5]^4$	0.00030	(.1928,.1908), (.1231,.1358)
$F_{16} = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	$[-5, 5]^2$	-1.0316	(.089,.712), (-.089,.712)
$F_{17} = \Big(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\Big)^2 + 10\Big(1 - \frac{1}{8\pi}\Big)\cos x_1 + 10$	[-5,10] $\times$ [0,15]	0.398	(-3.14,12.27), (3.14,12.275), (9.42,2.42)
$F_{18} = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)$	$[-5, 5]^2$	3	[0,-1]
$F_{19} = -\sum_{i=1}^{4} c_i \exp\Big(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\Big)$	$[0, 1]^2$	-3.86	(0.114,.556,.852)
$F_{20} = -\sum_{i=1}^{4} c_i \exp\Big(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\Big)$	$[0, 1]^6$	-3.32	(.201,.15,.477, .275,.311,.657)
$F_{21} = -\sum_{i=1}^{5}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	$[0, 14]^4$	-10.1532	$5a_{ij}$
$F_{22} = -\sum_{i=1}^{7}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	$[0, 10]^4$	-10.4028	$7a_{ij}$
$F_{23} = -\sum_{i=1}^{10}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	$[0, 10]^4$	-10.5363	$10a_{ij}$

4.2. Parameter tuning

The proposed algorithm has two control parameters namely α and k. This section mainly focuses on finding the most optimal parameter combination. For this purpose, two functions from each category are selected: F_1, F_6 from uni-modal functions, F_9, F_{11} from multi-modal functions and $F_{20}, F22$ from multi-modal functions with fixed dimensions. The values considered for each parameter are as follows : $\alpha = [0.5, 0.55, 0.6, 0.65, 0.7]$ and $k = [0.4, 0.5, 0.6, 0.7]$. As a result, a total of 20 $(5 * 4 = 20)$ combinations are checked. During the testing, number of runs, GWSs and iterations remain fixed as $30, 50$ and 1000 respectively.

The best values obtained for each function is recorded and summed for every parameter combination. The best parameter combination is the one which has produced the best sum value. A visual interpretation of the obtained results are provided in Figure 4. From the experimentation, the best values of α and k are found to be 0.65 and 0.7 respectively. The final values for all the parameters are provided in Table 4.

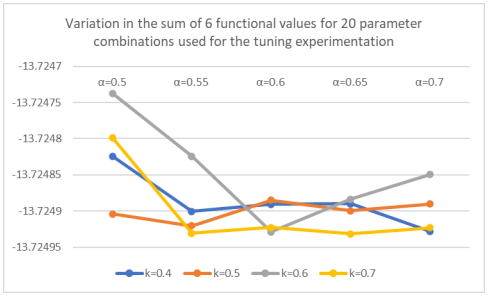

Figure 4: Results obtained by varying the parameters of GWFA

Table 4: Final values of all the parameters.

Parameters	**Final Values**
Number of runs	30
Number of Water Source	50
Number of Iteration	1000
α	0.65
k	0.7
p	20

4.3. Results and comparison

The proposed algorithm is evaluated over 23 benchmark functions mentioned in Section 4.1. For proper experimentation, the algorithm is run for 30 times and the best, average and standard deviation of the obtained values are stored. The result obtained for GWFA is compared with other state-of-the-art methods that include GSA, PSO, Gravitational Particle Swarm (GPS), PSOGSA, Grey Wolf Optimizer (GWO), Elite Opposition based GWO (EOGWO), Improved GWO (IGWO), Coyote Optimization Algorithm (COA) and Equilibrium Optimizer (EO). To maintain a fair comparison environment, population size and number of iterations for each algorithm are taken as 50 and 1000 respectively (same as GWFA). The best value, average value and the standard deviation are also calculated for every method. All the compared methods have some manually tuned parameters. The values of those parameters for each algorithm are tabulated in Table 5.

Table 5: Parameter settings of the methods with which the proposed method is compared.

Algorithm	**Parameters and their corresponding values**
GSA	$c_1 = 2, c_2 = 2, G_0 = 1$
PSO	$c_1 = 2, c_2 = 2$
GPS	$c_1 = 2, c_2 = 2, c_3 = 0.5, c_4 = 1.5$
PSOGSA	$c_1 = 0.5, c_2 = 1.5$
GWO	$\vec{d} = 2$(linearly decreased over iterations)
EOGWO	$\vec{d} = 2(linearlydecreasedoveriterations)$
IGWO	$\vec{d} = 2(linearlydecreasedoveriterations)$
COA	$nfeval_{max} = 20000, n_p = 20, n_c = 5$
SOGWO	$\vec{d} = 2(linearlydecreasedoveriterations)$
EO	$a1 = 2, a2 = 1, GP = 0.5$

The detailed comparative study is separately carried out for all three categories of functions. Tables 6, 7 and 8 contain the results for uni-modal functions, multi-modal functions and multi-modal functions with fixed dimensions respectively. A method is said to have the best result for a particular function if it has obtained the lowest best value. If multiple methods have the same best values, average values are checked and if still some methods are indistinguishable in terms of average value, finally standard deviation is checked to decide the best one.

The uni-modal functions mainly test the exploitation ability of any algorithm. As it is evident from Table 6, the proposed GWFA has outperformed all the other algorithms in six ($F_1 - F_4, F_6, F_7$) out of seven uni-modal functions. Not only that, the proposed algorithm has also surpassed all the other algorithms in terms of average value and standard deviation for these six functions, which proves the superiority of the algorithm. Though the proposed algorithm has not achieved the best outcome in F_5, the result is indeed competitive with others. Hence, based on the properties of the uni-modal functions, it can be stated that GWFA is capable of handling exploitation in a competent way.

multi-modal functions are predominantly used for analyzing the exploration

property due to its large number of local optima. Tables 7 shows the outstanding performance of the proposed GWFA. GWFA has achieved best result in three (F_8, F_9 *and* F_{11}) out of six functions not only in terms of best value but also in terms of average value and standard deviation. GWFA gets second best result in F_{10} and the result difference is marginal. Obtained outcomes in other two functions (F_{12}, F_{13}) are very close to the global optima. The performance in multi-modal functions clearly reveals the strong exploration ability of GWFA. Apart from GWFA, EO is the next best performer with 5 best results (2 multi-modal and 3 fixed-dimensional multi-modal functions) and SOGWO is the third best performer with 4 best values (3 multi-modal and 1 fixed-dimensional multi-modal functions).

Table 8 shows the result of GWFA in multi-modal function with fixed dimensions. In this category, GWFA has achieved impressive results as well. It has outperformed others in three (F_{17}, F_{20}, F_{22}) out ten functions. If we consider only best values, GWFA is able to achieve the best result for 7 out of the 10 functions. In case of F_{16} *and* F_{18}, the difference between the best result and GWFA result is only in terms of standard deviation. So, it is a clear indication that although GWFA has not achieved the best results in certain situations, it has given a very close competition to the best method in such situations.

In total, the proposed GWFA has achieved best results for 12 out of 23 (approx 52%) benchmark functions. The outcomes in other functions are also very close to global optima and competitive with other algorithms. Excellent performance on both uni-modal and multi-modal functions also establishes the strong exploration and exploitation abilities of GWFA.

Table 6: Comparison of obtained results over 7 uni-modal functions

Function		GWFA	GSA	PSO	GPS	PSOGSA	GWO	EOGWO	IGWO	COA	SOGWO	EO
F_1	Best	**2.08E-134**	1.10E-17	1.10E-15	6.60E-19	3.29E-19	9.07E-73	1.76E-73	3.36E-76	1.0300E -12	3.81E-79	1.95E-44
	Avg	**4.03E-133**	2.00E-17	1.30E-11	1.20E-18	4.74E-19	1.36E-70	2.81E-71	4.75E-73	1.20E-10	6.05E-77	3.32E-40
	SD	**4.05E-133**	5.50E-18	8.80E-11	3.00E-19	8.04E-19	2.57E-70	8.46E-71	9.53E-73	1.03E-10	1.49E-76	6.78E-40
F_2	Best	**1.44E-66**	1.40E-08	4.40E-09	3.30E-09	2.47E-09	3.59E-42	3.00E-42	3.77E-43	1.0308-10	3.50E-46	4.64E-24
	Avg	**7.05E-66**	2.40E-08	2.90E-06	5.20E-09	2.93E-09	5.64E-41	2.63E-41	4.32E-42	1.01E-09	1.18E-44	7.12E-23
	SD	**3.37E-66**	4.40E-09	1.30E-05	9.00E-10	2.64E-10	6.43E-41	2.73E-41	7.88E-42	4.54E-08	1.34E-44	6.36E-23
F_3	Best	**8.49E-116**	7.50E+01	1.90E+01	3.10E+00	2.92E+02	2.56E-25	1.01E-26	5.00E-26	3.38E+09	1.17E-28	7.17E-13
	Avg	**8.61E-115**	2.30E+02	1.20E+02	9.70E+01	1.82E+03	1.09E-19	1.29E-20	1.52E-20	3.38E+09	5.40E-22	8.06E-09
	SD	**9.19E-115**	1.00E+02	7.50E+01	1.10E+02	4.82E+02	3.11E-19	3.40E-20	4.02E-20	0.00E+00	2.60E-21	1.60E-08
F_4	Best	**1.64E-64**	2.10E-09	1.40E-01	8.20E-10	1.30E+01	1.28E-18	1.26E-19	4.81E-20	1.00E+02	7.08E-21	2.01E-11
	Avg	**4.79E-64**	6.40E-02	4.20E-01	1.30E+00	2.20E+01	1.94E-17	2.39E-17	8.07E-19	1.00E+02	1.18E-19	5.39E-10
	SD	**2.75E-64**	2.50E-01	1.90E-01	9.80E-01	3.89E+00	3.68E-17	5.33E-17	1.11E-18	0.00E+00	1.51E-19	1.38E-09
F_5	Best	2.81E+01	2.60E+01	2.50E+01	2.30E+01	**1.60E+01**	2.51E+01	2.59E+01	2.51E+01	7.49E+09	2.50E+01	2.49E+01
	Avg	2.86E+01	2.80E+01	2.70E+01	2.60E+01	**2.60E+01**	2.63E+01	2.67E+01	2.63E+01	7.49E+09	2.65E+01	2.53E+01
	SD	1.96E-01	1.00E+01	8.40E+00	8.80E+00	**2.50E+00**	6.69E-01	6.14E-01	7.36E-01	0.00E+00	7.62E-01	1.70E-01
F_6	Best	**0.00E+00**	7.40E-18	8.30E-16	6.00E-19	3.30E-19	1.09E-05	8.09E-06	1.27E-05	1.01E+06	6.19E-06	1.90E-06
	Avg	**0.00E+00**	1.90E-17	1.30E-12	1.20E-18	5.05E-19	4.12E-01	2.15E-01	3.29E-01	1.01E+06	2.83E-01	8.29E-06
	SD	**0.00E+00**	6.40E-18	7.10E-12	3.30E-19	9.40E-20	2.45E-01	2.30E-01	2.46E-01	0.00E+00	2.47E-01	5.02E-06
F_7	Best	**3.99E-07**	8.40E-03	1.70E-03	1.10E-03	1.50E-02	1.49E-04	1.31E-04	1.41E-04	1.36E+04	8.06E-05	3.46E-04
	Avg	**2.64E-05**	2.80E-02	7.00E-03	3.10E-03	3.30E-02	5.68E-04	4.34E-04	6.07E-04	1.36E+04	4.93E-04	1.17E-03
	SD	**3.10E-05**	1.70E-02	2.50E-03	1.20E-03	9.30E-03	3.54E-04	2.77E-04	4.33E-04	8.09E-05	2.71E-04	6.54E-04

Table 7: Comparison of obtained results over 6 multi-modal functions

Function		GWFA	GSA	PSO	GPS	PSOGSA	GWO	EOGWO	IGWO	COA	SOGWO	EO
F_8	Best	**-1.10E+04**	-4.20E+03	-1.00E+04	-8.90E+03	-8.85E+03	-7.09E+03	-7.79E+03	-7.60E+03	1.81E+04	-8.18E+03	-1.07E+04
	Avg	**-1.05E+04**	-2.70E+03	-9.00E+03	-7.50E+03	-8.09E+03	-6.07E+03	-6.27E+03	-6.31E+03	1.81E+04	-6.57E+03	-9.02E+03
	SD	**2.78E+02**	4.70E+02	5.20E+02	7.70E+02	4.71E+02	5.37E+02	7.71E+02	9.81E+02	3.83E-12	8.03E+02	5.95E+02
F_9	Best	**0.00E+00**	9.00E+00	1.80E+01	9.00E+00	4.48E+01	0.00E+00	0.00E+00	0.00E+00	2.89E+03	**0.00E+00**	**0.00E+00**
	Avg	**0.00E+00**	1.70E+01	4.10E+01	2.10E+01	7.42E+01	5.20E+00	5.29E-01	0.00E+00	2.89E+03	**0.00E+00**	**0.00E+00**
	SD	**0.00E+00**	4.30E+00	1.50E+01	6.10E+00	1.10E+01	1.89E+00	1.94E+00	1.20E-01	4.79E-13	**0.00E+00**	**0.00E+00**
F_{10}	Best	4.00E-15	2.20E-09	4.60E-09	5.40E-10	4.32E-10	7.99E-15	7.99E-15	7.99E-15	2.00E+01	**8.88E-16**	7.99E-15
	Avg	4.00E-15	3.40E-09	9.10E-08	8.80E-10	5.07E-10	1.31E-14	1.40E-14	1.11E-14	2.00E+01	**8.88E-16**	8.34E-14
	SD	2.41E-30	4.10E-10	2.00E-07	1.30E-10	4.70E-11	2.73E-15	3.20E-15	3.74E-15	0.00E+00	**0.00E+00**	2.53E-14
F_{11}	Best	**0.00E+00**	2.00E+00	5.10E-15	0.00E+00	2.86E-06	0.00E+00	0.00E+00	**0.00E+00**	9.00E+03	**0.00E+00**	**0.00E+00**
	Avg	**0.00E+00**	4.30E+00	1.20E-02	2.30E-02	2.33E-01	5.23E-04	1.68E-03	**0.00E+00**	9.00E+03	**0.00E+00**	**0.00E+00**
	SD	**0.00E+00**	1.60E+00	1.20E-02	3.00E-02	3.50E-01	2.61E-03	4.80E-03	**0.00E+00**	0.00E+00	**0.00E+00**	**0.00E+00**
F_{12}	Best	6.76E-02	6.20E-20	1.60E-18	**4.70E-21**	1.01E+00	6.57E-03	1.72E-06	1.32E-02	2.56E+10	2.62E-02	3.63E-08
	Avg	1.26E-01	2.50E-02	1.50E-02	**5.00E-02**	4.46E+00	2.66E-02	2.29E-02	3.29E-02	2.56E+10	5.61E-02	7.97E-07
	SD	3.97E-02	6.10E-02	3.60E-02	**1.30E-01**	1.80E+00	1.55E-02	1.85E-02	1.76E-02	0.00E+00	1.42E-02	7.69E-07
F_{13}	Best	9.79E-01	1.22E-18	**9.90E-131**	3.75E-08	9.29E-20	1.58E-05	1.58E-05	8.94E-02	4.10E+10	1.42E-05	1.84E-06
	Avg	1.53E+00	2.10E-18	**2.00E-31**	8.48E-02	2.20E-03	3.25E-01	2.57E-01	3.46E-01	4.10E+10	3.53E-01	2.93E-02
	SD	2.97E-01	5.00E-19	**4.30E-31**	8.00E-02	4.50E-03	1.56E-01	1.64E-01	1.73E-01	0.00E+00	1.28E-01	3.53E-02

Table 8: Comparison of obtained results over 10 fixed dimension multi-modal functions

Function		GWFA	GSA	PSO	GPS	PSOGSA	GWO	EOGWO	IGWO	COA	SOGWO	EO
	Best	9.98E-01	1.00E+00	1.00E+00	1.00E+00	1.00E+00	9.98E-01	9.98E-01	9.98E-01	5.00E+02	9.98E-01	**9.98E-01**
F_{14}	Avg	8.03E+00	3.80E+00	1.00E+00	1.00E+00	1.39E+00	3.11E+00	3.82E+00	2.88E+00	5.00E+02	3.43E+00	**9.98E-01**
	SD	4.16E+00	2.60E+00	3.20E-17	5.80E-01	8.75E-01	3.73E+00	3.86E+00	3.59E+00	5.99E-14	3.72E+00	**1.54E-16**
	Best	3.08E-04	1.40E-03	3.10E-04	3.10E-04	3.10E-04	3.07E-04	3.07E-04	3.08E-04	4.10E+01	**3.07E-04**	3.08E-04
F_{15}	Avg	3.09E-04	4.10E-03	1.20E-03	4.10E-04	4.10E-04	4.36E-03	5.24E-03	3.10E-04	4.10E+01	**2.38E-03**	2.40E-03
	SD	1.42E-06	3.20E-03	4.00E-03	3.40E-04	3.40E-04	8.17E-03	8.68E-03	4.44E-08	7.49E-15	**6.03E-03**	6.10E-03
	Best	-1.03E+00	-1.00E+00	-1.00E+00	-1.00E+00	-1.00E+00	-1.03E+00	-1.03E+00	-1.03E+00	6.42E+03	-1.03E+00	**-1.03E+00**
F_{16}	Avg	-1.03E+00	-1.00E+00	-1.00E+00	-1.00E+00	-1.00E+00	-1.02E+00	-1.02E+00	-1.02E+00	6.42E+03	-1.03E+00	**-1.03E+00**
	SD	8.73E-08	4.00E-16	2.30E-16	2.80E-16	2.80E-16	4.80E-09	3.45E-09	4.13E-09	9.59E-13	3.75E-09	**6.04E-16**
	Best	**3.98E-01**	4.00E-01	4.00E-01	4.00E-01	**3.98E-01**	3.98E-01	3.98E-01	3.98E-01	9.00E+03	3.98E-01	**3.98E-01**
F_{17}	Avg	**3.98E-01**	4.00E-01	4.00E-01	4.00E-01	**3.98E-01**	3.98E-01	3.98E-01	3.98E-01	9.00E+03	3.97E-01	**3.98E-01**
	SD	**0.00E+00**	3.40E-16	3.40E-16	3.40E-16	**0.00E+00**	3.36E-07	4.82E-07	1.18E-04	0.00E+00	4.86E-07	**0.00E+00**
	Best	3.00E+00	3.00E+00	3.00E+00	3.00E+00	**3.00E+00**	3.00E+00	3.00E+00	3.00E+00	7.67E+04	3.00E+00	3.00E+00
F_{18}	Avg	3.00E+00	3.00E+00	3.00E+00	3.00E+00	**3.00E+00**	3.00E+00	3.00E+00	3.00E+00	7.67E+04	3.00E+00	3.00E+00
	SD	3.85E-06	2.20E-15	3.10E-15	1.60E-15	**8.40E-16**	4.88E-06	3.60E-06	1.69E-06	0.00E+00	4.63E-06	1.56E-15
	Best	-3.86E+00	-3.90E+00	**-3.90E+00**	**-3.90E+00**	-3.86E+00	-3.86E+00	-3.86E+00	-3.86E+00	-3.86E+00	-3.86E+00	-3.86E+00
F_{19}	Avg	-3.86E+00	-3.60E+00	**-3.90E+00**	**-3.90E+00**	-3.86E+00	-3.86E+00	-3.86E+00	-3.86E+00	-3.86E+00	-3.86E+00	-3.86E+00
	SD	4.02E-07	3.00E-01	**3.10E-15**	**3.10E-15**	2.19E-15	1.05E-03	2.36E-03	3.59E-06	4.55E-13	2.71E-03	2.59E-15
	Best	**-3.32E+00**	-3.30E+00	-3.30E+00	-3.31E+00	-3.32E+00	-3.32E+00	-3.32E+00	-3.32E+00	-3.32E+00	-3.32E+00	-3.32E+00
F_{20}	Avg	**-3.32E+00**	-1.90E+00	-3.30E+00	-3.30E+00	-3.31E+00	-3.25E+00	-3.27E+00	-3.26E+00	-3.32E+00	-3.27E+00	-3.27E+00
	SD	**1.97E-05**	5.40E-01	5.50E-02	2.40E-02	6.07E-01	7.04E-02	7.55E-02	6.55E-02	6.66E-05	7.37E-02	5.70E-02
	Best	-1.02E+01	-5.10E+00	-1.00E+01	-1.00E+01	-1.02E+01	-1.02E+01	-1.02E+01	-1.02E+01	**-1.02E+01**	-1.02E+01	-1.02E+01
F_{21}	Avg	-9.97E+00	-5.10E+00	-7.20E+00	-8.50E+00	-6.17E+00	-9.95E+00	-9.14E+00	-8.94E+00	**-1.02E+01**	-9.66E+00	-8.55E+00
	SD	9.30E-01	7.40E-03	3.30E+00	3.10E+00	3.74E+00	1.01E+00	2.07E+00	2.21E+00	**1.63E-04**	1.51E+00	2.76E+00
	Best	**-1.04E+01**	-1.00E+01	-1.00E+01	-1.00E+01	-1.04E+01	-1.04E+01	-1.04E+01	-1.04E+01	-1.04E+01	-1.04E+01	-1.04E+01
F_{22}	Avg	**-1.04E+01**	-7.50E+00	-9.10E+00	-1.00E+01	-8.87E+00	-1.02E+01	-1.02E+01	-1.02E+01	-1.04E+01	-1.04E+01	-9.34E+00
	SD	**2.66E-04**	2.70E+00	2.80E+00	7.20E-15	3.14E+00	1.05E+00	1.05E+00	1.05E+00	2.94E-03	4.43E-04	2.44E+00
	Best	-1.05E+01	**-1.10E+01**	-1.10E+01	-1.10E+01	-1.05E+01	-1.05E+01	-1.05E+01	-1.05E+01	-1.05E+01	-1.05E+01	-1.05E+01
F_{23}	Avg	-1.04E+01	**-1.00E+01**	-9.40E+00	-1.00E+01	-7.90E+00	-1.03E+01	-1.02E+01	-1.05E+01	-1.05E+01	-1.05E+01	-9.64E+00
	SD	9.87E-01	**7.80E-01**	2.80E+00	1.60E+00	3.69E+00	1.08E+00	1.62E+00	2.20E-04	9.45E-05	5.41E-01	2.39E+00

4.4. Convergence testing

For any optimization algorithm, convergence is one of the most important criteria for comparison. A good optimization approach should be able to provide fast but steady convergence. That is why convergence test is necessary for

any optimization algorithm to prove its efficiency. This section describes the convergence capability of the proposed GWFA. The number of GWSs is kept at 50. On the other hand, the iteration is confined to 500 for better visualization of the convergence graph. Figures 5, 6 and 7 demonstrate the convergence graph for uni-modal functions, multi-modal functions, and multi-modal functions with fixed dimensions respectively. For comparison of convergence, the best two procedures apart from GWFA according to Section 4.3 namely EO and SOGWO are selected. There are two sub-parts of each convergence graph: parameter space and objective space. Parameter space provides the visual representation of the functions in 2D while the objective space contains the convergence curves. The X-axis in objective space indicates the iteration number, whereas the Y-axis depicts the global best value among of all population obtained till the current iteration. The convergence graphs clearly indicate that GWFA has the ability to provide steady convergence. The convergence rate is so high that in many cases, the curve has nearly touched the Y -axis. From the graphs, it is also clear that EO and SOGWO are good performers as well in terms of convergence. From Figure 5, it can be observed that GWFA has provided marginally better convergence compared to both its competitors. The trend remains almost same in Figure 6 as well except for function F_8 where SOGWO provides better convergence. But in case of fixed-dimensional multi-modal functions presented in Figure 7, EO displays best convergence capabilities in most of the cases.

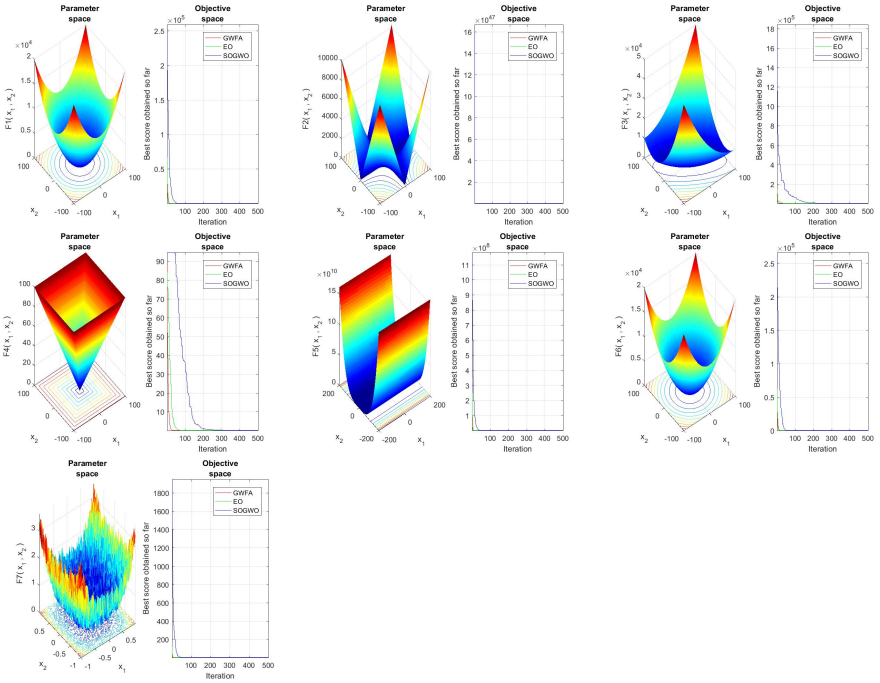

Figure 5: 2-D representations and convergence curves for 7 uni-modal benchmark functions

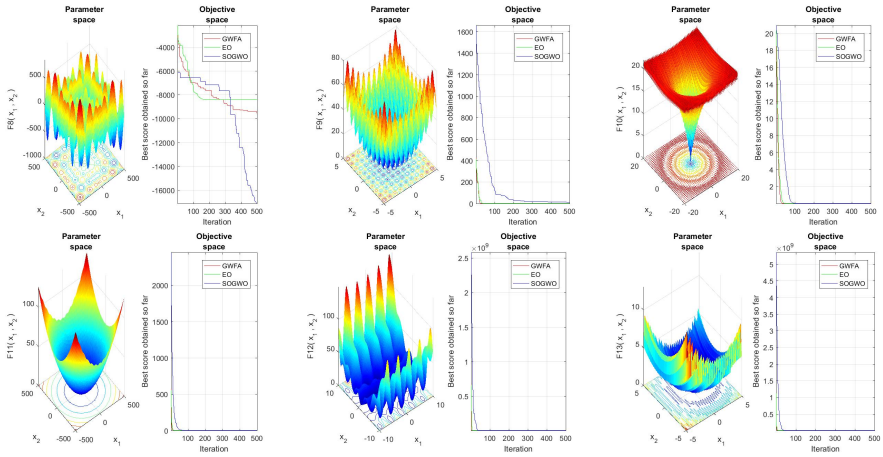

Figure 6: 2-D representations and convergence curves for 6 multi-modal benchmark functions

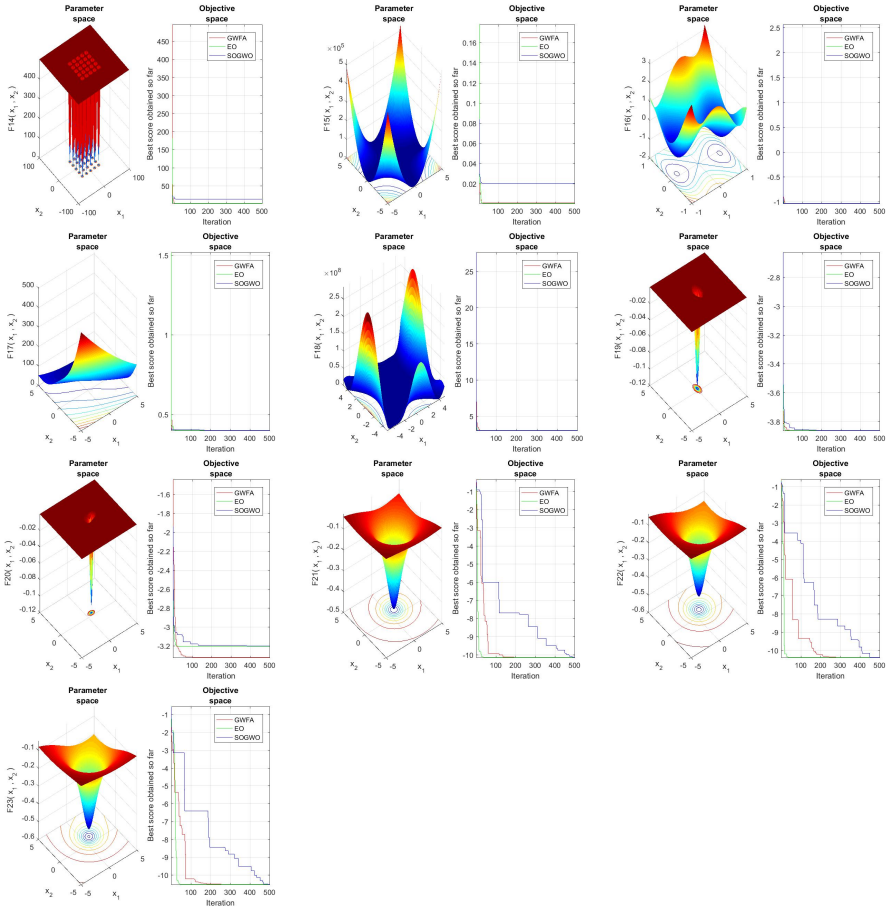

Figure 7: 2-D representations and convergence curves for 10 multi-modal benchmark functions with fixed dimensions

4.5. Statistical Analysis

To determine the statistical significance of the proposed GWFA algorithm, a non-parametric statistical test, known as Wilcoxon rank-sum test [54], has been performed. This is done in order to check whether the results of an algorithm are statistically different from other algorithms [55].

The null hypothesis states that if two sets of results are from the same distribution, any difference in the two mean ranks comes only from sampling error. If the distributions of two results are statistically different, then the generated p-value from the test statistics will be < 0.05 (level of significance), as

we have performed the test at 0.05% significance level, resulting in the rejection of the null hypothesis. The final results of Wilcoxon test is provided in Table 9.

Table 9: Wilcoxon rank-sum test results for 23 benchmark functions

	PSOGSA	GPS	PSO	GWO	SOGWO	EO
F_1	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
F_2	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
F_3	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
F_4	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
F_5	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
F_6	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
F_7	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
F_8	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
F_9	**0.000**	**0.000**	**0.000**	0.180	0.180	0.317
F_{10}	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
F_{11}	**0.000**	**0.000**	**0.000**	**0.000**	0.180	0.180
F_{12}	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
F_{13}	**0.000**	0.943	**0.000**	**0.001**	**0.000**	**0.000**
F_{14}	**0.000**	**0.000**	**0.000**	**0.000**	**0.001**	**0.000**
F_{15}	**0.000**	**0.000**	**0.000**	0.057	0.975	0.004
F_{16}	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.012**
F_{17}	**0.000**	**0.000**	**0.000**	0.959	0.702	**0.001**
F_{18}	**0.000**	**0.000**	**0.001**	**0.001**	**0.000**	**0.001**
F_{19}	**0.000**	**0.000**	0.253	**0.000**	**0.000**	**0.000**
F_{20}	**0.000**	**0.000**	**0.047**	**0.021**	0.159	**0.004**
F_{21}	**0.000**	**0.000**	0.299	0.530	0.629	0.766
F_{22}	**0.000**	**0.000**	0.371	0.185	0.120	**0.015**
F_{23}	**0.000**	**0.000**	**0.002**	0.075	**0.020**	**0.015**

Friedman test [56] is also performed for determining the statistical significance of the results obtained by GWFA. Friedman test is a non-parametric statistical test which provides the measure of the differences between multiple methods. It is normally used for answering the following kind of question: in

a set of n number of results (where $n \geq 2$), does at least two of the sets represent results with different median values? The null hypothesis considered here is: no significant difference in the results of the methods at 0.05% significance level. From the test results provided in Table 10, all the obtained p-values are < 0.05. This clearly proves the presence of at least one set of significant results. Kruskal-Walli's test [57] is also performed to prove the consistency of the proposed GWFA. This is a non-parametric method which compares two or more groups of results obtained. This test is also conducted at 0.05% significance level and the obtained results, mentioned in Table 10, proves the significance of the proposed method.

Table 10: p-values generated by Kruskal-Walis and Friedman test for 23 benchmark functions

Function	p-value of Kruskal Walis test	p-value of Friedman test
F_1	0.000	0.000
F_2	0.000	0.000
F_3	0.000	0.000
F_4	0.000	0.009
F_5	0.004	0.000
F_6	0.000	0.000
F_7	0.000	0.000
F_8	0.000	0.000
F_9	0.000	0.000
F_{10}	0.000	0.004
F_{11}	0.000	0.000
F_{12}	0.000	0.000
F_{13}	0.000	0.000
F_{14}	0.001	0.000
F_{15}	0.000	0.000
F_{16}	0.000	0.000
F_{17}	0.000	0.001
F_{18}	0.000	0.000
F_{19}	0.000	0.000
F_{20}	0.010	0.000
F_{21}	0.000	0.000
F_{22}	0.000	0.000
F_{23}	0.027	0.000

4.6. Application of GWFA in Engineering Problems

Engineering problems are some real world problems that involve designing and building of systems and/or products. It is basically a decision making process that contains complex objective functions and large number of decision

variables. Generally, meta-heuristic algorithms perform better than any traditional algorithm due to the proper convergence of the former to an optimal solution. Besides, meta-heuristic algorithms also have the ability to handle non-convex and non-differentiable functions. Mainly complex engineering problems involve large number of design variables. The influence of those variables on the objective function, which is to be optimized, turns out to be very troublesome and non-linear in nature. Five classical engineering design problems namely, spring, gear train, welded beam, pressure vessel and closed coil helical spring design are considered in this study. The problems are explained in this section. These problems have many local optima, but the task of the algorithm is to find the global optimum. Hence, an efficient optimization algorithm is required to solve the problems.

4.6.1. Tension/Compression Spring Design Problem

In this problem, the main objective is to minimize the weight of the coil involving three decision parameters - wire diameter d, mean coil diameter D and number of active coils N along with four inequality constraints. The objective function is given in equation 23. Any solution of this problem can be represented as: $\vec{x} = [x_1 x_2 x_3] = [dDN]$. The variable ranges are: $x_1 \in [0.05, 2.00]$, $x_2 \in [0.25, 1.30]$ *and* $x_3 \in [2.00, 15.00]$. The visual description of this problem is provided in Figure 8.

$$EF_1 = (x_3 + 2) * x_2 * {x_1}^2 \tag{23}$$

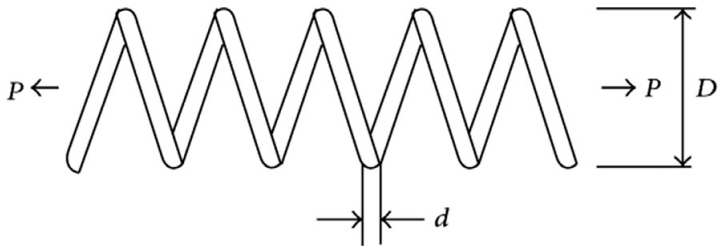

Figure 8: Pictorial representation of Tension/Compression Spring design problem

4.6.2. Gear Train Design Problem

The aim of this design problem is to minimize the cost of gear ratio of the gear train. The four decision parameters present in this problem are : T_a, T_b, T_d *and* T_f. There is no inequality constraints present in this problem. A pictorial representation of gear train design problem is given in Figure 9.

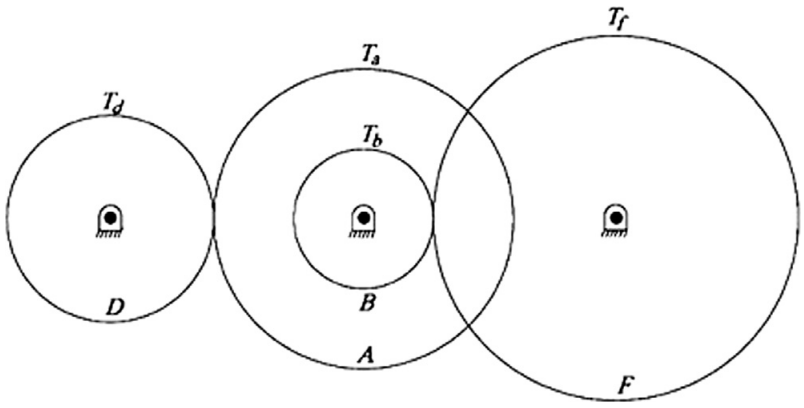

Figure 9: Pictorial representation of Gear Train design problem

The gear ratio is formulated as : $T_b T_d / T_f T_a$. On the other hand, the objective function is demonstrated in equation 24. Representation of a solution can be given as : $\vec{x} = [x_1 x_2 x_3 x_4] = [T_a T_b T_d T_f]$. The variable range is given by $x_i \in [12, 60]$

$$EF_2 = ((1/6.931) - (x_3 x_2 / x_1 x_4))^2 \tag{24}$$

4.6.3. Welded Beam Design Problem

Welded beam design problem is mainly a minimization problem consisting of four variables. The variables are : weld thickness (h), length of the bar attached to the weld (l), bar's height (t), and bar's thickness (b). The available constraints for this problem include bending stress (θ), bean deflection (δ), shear stress (τ), buckling load (P_c) and other side constraints. The problem is displayed in Figure 10.

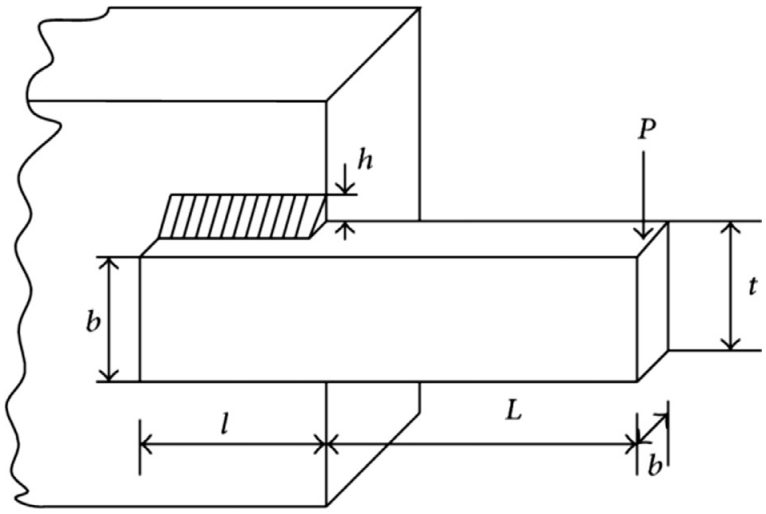

Figure 10: Pictorial representation of Welded Beam design problem

Population of this problem is represented as : $\vec{x} = [x_1 x_2 x_3 x_4] = [hltb]$. The variable ranges are : $x_1 \in [0.1, 2]$, $x_2 \in [0.1, 10]$, $x_3 \in [0.1, 10]$ *and* $x_4 \in [0.1, 2]$. Objective function is formulated in equation 25

$$EF_3 = 1.1047 x_1^2 x_2 + 0.04811 x_3 x_4 (14.0 + x_2) \tag{25}$$

4.6.4. Pressure Design Vessel Problem

The main objective of this problem is to minimize the manufacturing, welding and material cost of the pressure vessel. Four variables associated with this problem are - thickness of shell (T_S), the thickness of head (T_h) which are discrete decision variables, inner radius (R) and length of the cylindrical section of the vessel (L) those are continuous decision variables. An image is provided in Figure 11 for better visualization of the problem.

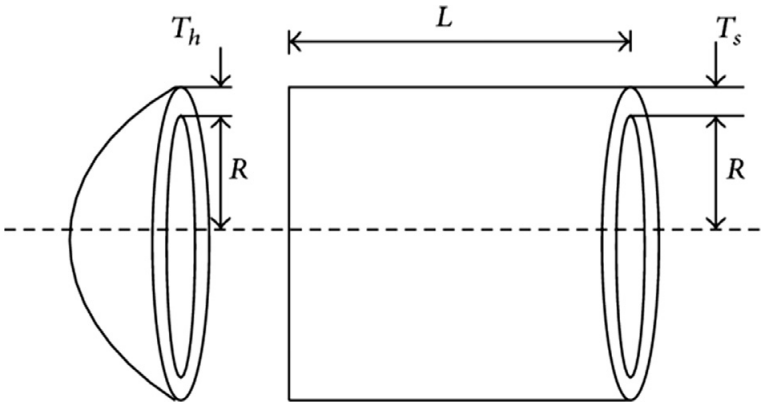

Figure 11: Pictorial representation of Pressure Design Vessel problem

Any population is formulated as: $\vec{x} = [x_1 x_2 x_3 x_4] = [T_s T_h R L]$. Objective function is described in equation 26. The variables lie in a certain range of values, which are as follows - $x_1 \in [0, 100],\ x_2 \in [0, 100],\ x_3 \in [10, 200]$ *and* $x_4 \in [10, 200]$.

$$EF_4 = 0.6224 x_1 x_3 x_4 + 1.7781 x_2 x_3^2 + 3.1661 x_1^2 x_4 + 19.84 x_1^2 x_3 \qquad (26)$$

4.6.5. Closed Coil Helical Spring Design Problem

The main aim of this design constraint problem is to decrease the volume of closed coil helical spring. Helical spring consists of closed coiled wire having the shape of a helix and is intended for tensile and compressive load. The two variables related to this problem are as follows - coil diameter(D) and wire diameter(d). There is another parameter, the number of coils (n), which can be set beforehand. The pictorial description of the problem is given in Figure 12.

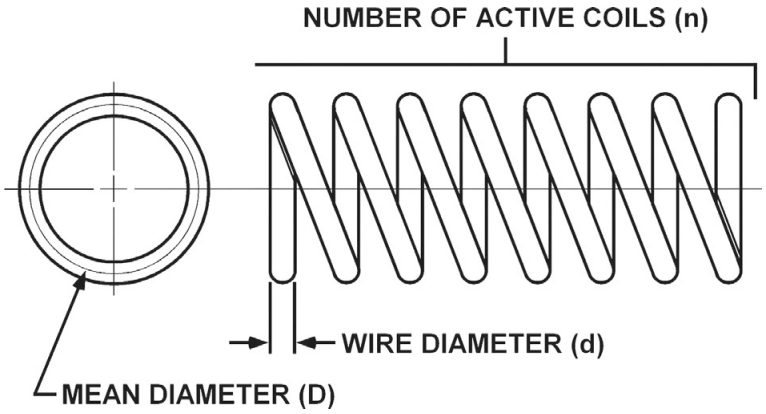

Figure 12: Pictorial representation of Closed Coil Helical Spring design problem

Any population of this problem can be devised as : $\vec{x} = [x_1 x_2 x_3] = [dDn]$. The variables maintain certain range of values which are provided as follows : $x_1 \in [0.508, 1.016]$, $x_2 \in [1.270, 7.620]$, $and\ x_3 \in [15, 25]$. The objective function is formulated in Equation 27.

$$EF_5 = (\pi^2/4)(x_3 + 2)x_2 x_1^2 \tag{27}$$

All the experiments are carried out using the optimal values of the two parameters - α and k. The number of iteration are kept 1000 and the number of runs are 30. The detailed result obtained on the five engineering design problems are given in Table 11. The best value, average value and standard deviation of all runs are also provided in Table 11. The outcomes are compared with six other state-of-the-art methods - PSOGSA, GPS, PSO, GWO, SOGWO and EO. It is to be noted that the proposed GWFA algorithm is able to achieve the lowest value for three out of five engineering problems. For the rest two functions, GWFA has achieved results very close to the global optimum. Thus, this test clearly indicates that GWFA is highly robust in nature and is applicable to a large variety of mathematical optimization problem.

Table 11: Results of GWFA on five standard Engineering Applications along with comparative study that contain methods like - PSOGSA, GPS, PSO, GWO, SOGWO and EO.

Function		GWFA	PSOGSA	GPS	PSO	GWO	SOGWO	EO
EF1	best	2.50E-03	2.50E-03	6.90E-03	2.50E-03	2.50E-03	2.50E-03	2.50E-03
	Avg	2.50E-03	2.50E-03	1.07E-01	2.50E-03	2.50E-03	2.50E-03	2.50E-03
	Std	4.34E-19	8.64E-10	2.50E-03	0.00E+00	0.00E+00	0.00E+00	0.00E+00
EF2	best	1.14E-15	3.60E-03	2.88E-07	0.00E+00	3.05E-17	2.43E-14	0.00E+00
	Avg	9.00E-12	2.00E-03	2.00E-03	0.00E+00	3.05E-17	2.43E-14	0.00E+00
	Std	1.25E-11	4.93E-09	4.00E-03	0.00E+00	0.00E+00	0.00E+00	0.00E+00
EF3	best	7.89E-03	0.00E+00	0.00E+00	7.89E-03	7.89E-03	7.89E-03	7.89E-03
	Avg	7.89E-03	0.00E+00	0.00E+00	7.89E-03	7.89E-03	7.89E-03	7.89E-03
	Std	1.73E-18	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
EF4	best	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Avg	0.00E+00	3.33E+02	8.88E+03	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Std	0.00E+00	4.79E+02	4.48E+04	0.00E+00	0.00E+00	0.00E+00	0.00E+00
EF5	best	1.37E+01	1.38E+01	1.38E+01	1.37E+01	1.37E+01	1.37E+01	1.37E+01
	Avg	1.37E+01	1.38E+01	1.38E+01	1.37E+01	1.37E+01	1.37E+01	1.37E+01
	Std	3.55E-15	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00

5. Conclusion

An optimization algorithm based on a hydro-geological phenomena known as groundwater flow is proposed in this paper. The algorithm, named as GWFA, is inspired from the movement of groundwater from recharge areas to discharge areas following Darcy's law. Exploration and exploitation abilities of the algorithm are carefully taken care of through the usage of control factor. From the experimental outcomes and associated discussion, it has proved to be a very efficient way to perform mathematical optimization. The algorithm has been tested over 23 benchmark functions and 5 classical engineering problems. Comparison with some classical and recently published algorithms proves the superiority of the proposed algorithm. GWFA has been able to outperform the other methods over 12 out of the 23 functions. The convergence test has also

confirmed that GWFA is able to provide fast yet steady convergence. Besides, significant tests are conducted to ensure the statistical significance of the proposed algorithm. Therefore, by analyzing the results and related discussion, it can be concluded that GWFA is comparable to most of the state-of-the-art algorithms and applicable to a variety of optimization problems. In future, we want to modify the model further by hybridization with some other meta-heuristic methods and apply the new model in other areas like feature selection, image contrast enhancement etc.

NOTE: Upon positive response, all the related codes used for GWFA will be made online with proper documentation.

References

[1] D. P. Bertsekas, A. Scientific, Convex optimization algorithms, Athena Scientific Belmont, 2015.

[2] P. M. Pardalos, J. B. Rosen (Eds.), Constrained Global Optimization: Algorithms and Applications, Springer-Verlag, 1987. doi:10.1007/bfb0000035.
URL https://doi.org/10.1007/bfb0000035

[3] E. Elbeltagi, T. Hegazy, D. Grierson, Comparison among five evolutionary-based optimization algorithms, Advanced Engineering Informatics 19 (1) (2005) 43–53. doi:10.1016/j.aei.2005.01.004.
URL https://doi.org/10.1016/j.aei.2005.01.004

[4] M. Cavazzuti, Deterministic optimization, in: Optimization Methods, Springer Berlin Heidelberg, 2012, pp. 77–102. doi:10.1007/978-3-642-31187-1_4.

[5] M.-H. Lin, J.-F. Tsai, C.-S. Yu, A review of deterministic optimization methods in engineering and management, Mathematical Problems in Engineering 2012 (2012) 1–15. doi:10.1155/2012/756023.

[6] V. Lo, Heuristic algorithms for task assignment in distributed systems, IEEE Transactions on Computers 37 (11) (1988) 1384–1397. doi:10.1109/12.8704.

[7] O. H. Ibarra, C. E. Kim, Heuristic algorithms for scheduling independent tasks on nonidentical processors, Journal of the ACM (JACM) 24 (2) (1977) 280–289. doi:10.1145/322003.322011.

[8] I. Boussaïd, J. Lepagnot, P. Siarry, A survey on optimization metaheuristics, Information Sciences 237 (2013) 82–117. doi:10.1016/j.ins.2013.02.041.

[9] F. Glover, Future paths for integer programming and links to artificial intelligence, Computers & Operations Research 13 (5) (1986) 533–549. doi:10.1016/0305-0548(86)90048-1.

[10] M. Khishe, M. Mosavi, Chimp optimization algorithm, Expert Systems with Applications 149 (2020) 113338. doi:10.1016/j.eswa.2020.113338.

[11] B. Chatterjee, T. Bhattacharyya, K. K. Ghosh, P. K. Singh, Z. W. Geem, R. Sarkar, Late acceptance hill climbing based social ski driver algorithm for feature selection, IEEE Access (2020). doi:10.1109/access.2020.2988157.

[12] L. Davis, Handbook of genetic algorithms (1991).

[13] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of ICNN'95 - International Conference on Neural Networks, Vol. 4, 1995, pp. 1942–1948 vol.4. doi:10.1109/ICNN.1995.488968.

[14] M. Dorigo, M. Birattari, T. Stutzle, Ant colony optimization, IEEE Computational Intelligence Magazine 1 (4) (2006) 28–39. doi:10.1109/mci.2006.329691.

[15] M. Gendreau, J.-Y. Potvin, Metaheuristics in combinatorial optimization, Annals of Operations Research 140 (1) (2005) 189–213. doi:10.1007/s10479-005-3971-7.

[16] I. F. Jr., X.-S. Yang, I. Fister, J. Brest, D. Fister, A brief review of nature-inspired algorithms for optimization (2013). arXiv:1307.4186.

[17] M. Abdel-Basset, L. Abdel-Fatah, A. K. Sangaiah, Metaheuristic algorithms: A comprehensive review, in: Computational Intelligence for Multimedia Big Data on the Cloud with Engineering Applications, Elsevier, 2018, pp. 185–231. doi:10.1016/b978-0-12-813314-9.00010-4.

[18] A. F. Nematollahi, A. Rahiminejad, B. Vahidi, A novel meta-heuristic optimization method based on golden ratio in nature, Soft Computing 24 (2) (2019) 1117–1151. doi:10.1007/s00500-019-03949-w.

[19] W. Hillis, Co-evolving parasites improve simulated evolution as an optimization procedure, Physica D: Nonlinear Phenomena 42 (1-3) (1990) 228–234. doi:10.1016/0167-2789(90)90076-2.

[20] X. Xue, M. Yao, R. Cheng, A novel selection operator of cultural algorithm, in: Advances in Intelligent and Soft Computing, Springer Berlin Heidelberg, 2011, pp. 71–77. doi:10.1007/978-3-642-25661-5_10.

[21] J. R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection, MIT Press, Cambridge, MA, USA, 1992.

[22] C. Ryan, J. Collins, M. O. Neill, Grammatical evolution: Evolving programs for an arbitrary language, in: Lecture Notes in Computer Science, Springer Berlin Heidelberg, 1998, pp. 83–96. doi:10.1007/bfb0055930.

[23] D. Simon, Biogeography-based optimization, IEEE Transactions on Evolutionary Computation 12 (6) (2008) 702–713. doi:10.1109/tevc.2008.919004.

[24] H. Salimi, Stochastic fractal search: A powerful metaheuristic algorithm, Knowledge-Based Systems 75 (2015) 1–18. doi:10.1016/j.knosys.2014.07.025.

[25] M. Eusuff, K. Lansey, F. Pasha, Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization, Engineering Optimization 38 (2) (2006) 129–154. doi:10.1080/03052150500384759.

[26] K. M. Passino, Biomimicry of bacterial foraging for distributed optimization and control, IEEE Control Systems Magazine 22 (3) (2002) 52–67. doi:10.1109/MCS.2002.1004010.

[27] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, Journal of Global Optimization 39 (3) (2007) 459–471. doi:10.1007/s10898-007-9149-x.

[28] X.-S. Yang, Firefly algorithms for multimodal optimization, in: Stochastic Algorithms: Foundations and Applications, Springer Berlin Heidelberg, 2009, pp. 169–178. doi:10.1007/978-3-642-04944-6_14.

[29] S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey wolf optimizer, Advances in Engineering Software 69 (2014) 46–61. doi:10.1016/j.advengsoft.2013.12.007.

[30] S. Mirjalili, The ant lion optimizer, Advances in Engineering Software 83 (2015) 80–98. doi:10.1016/j.advengsoft.2015.01.010.

[31] S. Mirjalili, A. Lewis, The whale optimization algorithm, Advances in Engineering Software 95 (2016) 51–67. doi:10.1016/j.advengsoft.2016.01.008.

[32] S. Saremi, S. Mirjalili, A. Lewis, Grasshopper optimisation algorithm: Theory and application, Advances in Engineering Software 105 (2017) 30–47. doi:10.1016/j.advengsoft.2017.01.004.

[33] M. Jain, V. Singh, A. Rani, A novel nature-inspired algorithm for optimization: Squirrel search algorithm, Swarm and Evolutionary Computation 44 (2019) 148–175. doi:10.1016/j.swevo.2018.02.013.

[34] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, H. Chen, Harris hawks optimization: Algorithm and applications, Future Generation Computer Systems 97 (2019) 849–872. doi:10.1016/j.future.2019.02.028.

[35] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, Optimization by simulated annealing, Science 220 (4598) (1983) 671–680. doi:10.1126/science.220.4598.671.

[36] P. J. M. van Laarhoven, E. H. L. Aarts, Simulated annealing, in: Simulated Annealing: Theory and Applications, Springer Netherlands, 1987, pp. 7–15. doi:10.1007/978-94-015-7744-1_2.

[37] E. Rashedi, H. Nezamabadi-pour, S. Saryazdi, GSA: A gravitational search algorithm, Information Sciences 179 (13) (2009) 2232–2248. doi:10.1016/j.ins.2009.03.004.

[38] T. Vicsek, A. Czirók, E. Ben-Jacob, I. Cohen, O. Shochet, Novel type of phase transition in a system of self-driven particles, Physical Review Letters 75 (6) (1995) 1226–1229. doi:10.1103/physrevlett.75.1226.

[39] Z. W. Geem, J. H. Kim, G. Loganathan, A new heuristic optimization algorithm: Harmony search, SIMULATION 76 (2) (2001) 60–68. doi:10.1177/003754970107600201.

[40] A. Hatamlou, Black hole: A new heuristic optimization approach for data clustering, Information Sciences 222 (2013) 175–184. doi:10.1016/j.ins.2012.08.023.

[41] S. Mirjalili, SCA: A sine cosine algorithm for solving optimization problems, Knowledge-Based Systems 96 (2016) 120–133. doi:10.1016/j.knosys.2015.12.022.

[42] S. Mirjalili, S. M. Mirjalili, A. Hatamlou, Multi-verse optimizer: a nature-inspired algorithm for global optimization, Neural Computing and Applications 27 (2) (2015) 495–513. doi:10.1007/s00521-015-1870-7.

[43] A. H. Kashan, R. Tavakkoli-Moghaddam, M. Gen, Find-fix-finish-exploit-analyze (f3ea) meta-heuristic algorithm: An effective algorithm with new evolutionary operators for global optimization, Computers & Industrial Engineering 128 (2019) 192–218. doi:10.1016/j.cie.2018.12.033.

[44] R. Rao, V. Savsani, D. Vakharia, Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems, Computer-Aided Design 43 (3) (2011) 303–315. doi:10.1016/j.cad.2010.12.015.

[45] T. Ray, K. Liew, Society and civilization: an optimization algorithm based on the simulation of social behavior, IEEE Transactions on Evolutionary Computation 7 (4) (2003) 386–396. doi:10.1109/tevc.2003.814902.

[46] A. H. Kashan, League championship algorithm (LCA): An algorithm for global optimization inspired by sport championships, Applied Soft Computing 16 (2014) 171–200. doi:10.1016/j.asoc.2013.12.005.

[47] Y. Tan, Y. Zhu, Fireworks algorithm for optimization, in: Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2010, pp. 355–364. doi:10.1007/978-3-642-13495-1_44.

[48] A. Kaveh, Tug of war optimization, in: Advances in Metaheuristic Algorithms for Optimal Design of Structures, Springer International Publishing, 2016, pp. 451–487. doi:10.1007/978-3-319-46173-1_15.

[49] R. Moghdani, K. Salimifard, Volleyball premier league algorithm, Applied Soft Computing 64 (2018) 161–185. doi:10.1016/j.asoc.2017.11.043.

[50] D. Wolpert, W. Macready, No free lunch theorems for optimization, IEEE Transactions on Evolutionary Computation 1 (1) (1997) 67–82. doi:10.1109/4235.585893.

[51] P. GLEICK, Water resources, Encyclopedia of climate, weather 2 (1996) 817–823.
URL https://ci.nii.ac.jp/naid/10030346693/en/

[52] H. P. G. Darcy, The public fountains of the city of Dijon. Exposure and application of principles ‘a to follow and formulas à to use in questions of water distribution, etc, V. Dalamont, 1856.

[53] X. Yao, Y. Liu, G. Lin, Evolutionary programming made faster, IEEE Transactions on Evolutionary Computation 3 (2) (1999) 82–102. doi:10.1109/4235.771163.

[54] F. Wilcoxon, Individual comparisons by ranking methods, in: Springer Series in Statistics, Springer New York, 1992, pp. 196–202. doi:10.1007/978-1-4612-4380-9_16.

[55] J. Derrac, S. García, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, Swarm and Evolutionary Computation 1 (1) (2011) 3 – 18. doi:https://doi.org/10.1016/j.swevo.2011.02.002.

[56] M. Friedman, The use of ranks to avoid the assumption of normality implicit in the analysis of variance, Journal of the American Statistical Association 32 (200) (1937) 675–701. doi:10.1080/01621459.1937.10503522.

[57] W. H. Kruskal, W. A. Wallis, Use of ranks in one-criterion variance analysis, Journal of the American Statistical Association 47 (260) (1952) 583–621. doi:10.1080/01621459.1952.10483441.