Developing Four Metaheuristic Algorithms for Multiple-Objective Management of Groundwater

H.A. El-Ghandour¹ and E. Elbeltagi²

1. Associate Professor, Irrigation & Hydraulics Dept., Fac. of Engrg., Mansoura Univ., Mansoura 35516, Egypt.

Corresponding author: hamdy_a@mans.edu.eg

http://dx.doi.org/10.22115/SCCE.2018.128344.1057

ARTICLE INFO

Article history:
Received: 23 April 2018
Revised: 19 June 2018
Accepted: 20 June 2018

Keywords:
Genetic algorithms,
Memetic algorithms,
Particle swarm,
Shuffled frog leaping,
Compromise solution,
Multiple objectives optimization.

ABSTRACT

Groundwater is one of the important sources of freshwater and accordingly, there is a need for optimizing its usage. In this paper, four multi-objective metaheuristic algorithms with new evolution strategy are introduced and compared for the optimal management of groundwater namely: Multi-objective genetic algorithms (MOGA), multi-objective memetic algorithms (MOMA), multi-objective particle swarm optimization (MOPSO), and multi-objective shuffled frog leaping algorithm (MOSFLA). The suggested evolution process is based on determining a unique solution of the Pareto solutions called the Pareto-compromise (PC) solution. The advantages of the current development stem from: 1) The new multiple objectives evolution strategy is inspired from the single objective optimization, where fitness calculations depend on tracking the PC solution only through the search history; 2) a comparison among the performance of the four algorithms is introduced. The development of each algorithm is briefly presented. A comparison study is carried out among the formulation and the results of the four algorithms. The developed four algorithms are tested on two multiple-objective optimization benchmark problems. The four algorithms are then used to optimize two-objective groundwater management problem. The results prove the ability of the developed algorithms to accurately find the Pareto-optimal solutions and thus the potential application on real-life groundwater management problems.
1. Introduction

Groundwater is considered as one of the freshwater sources needed to serve human activities, such as drinking, industrial, domestic and irrigation. Due to irregular pumping of groundwater, serious environmental hazards have been occurred including groundwater levels decline and interference among wells.

To pump the maximum amount of groundwater without aquifer depletion and achieving minimum operation cost, powerful simulation-optimization models have to be applied to obtain the best strategy. These models are adopted to deal with both the design and operation problems corresponding to remediation, water supply and controlling of groundwater flow [1]. Optimization models can be classified according to their objectives into single objective and multiple objectives. In single objective optimization, a unique optimal solution is produced related to the maximization or minimization of a single objective [1 – 19]. In multiple objectives optimization, a group of non-dominated solutions, called Pareto-front (PF), are usually produced [20 – 25]. Optimization models are classified into deterministic and metaheuristic. Deterministic methods include linear, non-linear and dynamic programming [14 – 17, 26 – 33]. While, metaheuristic algorithms are stochastic optimization techniques that mimic nature perception, learning and reasoning such as: Genetic Algorithms (GA), Memetic Algorithms (MA), Particle Swarm (PSO) and Shuffled Frog Leaping (SFLA) [1 – 13, 18 – 25].

Deterministic optimization methods tend to become overly complicated and require massive computations when dealing with complex optimization problems with numerous constraints and decision variables, such as real-life water related optimization problems. While, metaheuristic optimization algorithms are usually used due to their direct getting solutions without the need for gradients and initial solutions. Simulation models are coupled with optimization techniques to evaluate proposed objective function(s). Simulation models apply either numerical or analytical approaches to simulate groundwater flow and calculate the hydraulic heads of the studied aquifer. Numerical approaches include finite element (FEM), finite difference and boundary element methods while, analytical approaches use analytical element method and Fourier series [7 – 8, 10].

The development made in the current paper is different from previous works in many aspects. The use of a new evolution strategy for generating the Pareto optimal solutions in four different multi-objectives algorithms namely: MOGA, MOMA, MOPSO and MOSFLA. Using such evolution strategy resembles solving single optimization problems. Performing a comparison study among the four multi-objectives optimization algorithms for optimizing ground water management problems. While, all previous comparisons have been performed among single objective optimization algorithms [34 – 35].

2. New evolution strategy

Multiple objectives optimization problems are characterized by the existence of many optimal solutions (non-dominated) called Pareto-Front (PF). Many different algorithms are used to solve
multi-objectives optimization problems. Grierson [36] proposed a methodology to determine a single Pareto-Compromise (PC) solution located in the PF which fairly satisfies all objectives. This research adopts such PC solution for the fitness calculations and then driving the evolution process in MOGA, MOMA, MOPSO, and MOSFLA algorithms.

The proposed new evolutionary strategy which is based on finding the PC solution is adopted due to their proven effectiveness in terms of the processing time and number of Pareto solutions obtained in the PF, as stated by Elbeltagi et al. [37]. Also, the proposed multi-objective evolutionary strategy is reminiscent of single-objective optimization, where its fitness assignment is based on tracking a single evolving solution over the search history. In addition, knowing the PC solution helps decision makers to choose a single solution that uniquely represents a mutually agreeable trade-off between all competing objectives for the problem.

The suggested methodology for developing each algorithm to handle multiple objective problems depends mainly on the principle of non-dominant solutions located in more densely populated regions in the objective space having lower probabilities to be selected. Consequently, better distributions of optimal solutions can occur in the PF. The following steps illustrate the suggested methodology:

1. All solutions in an evolutionary cycle are evaluated according to the values of their objectives to determine a set of non-dominated solutions called PF. Each non-dominated solution belongs to the PF can beat all other solutions in at least one objective.
2. For each obtained PF, the PC solution is determined mathematically as given by Grierson [36].
3. The distance \( d_{PC,k} \) between the PC and each Pareto optimal solution \((k)\) is calculated based on Eq. 1:

\[
d_{PC,k} = \sqrt{\sum_{i=1}^{n} (f_{PCi} - f_{ki})^2} \quad k = 1, 2, 3, \ldots, N
\]

in which, \( f_{PCi} \): the objective function \( i \) value corresponding to the PC, \( f_{ki} \): the objective function \( i \) value corresponding to solution \((k)\) located in the PF, \( N \): number of Pareto optimal solutions located in the PF, and \( n \): number of objective functions.

4. The fitness of each Pareto-optimal solution \((k)\), Eq. 2, equals its distance \( d_{PC,k} \) from the PC solution. This step ensures the disperse of the solutions along the solution space and prevents solutions crowdies.

\[
Fitness_k = d_{PC,k} \quad k = 1, 2, 3, \ldots, N
\]

5. The relative fitness of each solution in the PF is then calculated as follows:
Relative Fitness\(_k\) = \frac{\text{Fitness}_k}{\sum_{k=1}^{N} \text{Fitness}_k} \quad (3)

6. To form the next generation, solutions from the current PF are randomly selected based on their Relative Fitness. Accordingly, Pareto optimal solutions with larger values of Euclidean distance have the higher probability to be selected.

3. Descriptions of the proposed metaheuristic algorithms

A brief description of the MOGA, MOMA, MOPSO, MOSFLA algorithms are presented in the following subsections.

3.1. Multiple objectives genetic algorithms

GA simulates mechanisms of natural principle of the survival of the fittest. In GA, solutions are represented using chromosomes containing a group of genes, having values for the variables of an optimization problem. Each chromosome is, then, evaluated to determine its fitness using specific objective function. Then, best chromosomes are exchanging their information through different GA operation (i.e. selection, crossover and mutation), as follows, to produce offspring chromosomes:

- **Selection**: Fit chromosomes are likely to be selected to pass to next generations.
- **Crossover**: Parent chromosomes are exchanging their genetic information to produce offspring chromosomes.
- **Mutation**: It allows for random change of genetic information of individual genes. This process introduces new genetic information to the current chromosomes, this may lead to avoid stagnation and pre-mature convergence.

The GA operators are continued for large number of generations until an optimal or near-optimal solution is obtained. The parameters that affect the performance of the GA are: number of chromosomes, number of generations, crossover rate, mutation rate and crossover type. The probability of obtaining a global optimum solution is achieved by increasing the numbers of both population and generations but, this substantially increases processing time.

In the proposed MOGA, for each generation, the steps from (1) to (5) mentioned in the previous section are carried out then, step (6) is continued to produce offspring chromosomes through crossover and mutation operators until a new population of chromosomes is formed. Also, a replacement strategy is carried out to replace the weakest chromosome in the current generation with a randomly selected one located in the PF of the previous generation. These processes are repeated for large number of evolutionary cycles until an optimum or near-optimal solution is reached. Figure 1 shows a flow chart of the proposed MOGA.
Fig. 1. Flowchart describing both the MOGA and MOMA algorithms.
3.2. Multiple objectives memetic algorithm

Similar to the GA, MA chromosome’s elements are called memes, not genes. In MA, chromosomes are allowed to perform local search first, before being involved in the global evolutionary search [38]. Similar to GA, a population of chromosomes is, initially, generated randomly. Then, each generated solution is subjected to a local search to improve its fitness. Afterwards, the GA operators are applied, to produce new individuals. These new individuals are then performing some local search to maintain local optimality. In this paper, the local search is carried out through a pair-wise interchange (or in other words, swapping chromosome’s memes) as shown in Fig. 2, [38].

![Diagram of meme search](image)

**Fig. 2.** Applying local search using pair-wise interchange (Elbeltagi et al. 2005).

The maximum number of swaps is limited to 10 for each chromosome in order to maintain reasonable processing time. After swapping, if the chromosome’s fitness improves, the change is kept; otherwise, ignore the change. As previously mentioned, the MA parameters are the same as the parameters of the GA, in addition to the local-search mechanism.

The proposed MOMA is the same as the MOGA in addition to the local search previously mentioned. After performing each swap, a chromosome before change (ch\_before) is compared with its mate after (ch\_after) the swaps, as follows:

- If the ch\_after dominates the ch\_before, then the change is kept.
- If the ch\_before dominates the ch\_after, then the change is ignored and swaps are repeated for a specified number of times (10 times).
- If no domination occurs, then one of the two chromosomes is randomly chosen as the new chromosome.

Figure 1 shows the flow chart of the MOMA.

3.3. Multiple objectives particle swarm

The PSO was initially developed by Kennedy and Eberhart [39]. In single objective PSO, a set of random particles (solutions) is initially generated to begin the evolution process. Through such process, three values are monitored by each particle $i$: its current location $X_i(I)$, the best position it arrived $P_i(I)$, and its velocity of flying $V_i(I)$ where, $I$ is the current evolution cycle.

In each evolutionary cycle, the position of best particle ($P_g$) is determined which is considered the best fitness for all particles. Consequently, the velocity of each particle is updated as given by Shi and Eberhart [40]:
\[ V_i(I+1) = \omega \times V_i(I) + C_1 \times r_1 \times [P_i(I) - X_i(I)] + C_2 \times r_2 \times [P_g(I) - X_i(I)] \] 

\[ V_{\text{max}} \leq V_i(I+1) \leq -V_{\text{max}}, \ i = 1, \ldots , n_p \] 

Each particle position is updated as follows (Eq. 5):

\[ X_i(I+1) = X_i(I) + V_i(I+1) \] 

in which, \( V_i(I+1) \): particle \( i \) velocity in iteration \((I+1)\); \( V_i(I) \): particle \( i \) velocity in iteration \((I)\); \( P_i(I) \): particle \( i \) best position in iteration \((I)\); \( P_g(I) \): the global best particle position in iteration \((I)\); \( X_i(I) \): particle \( i \) position in iteration \((I)\); \( X_i(I+1) \): particle \( i \) position in iteration \((I+1)\); \( \omega \): constant called inertia weight; \( C_1 \) and \( C_2 \): learning factors \( (C_1 = C_2 = 2) \); \( r_1 \) and \( r_2 \): two random numbers (ranging from 0 to 1); \( V_{\text{max}} \): the maximum value of velocity change.

The main PSO parameters are: the population size; number of cycles; \( V_{\text{max}} \); \( C_1 \); \( C_2 \); and \( \omega \).

In the suggested MOPSO, both Eqs. 4 and 5 are, also, adopted but, the two parameters \( P_i \) and \( P_g \) need to be determined repetitively during each cycle [41]. Initially, \( P_i \) represents particle \( i \) initial position which updated in the subsequent iterations as given by Baltar and Fontane [42] for each particle \( i \):

- If the current position \( P_i \) dominates the new position, then the new \( P_i \) is set as the current \( P_i \).
- If the current \( P_i \) is dominated by the new position, then the new position is set the new \( P_i \).
- If no domination occurs, then the new \( P_i \) is selected randomly from the two positions.

In the MOPSO, there is no best particle position \( (P_g) \) for all particles, but there are several equally good particles positions \( (PF) \) in each evolution cycle stored in the external repository. For each evolution cycle, the steps from (1) to (6) mentioned in the previous section are carried out to select one solution from the stored ones which is considered as \( P_g \). The flow chart of the MOPSO is shown in Fig. 3.

3.4. Multiple objectives shuffled frog leaping

SFLA combines the advantages of both MA and PSO algorithms. In the single objective SFLA, the evolution process is initialized with a number of memeplexes, each contains a number of frogs (individuals). Throughout each memeplex, two individuals (frogs) are determined based on their fitness: the best individual \( (X_B) \) and the worst individual \( (X_W) \). Also, in each cycle of evolution, the individual with the highest global fitness \( (X_G) \) is determined. Then, the individual with the worst fitness is improved in each cycle (not all frogs) through a process similar to the PSO. Accordingly, the change in worst frog position \( (D_i) \) and its new position \( (X_{W\text{new}}) \) are calculated as follows [38, 43]:

\[ D_i = C \times r \times (X_B - X_W) \] 

\[ X_{W\text{new}} = X_{W\text{current}} + D_i, \quad D_{\text{max}} \geq D_i \geq -D_{\text{max}} \]
in which, $C$: the search acceleration factor; $r$: a random number in the range $[0, 1]$; and $D_{\text{max}}$: the maximum allowed frog’s position change.

**Fig. 3.** Flowchart describing the MOPSO algorithm.
If a better individual (solution) is produced when applying Eqs. 6 and 7, the worst frog is replaced by the new produced one. Otherwise, Eqs. 6 and 7 is repeatedly calculated but $X_B$ is exchanged with the global best frog $X_G$. If no improvement is occurred, a new frog (solution) is created randomly replacing the worst one. This process continues for a given number of iterations [38].

The SFLA parameters are: number of frogs; number of memeplexes; number of trials in each memeplex; number of iterations; $D_{max}$; and $C$.

In the proposed MOSFLA, there are several solutions for $(X_B)$ and several other solutions for $(X_W)$ in each memeplex. Consequently, both the $(X_B)$ and $(X_W)$ are randomly selected from the corresponding solutions. Also, there is no global best frog $(X_G)$ for all frogs, but there are several equally good frogs’ positions (PF) in each evolution cycle stored in the external repository. In this case, $X_G$ is determined by applying the steps from (1) to (6) mentioned in the previous section in each evolution cycle. Figure 4 shows the flow diagram of the MOSFLA algorithm.

4. Models implementation and verification

To facilitate the implementation of the proposed four algorithms, each algorithm is coded by the authors using the FORTRAN language. Accordingly, all parameters related to each algorithm are studied. The most common parameters presented in the literature are studied here in the current development. Then, the proposed algorithms’ performance is compared using two multiple objectives benchmark problems (Appendix A) where the obtained PFs are compared with the corresponding known true PFs of the two benchmark problems.

Selection of Parameters’ values for each algorithm is an essential part of this study. As such, a large number of trials are performed to obtain the most suitable parameters’ values that suit the two test problems and the example application (presented later). In this step, initial parameters’ values are set based on relevant literature. Then, parameters’ values are changed through many experiments while the results are monitored. The final parameters’ values adopted for each model are presented in Table 1. Also, different termination criteria have been experimented with including the solution convergence and number of evolutionary cycles. Based on many trials and experiments, the termination criterion has been set based on the number of evolutionary cycles (iteration-oriented) as presented in Table 1.

For the purpose of comparing the proposed algorithms performance, different metrics are used, namely: the generational distance ($G_d$); the distance between the $PC$ solution of the obtained PF and the true PF; the processing time; the number of Pareto-optimal solutions located on the PF and the mean square error between the obtained compromise solution and the nearest best alternative Pareto optimal solution. The values of all these metrics are calculated and presented in Table 2.
Fig. 4. Flowchart describing the MOSFLA algorithm.
Table 1.
Parameters’ values used in the two test problems and the example application.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Parameter names</th>
<th>Test problem (1): Kita function</th>
<th>Test problem (2): Kursawe function</th>
<th>Example application</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOGA</td>
<td>- Number of chromosomes: 200</td>
<td>100</td>
<td>Uniform</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Number of generations: 1000</td>
<td>500</td>
<td>Uniform</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>- Crossover ratio: 0.9</td>
<td>0.9</td>
<td>Uniform</td>
<td>0.8</td>
</tr>
<tr>
<td></td>
<td>- Mutation ratio: 0.2</td>
<td>0.2</td>
<td>Uniform</td>
<td>0.05</td>
</tr>
<tr>
<td></td>
<td>- Crossover type: Uniform</td>
<td>Uniform</td>
<td>Uniform</td>
<td></td>
</tr>
<tr>
<td>MOMA</td>
<td>- Number of chromosomes: 200</td>
<td>100</td>
<td>Uniform</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>- Number of generations: 1000</td>
<td>500</td>
<td>Uniform</td>
<td>300</td>
</tr>
<tr>
<td></td>
<td>- Crossover rate: 0.9</td>
<td>0.9</td>
<td>Uniform</td>
<td>0.8</td>
</tr>
<tr>
<td></td>
<td>- Mutation rate: 0.2</td>
<td>0.1</td>
<td>Uniform</td>
<td>0.05</td>
</tr>
<tr>
<td></td>
<td>- Crossover type: Uniform</td>
<td>Uniform</td>
<td>Uniform</td>
<td></td>
</tr>
<tr>
<td>MOPSO</td>
<td>- Population size: 100</td>
<td>100</td>
<td>Linear [1.4-0.9]</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>- Number of cycles: 500</td>
<td>500</td>
<td>Linear [1.4-0.9]</td>
<td>300</td>
</tr>
<tr>
<td></td>
<td>- The maximum velocity $V_{max}$: 0.35</td>
<td>0.6</td>
<td>Linear [1.4-0.9]</td>
<td>5500</td>
</tr>
<tr>
<td></td>
<td>- Values of learning factors $C_1$ and $C_2$: 2</td>
<td>2</td>
<td>Linear [1.4-0.9]</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>- Value of inertia weight $\omega$: Linear [1.4-0.9]</td>
<td>Linear [1.4-0.9]</td>
<td>Linear [1.3-0.9]</td>
<td></td>
</tr>
<tr>
<td>MOSFLA</td>
<td>- Number of frogs: 200</td>
<td>200</td>
<td>200</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Number of memeplexes: 20</td>
<td>20</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Number of trials before shuffling: 30</td>
<td>30</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Number of iterations after shuffling: 100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- The maximum change in frog position $D_{max}$: 7</td>
<td>10</td>
<td>7000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- The search acceleration factor $C$: 2.4</td>
<td>1.7</td>
<td>4.2</td>
<td></td>
</tr>
</tbody>
</table>

The $G_d$ metric [44] is considered to test the performance of the developed four models for coping with multiple objectives problems. The $G_d$ metric is used to measure the distance between the obtained and the true PFs:

$$G_d = \sqrt{\frac{1}{N} \sum_{i=1}^{N} d_i^2}$$

(8)

in which, $d_i$: the Euclidean distance between the non-dominated solution $i$ located in the obtained PF and the nearest one in the true PF and $N$ is the number of non-dominated solutions located in the obtained PF. A very small value for $G_d$ metric means that all the generated non-dominated solution nearly coincides with the true PF.
Table 2.
Comparison among algorithms’ results for the two test problems.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>metric</th>
<th>Test problem (1): Kita function</th>
<th>Test problem (2): Kursawe function</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOGA</td>
<td>$G_d$</td>
<td>0.00076</td>
<td>0.00159</td>
</tr>
<tr>
<td></td>
<td>Dist.</td>
<td>0.1843</td>
<td>0.1196</td>
</tr>
<tr>
<td></td>
<td>Processing time (second)</td>
<td>2.917</td>
<td>1.451</td>
</tr>
<tr>
<td></td>
<td>Compromise solution</td>
<td>(2.79, 8.11)</td>
<td>(-16.7, -4.72)</td>
</tr>
<tr>
<td></td>
<td>Best alternative</td>
<td>(2.63, 8.12)</td>
<td>(-16.69, -4.64)</td>
</tr>
<tr>
<td></td>
<td>MSE</td>
<td>0.001733</td>
<td>0.000163</td>
</tr>
<tr>
<td></td>
<td>No. of PF solutions</td>
<td>187</td>
<td>419</td>
</tr>
<tr>
<td>MOMA</td>
<td>$G_d$</td>
<td>0.00086</td>
<td>0.00051</td>
</tr>
<tr>
<td></td>
<td>Dist.</td>
<td>0.2951</td>
<td>0.0150</td>
</tr>
<tr>
<td></td>
<td>Processing time (second)</td>
<td>20.389</td>
<td>3.603</td>
</tr>
<tr>
<td></td>
<td>Compromise solution</td>
<td>(2.78, 8.11)</td>
<td>(-16.71, -4.7)</td>
</tr>
<tr>
<td></td>
<td>Best alternative</td>
<td>(3.11, 8.09)</td>
<td>(-16.72, -4.77)</td>
</tr>
<tr>
<td></td>
<td>MSE</td>
<td>0.007058</td>
<td>0.000089</td>
</tr>
<tr>
<td></td>
<td>No. of PF solutions</td>
<td>195</td>
<td>547</td>
</tr>
<tr>
<td>MOPSO</td>
<td>$G_d$</td>
<td>0.00064</td>
<td>0.00326</td>
</tr>
<tr>
<td></td>
<td>Dist.</td>
<td>0.061</td>
<td>0.0114</td>
</tr>
<tr>
<td></td>
<td>Processing time (second)</td>
<td>1.514</td>
<td>0.452</td>
</tr>
<tr>
<td></td>
<td>Compromise solution</td>
<td>(2.77, 8.11)</td>
<td>(-16.72, -4.73)</td>
</tr>
<tr>
<td></td>
<td>Best alternative</td>
<td>(2.75, 8.11)</td>
<td>(-16.72, -4.74)</td>
</tr>
<tr>
<td></td>
<td>MSE</td>
<td>0.000018</td>
<td>0.000002</td>
</tr>
<tr>
<td></td>
<td>No. of PF solutions</td>
<td>200</td>
<td>195</td>
</tr>
<tr>
<td>MOSFLA</td>
<td>$G_d$</td>
<td>0.00102</td>
<td>0.00290</td>
</tr>
<tr>
<td></td>
<td>Dist.</td>
<td>0.0051</td>
<td>0.0301</td>
</tr>
<tr>
<td></td>
<td>Processing time (second)</td>
<td>4.477</td>
<td>1.763</td>
</tr>
<tr>
<td></td>
<td>Compromise solution</td>
<td>(2.78, 8.11)</td>
<td>(16.71, -4.79)</td>
</tr>
<tr>
<td></td>
<td>Best alternative</td>
<td>(2.81, 8.11)</td>
<td>(-16.71, -4.78)</td>
</tr>
<tr>
<td></td>
<td>MSE</td>
<td>0.000061</td>
<td>0.000004</td>
</tr>
<tr>
<td></td>
<td>No. of PF solutions</td>
<td>169</td>
<td>179</td>
</tr>
</tbody>
</table>

The graphical comparison between the obtained PFs, associated with each proposed model, and the true PF is presented in Fig. 5. It can be seen that, the obtained PFs seem regular and distributed throughout the solution space. The $G_d$ value is found to be 0.00076, 0.00086, 0.00064, and 0.00102 for MOGA, MOMA, MOPSO, and MOSFLA respectively. Consequently, the obtained PFs are nearly similar to the true PF as per the very small values obtained of the $G_d$. Also, the minimum $G_d$ value obtained is associated with the MOPSO algorithm.

Figure 6, on the other hand, shows the obtained PFs for the proposed four models along with their corresponding true PF for the second benchmark problem. As noticed in the previous test problem, the obtained PFs seem distributed regularly all over the solution space. The $G_d$ values are calculated as: 0.00159, 0.00051, 0.00326, and 0.00290 for MOGA, MOMA, MOPSO, and MOSFLA models respectively. Then, the obtained PFs are nearly similar to the true PF as per the very small distances of $G_d$ metric, with the best value associated with the MOMA.
Fig. 5. PFs produced by the four proposed algorithms and the true PF for the Kita function.

The distance between the compromise solution of the obtained PF and the corresponding one of the true PF is calculated according to Eq. 9 and listed in Table 2 for each studied algorithm. The minimum distance reported is for the MOPSO with the second test problem and for the MOSFLA with the first test problem.

$$\text{Dist.} = \sqrt{(x_{\text{true}} - x_{\text{obt}})^2 + (y_{\text{true}} - y_{\text{obt}})^2}$$

(9)

in which, \(x_{\text{true}}\) and \(y_{\text{true}}\): the coordinates of the compromise solution corresponding to the true PF and \(x_{\text{obt}}\) and \(y_{\text{obt}}\): the coordinates of compromise solution corresponding to the obtained PF.

Also, the Mean Square Error (MSE) is calculated based on the values of the criteria \(f_j^0\) for the PC solution and the values of the corresponding criteria \(f_{ij}^*\) for each of the \(N\) Pareto-optimal solutions (Eq. 10) [36]:

$$\text{MSE} = \frac{1}{n} \times \sum (1 - f_{ij}^*/f_j^0)^2; (i = 1, \ldots, n)$$

(10)
where, \( n \): number of criteria (i.e. objective functions). The solution that has the smallest \( MSE \) value is considered as the best alternative solution of the \( PC \) solution.

![Fig. 6. PFs produced by the four proposed algorithms and the true PF for the Kursawe function.](image)

All codes are running on a personal computer of Intel (R) Core (TM) I3 CPU and having 4 GB Ram. The least processing time reported with the MOPSO for both test problems. The MOMA outperforms all other algorithms for the obtained number of solutions located on the PF for the second test problem, while the MOPSO outperforms other algorithms for the same metric in the first test problem. Generally, the results show that the MOPSO outperforms other algorithms in most of the comparison criteria as presented in Table 2.

The results of the previous two test problems show that, the obtained PFs corresponding to the four models are very close to the true PFs. Also, both the rapid convergence and good results are obtained in the two cases. Accordingly, the proposed four models having the ability for dealing with problems of multiple-objective functions.

5. Example application

In this section, the four proposed models with the new evolution strategy are applied on a popular groundwater management problem in order to maximize the pumping rates and
minimize the operation costs [24]. Also, a comparative study among the algorithms’ results is carried out. Both the plan of the studied aquifer and its cross-sectional elevation along with the required data are shown in Fig. 7. The aquifer is formed of sand and gravel with homogeneous and isotropic porous medium. The hydraulic conductivity ($K$) equals 50 m/day and it is assumed that the aquifer is subjected to uniform rainfall ($W$) of 0.001 m/day. The objective of this problem is to determine the pumping rates from the pre-specified system of the ten wells, achieving hydraulic constraints. The mathematical formulation of the optimization problem is as follows [6, 24]:

$$f_1 = \max \left[ \sum_{i=1}^{N_w} Q_i - P(h) \right]$$

$$f_2 = \min \left\{ \sum_{i=1}^{N_w} \left[ a_1 y(d_i)^b + a_2 (Q_i)^{b_1} (d_i - h_i)^{b_2} + a_3 Q_i (d_i - h_i) + P(h) + P(Q) \right] \right\}$$

Subjected to:

$$\sum_{i=1}^{N_w} Q_i \geq Q_{req}$$

$$h_i \geq h_{i, min}, \quad i = 1, 2, 3, \ldots, N_W$$

$$Q_{i, min} \leq Q_i \leq Q_{i, max}, \quad i = 1, 2, 3, \ldots, N_W$$

$$k = \begin{cases} 1 & \text{if } Q_i \neq 0 \\ 0 & \text{if } Q_i = 0 \end{cases} \quad i = 1, 2, 3, \ldots, N_W$$

$$P(h) = \begin{cases} h_{i, min} - h_i & \text{if } h_i < h_{i, min} \\ 0 & \text{if } h_i \geq h_{i, min} \end{cases} \quad i = 1, 2, 3, \ldots, N_W$$

$$P(Q) = \begin{cases} Q_{req} - \sum Q & \text{if } \sum Q < Q_{req} \quad i = 1, 2, 3, \ldots, N_W \\ 0 & \text{if } \sum Q > Q_{req} \end{cases}$$

where, $Q_i$: well $i$ pumping rate; $N_W$: number of wells; $P(h)$: head violation penalty; $h_{i, min}$: minimum head allowable at well $i$ ($= 0$); $Q_{i, min}$ and $Q_{i, max}$: well $i$ minimum and maximum allowable pumping rates (equals 0 and 7000 m$^3$/day respectively); $a_1$, $a_2$, and $a_3$: cost coefficients for drilling, pumping equipment and operation of well respectively ($a_1 = 4221$ S/m, $a_2 = 0.12$ S/m$^4$, $a_3 = 0.03$ S/m$^4$); $y$: the penalty term; $b_1$, $b_2$, and $b_3$: constants indicating economies of scale ($b_1 = 0.299$ and $b_2 = b_3 = 1$); $Q_{req}$: the water demand required ($= 30000$ m$^3$/day); $d_i$: the depth of well $i$; and $P(Q)$: penalty term for the minimum pumping violation.
In this application, a simulation model, based on FEM, is linked with each proposed optimization algorithm to predict the hydraulic head at each well location shown in the plan, Fig. 7. FEM is used to discretize the studied plan to number of linear rectangular elements and the two-dimensional flow equation, Eq. 19 is solved over each element [25].

\[
K \frac{\partial^2 \phi}{\partial x^2} + K \frac{\partial^2 \phi}{\partial y^2} = W - \sum_{i=1}^{N_W} Q_i \delta(x-x_i) \delta(y-y_i)
\]  \hspace{1cm} (19)

in which, \(K\): the hydraulic conductivity of aquifer; \(\phi\): the potential (= \(h^2/2\)); \(h\): the hydraulic head; \(W\): the uniform rainfall or uniform evaporation; \(Q_i\): the rate of injection or pumping for the well \(i\); \(\delta(z)\): the Dirac delta function which equals 1 if \(z\) equal zero otherwise equals zero; and \(N_W\): number of field wells.

In the simulation model, Eq. 19 is written as follows:

\[
[A] \times [\phi] = [F]
\]  \hspace{1cm} (20)
in which, $A$: the conductance/stiffness matrix, $\varphi$: the unknown vector of potentials and $F$: the load vector which contains the external source or sink of water and flux concentration.

In order to make a comparison among the obtained PFs of the developed four models, in addition to the metrics described before, the Spacing ($S_P$) metric [44] is considered, Eq. 21. The $S_P$ is used to measure the spread of the Pareto optimal solutions over the whole solution space.

$$S_P = \sqrt{\frac{1}{N-1} \sum_{i=1}^{N} (\bar{d} - d_i)^2}$$

(21)

in which, $d_i$: the minimum distance between Pareto-solution $i$ and all other solutions located in the PF, $\bar{d}$: the arithmetic mean of $d_i$, and $N$: the number of solutions located in the PF.

![Graphs showing PFs for MOGA, MOMA, MOPSO, and MOSFLA models](image)

**Fig. 8.** PFs for the example application produced by the developed four algorithms

A small value of $S_P$ means that solutions located in the PF are equidistantly spaced. Figure 8 illustrates the obtained PFs associated with each developed model. As presented in Table 3, the $S_P$ metric is computed for each algorithm as 231.6, 2842.6, 129.3, and 479.9 for MOGA, MOMA, MOPSO, and MOSFLA respectively. The $S_P$ value for MOPSO model is the minimum then, the performance of this model is the best compared with the other models. Accordingly, the obtained PF from MOPSO model is more regular and distributed throughout the solution space.
Also, the processing time of MOPSO model (138.97 sec) is found to be the minimum compared with the processing time of the other models. Also, the highest number of Pareto optimal solutions obtained was with the MOPSO. From the visual inspection of Fig. 8, the PFs obtained using MOGA and MOMA models are more uniformly distributed compared with the other two models. The Pareto compromise solution for each obtained PF is, also, determined as shown in Fig. 8. It is noticed that, the deviation among the values of compromise solutions is small. In general, the results showed the effectiveness of using the suggested evolution strategy in each model to drive the final PF for the example application. Consequently, the proposed four models can be used to deal with groundwater real life applications having multiple objectives.

Table 3.
Comparison among algorithms results for the example application.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>$S_p$ (sec)</th>
<th>Processing time (second)</th>
<th>Compromise solution</th>
<th>Best alternative</th>
<th>MSE</th>
<th>No. of PF solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOGA</td>
<td>231.6</td>
<td>174.408</td>
<td>(45094.6, 186891.0)</td>
<td>(45077.5, 186944.6)</td>
<td>0.000000114</td>
<td>1221</td>
</tr>
<tr>
<td>MOMA</td>
<td>2842.6</td>
<td>745.166</td>
<td>(45792.4, 189431.5)</td>
<td>(45868.7, 189665.6)</td>
<td>0.000000215</td>
<td>1551</td>
</tr>
<tr>
<td>MOPSO</td>
<td>129.3</td>
<td>138.965</td>
<td>(45582.2, 189013.0)</td>
<td>(45625.6, 189477.3)</td>
<td>0.000000347</td>
<td>3968</td>
</tr>
<tr>
<td>MOSFLA</td>
<td>479.9</td>
<td>593.794</td>
<td>(44672.6, 184592.2)</td>
<td>(44655.1, 184663.4)</td>
<td>0.000000151</td>
<td>186</td>
</tr>
</tbody>
</table>

In general, the MOPSO outperforms all other algorithms in most of the comparison criteria, which confirm with some previous studies. Also, identifying a unique PC solution helps the decision makers to determine a unique solution that satisfies all objectives fairly.

6. Limitations and suggested improvements

The four developed metaheuristic models have been applied on a groundwater example application and worked effectively. Also, the models were experimented with two benchmark problems with different shapes of PFs and performed well. The developed models used a unique solution that satisfies fairly all competing criteria of the problem "PC solution" to drive the fitness. As such, tracking a single point resembles optimizing a single-objective. Also, it helps the decision maker by presenting a unique PC solution especially in complex objective spaces.

Despite their perceived benefits, the developed model still has some limitations with a number of possible improvements, including:

- For the groundwater management problem, considering other objectives in addition to the total pumping and total cost;
- studying different aquifers shapes other than rectangular or near rectangular shapes.; and
- Compare the performance of the proposed multi-objective optimization strategy with other strategies used in the literature.
7. Conclusions

This research applied new evolution strategy of tracking a single compromise solution to modify the GA, MA, PSO, and SFLA algorithms to tackle multiple objectives problems. Two multiple objectives standard test problems with known PFs are used to check the ability of the proposed models with the proposed evolution strategy to arrive at the correct PFs. The proposed four algorithms are then applied on a popular groundwater management example for maximizing pumping rates and minimizing operation costs.

The suggested four models proved their ability to obtain the close true PFs for the two benchmark problems in addition to the groundwater application. Accordingly, they can effectively be used to deal with real-life groundwater management application. The performance of the different multi-objective algorithms is studied and compared. The MOPSO outperformed all other algorithms based on the used performance metrics in terms of regularity and distribution of PF throughout the solution space and smallest processing time when compared with other multi-objective optimization algorithms. In this application, a finite element-based simulation model, with a new idea to reduce the run time, is linked with each proposed model for evaluating the potential solution.

Appendix A

A.1 Benchmark problem (1)

The first benchmark problem is called the “Kita” problem with two objectives (Eq. A.1), two unknowns, and three constraints (Eq. 9) [44]:

\[ f_1(x, y) = y - x^2 - p, \quad f_2(x, y) = 0.5x + y + 1 - p, \quad 0 \leq x, y \leq 7 \] (A.1)

Subjected to:

\[ p_1 = \left( \frac{1}{6}x + y - \frac{13}{2} \right) \leq 0, \quad p_2 = \left( \frac{1}{2}x + y - \frac{15}{2} \right) \leq 0, \quad p_3 = (5x + y - 30) \leq 0 \] (A.2)

\[ P = \begin{cases} 12p_1 + p_2 + p_3 & \text{if} \quad p_1, p_2, p_3 > 0 \\ 0 & \text{Otherwise} \end{cases} \] (A.3)

where, \( P \) is the penalty term for constraints violations. In this study, different penalty functions are experimented with (linear, exponential, etc.) and the best results were associated with the proposed linear penalty function (Eq. A.3).

A.2. Benchmark problem (2)

The second benchmark problem is called the “Kursawe” problem with two objectives and three unknowns [44].
\[ f_1(x_i) = \sum_{i=1}^{n-1} \left[ -10 \exp \left( -0.2 \sqrt{x_i^2 + x_{i+1}^2} \right) \right] \]  
(A.4)

\[ f_2(x_i) = \sum_{i=1}^{n-1} \left[ x_i^{0.8} + 5 \sin(x_i) \right] \]  
(A.5)

where, the three unknowns are in the range [-5, 5].

References


