Contents lists available at **SCCE**

Journal of Soft Computing in Civil Engineering

Journal homepage: www.jsoftcivil.com

Developing Four Metaheuristic Algorithms for Multiple-Objective Management of Groundwater

H.A. El-Ghandour[1]*, E. Elbeltagi[2]

1. Associate Professor, Irrigation & Hydraulics Department, Faculty of Engineering Mansoura University, Mansoura 35516, Egypt
2. Professor, Structural Engineering Department, Faculty of Engineering Mansoura University, Mansoura 35516, Egypt

Corresponding author: *hamdy_a@mans.edu.eg*

https://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.

How to cite this article: El-Ghandour HA, Elbeltagi E. Developing four metaheuristic algorithms for multiple-objective management of groundwater. J Soft Comput Civ Eng 2018;2(4):01–22. https://doi.org/10.22115/scce.2018.128344.1057.

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 [6–9,26–33]. 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) [1–5,10–25].

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 [16,17,19].

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:

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

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.

$$Fitness_k = d_{PC,k} \qquad k = 1, 2, 3 \ldots\ldots N \tag{2}$$

5. The relative fitness of each solution in the PF is then calculated as follows:

$$Relative\ Fitness_k = \frac{Fitness_K}{\sum_{k=1}^{N} Fitness_k} \tag{3}$$

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 chromones 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.

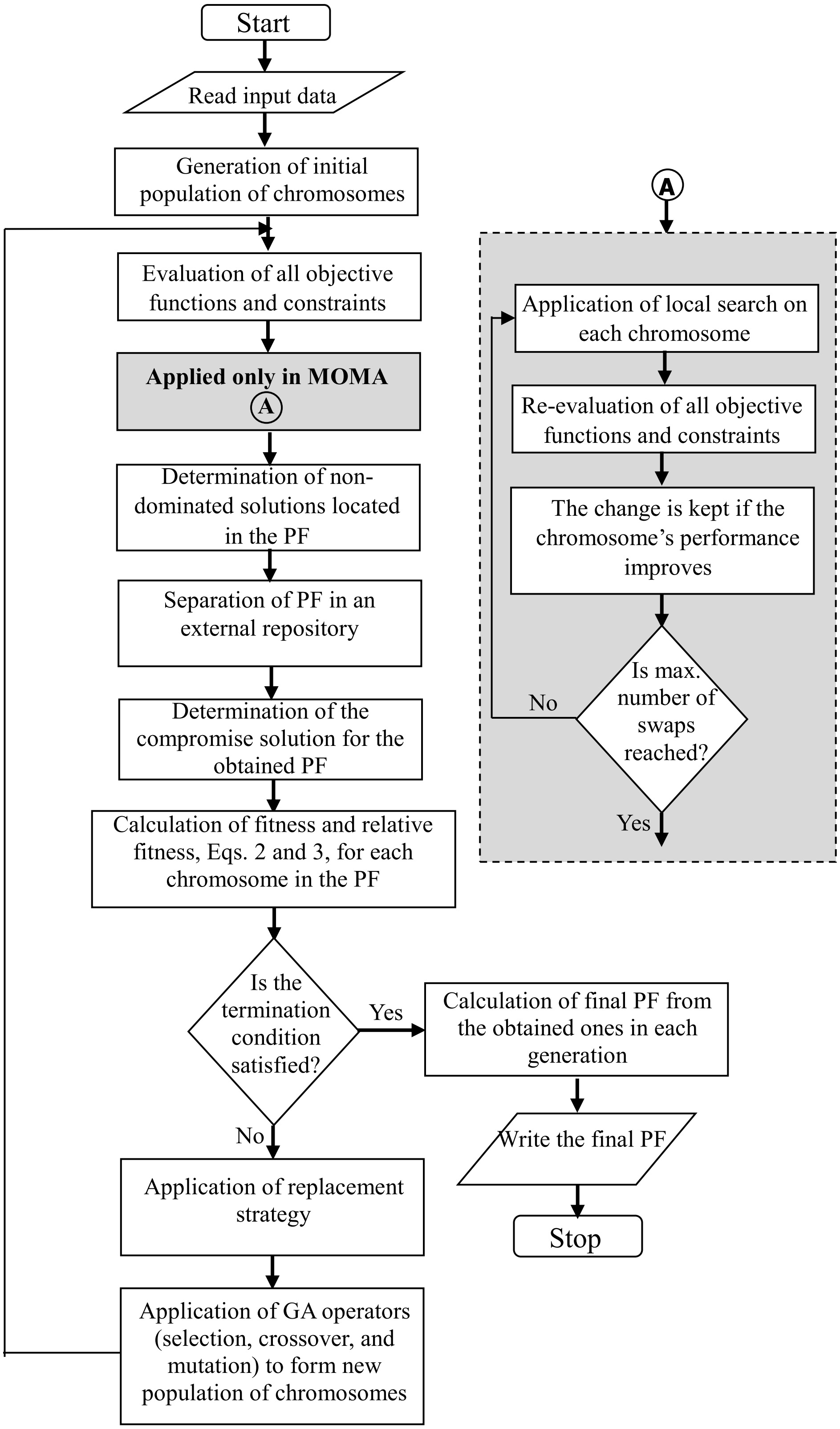
Start
Read input data
Generation of initial population of chromosomes
Evaluation of all objective functions and constraints
Applied only in MOMA
A
Determination of non-dominated solutions located in the PF
Separation of PF in an external repository
Determination of the compromise solution for the obtained PF
Calculation of fitness and relative fitness, Eqs. 2 and 3, for each chromosome in the PF
Is the termination condition satisfied?
Yes
No
Application of replacement strategy
Application of GA operators (selection, crossover, and mutation) to form new population of chromosomes
Calculation of final PF from the obtained ones in each generation
Write the final PF
Stop
A
Application of local search on each chromosome
Re-evaluation of all objective functions and constraints
The change is kept if the chromosome's performance improves
Is max. number of swaps reached?
No
Yes

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].

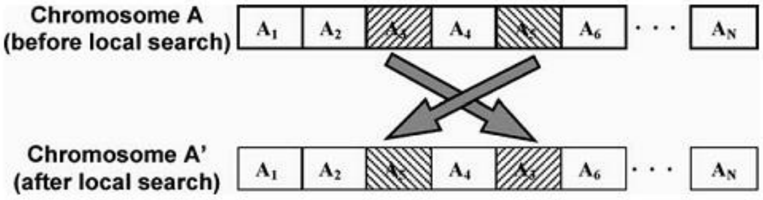

Fig. 2. Applying local search using pair-wise interchange (Elbeltagi et al. 2005).

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]:

$$V_i(I+1)=\omega\times V_i(I)+C_1\times r_1\times[P_i(I)-X_i(I)]+C_2\times r_2\times[P_g(I)-X_i(I)], \quad (4)$$

$$V_{\max}\geq V_i(I+1)\geq -V_{\max},\ i=1,.....,n_p$$

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

$$X_i(I+1)=X_i(I)+V_i(I+1) \quad (5)$$

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 $(I+1)$; ω: 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_{max}: the maximum value of velocity change.

The main PSO parameters are: the population size; number of cycles; V_{max}; C_1; C_2; and ω.

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_{Wnew}) are calculated as follows [38,43]:

$$D_i = C \times r \times (X_B - X_W) \quad (6)$$

$$X_{Wnew}= X_{Wcurrent} + D_i, \qquad D_{max} \geq D_i \geq - D_{max} \quad (7)$$

in which, C: the search acceleration factor; r: a random number in the range [0, 1]; and D_{max}: the maximum allowed frog's position change.

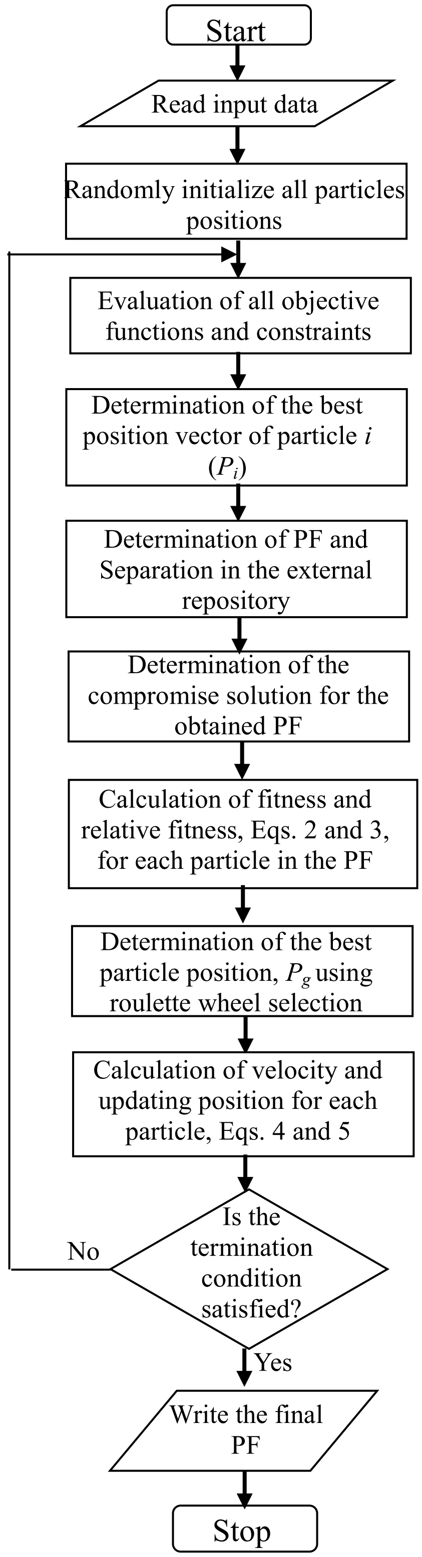
Start
Read input data
Randomly initialize all particles positions
Evaluation of all objective functions and constraints
Determination of the best position vector of particle i (P_i)
Determination of PF and Separation in the external repository
Determination of the compromise solution for the obtained PF
Calculation of fitness and relative fitness, Eqs. 2 and 3, for each particle in the PF
Determination of the best particle position, P_g using roulette wheel selection
Calculation of velocity and updating position for each particle, Eqs. 4 and 5
Is the termination condition satisfied?
No
Yes
Write the final PF
Stop

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.

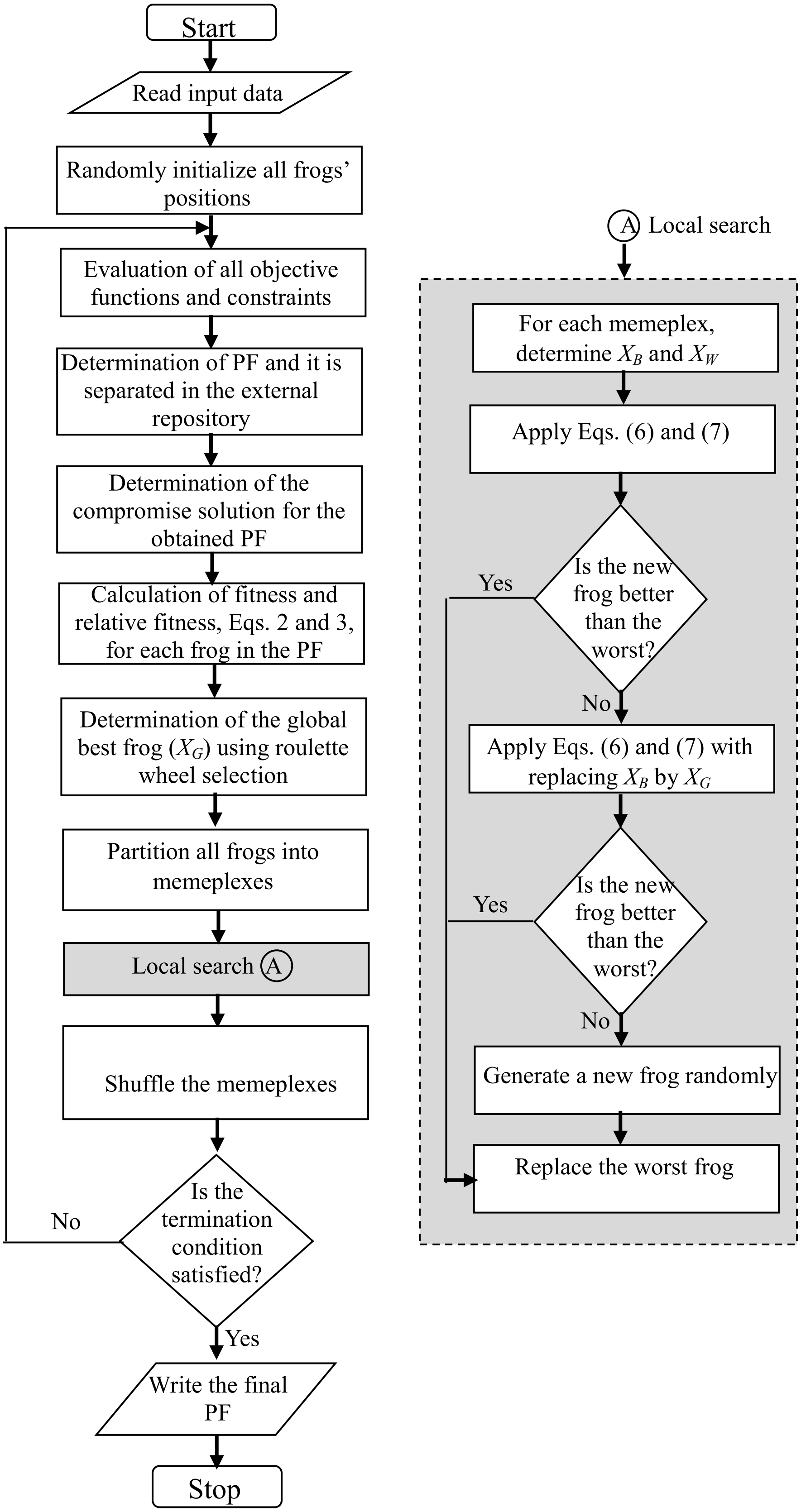
Start
Read input data
Randomly initialize all frogs' positions
Evaluation of all objective functions and constraints
Determination of PF and it is separated in the external repository
Determination of the compromise solution for the obtained PF
Calculation of fitness and relative fitness, Eqs. 2 and 3, for each frog in the PF
Determination of the global best frog (X_G) using roulette wheel selection
Partition all frogs into memeplexes
Local search A
Shuffle the memeplexes
Is the termination condition satisfied?
No
Yes
Write the final PF
Stop
A Local search
For each memeplex, determine X_B and X_W
Apply Eqs. (6) and (7)
Is the new frog better than the worst?
Yes
No
Apply Eqs. (6) and (7) with replacing X_B by X_G
Is the new frog better than the worst?
Yes
No
Generate a new frog randomly
Replace the worst frog

Fig. 4. Flowchart describing the MOSFLA algorithm.

Table 1
Parameters' values used in the two test problems and the example application.

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

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:

$$G_d = \frac{\sqrt{\sum_{i=1}^{N} d_i^2}}{N} \tag{8}$$

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.

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

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.

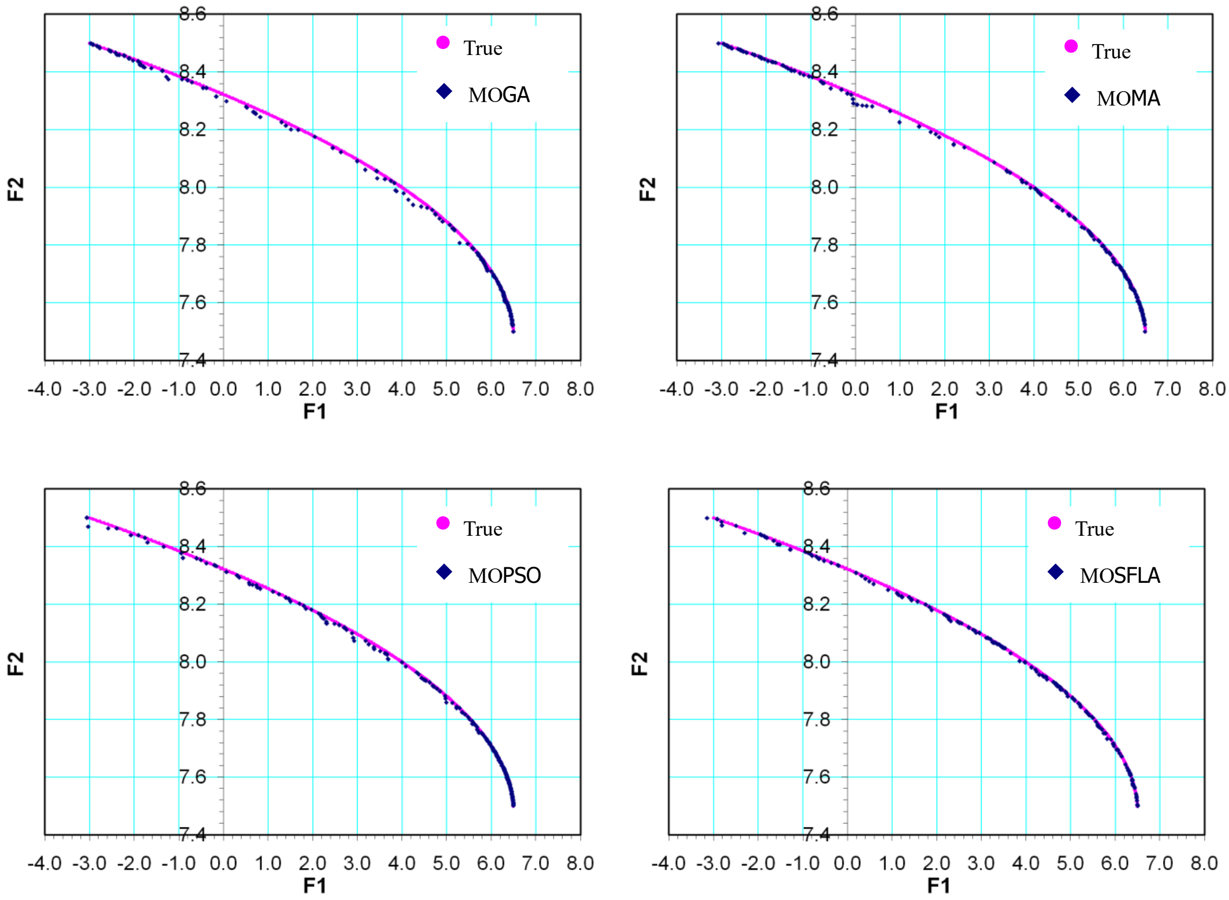

Fig. 5. PFs produced by the four proposed algorithms and the true PF for the Kita function.

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.

$$Dist. = \sqrt{\left(x_{true} - x_{obt}\right)^2 - \left(y_{true} - y_{obt}\right)^2} \tag{9}$$

in which, x_{true} and y_{true}: the coordinates of the compromise solution corresponding to the true PF and x_{obt} and y_{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]:

$$MSE = (1/n) \times \sum \left(1 - f_i^* / f_i^0\right)^2; (i = 1, \ldots\ldots, n) \tag{10}$$

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.

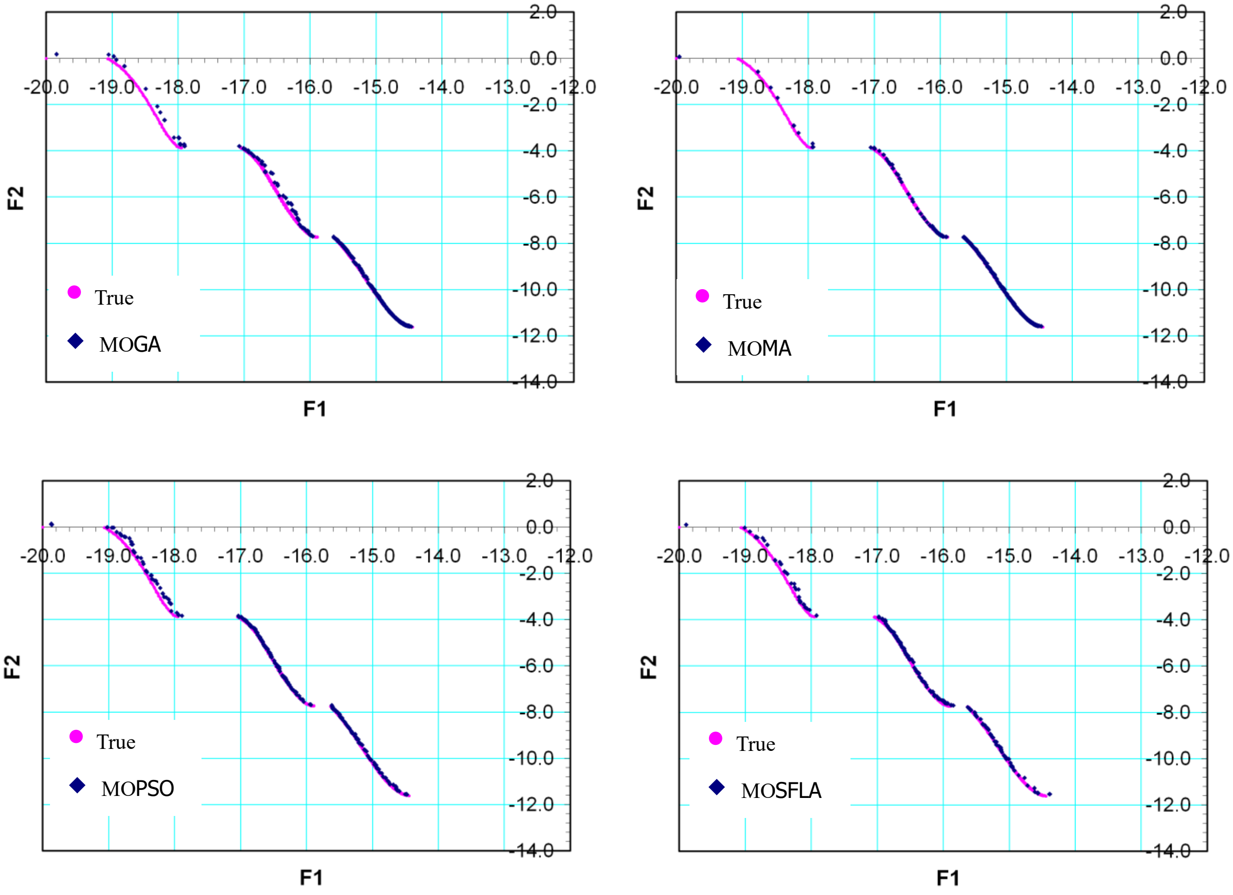

Fig. 6. PFs produced by the four proposed algorithms and the true PF for the Kursawe function.

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 [15,24]:

$$f_1 = \max.\ \left[\sum_{i=1}^{N_W} Q_i - P(h)\right] \tag{11}$$

$$f_2 = \min.\ \left\{\sum_{i=1}^{N_W}\left[a_1 y(d_i)^{b_1} + a_2(Q_i)^{b_2}(d_i - h_i)^{b_3} + a_3 Q_i(d_i - h_i) + P(h) + P(Q)\right]\right\} \tag{12}$$

Subjected to:

$$\sum_{i=1}^{N_W} Q_i \geq Q_{req.} \tag{13}$$

$$h_i \geq h_{i,\, min}, \qquad i = 1, 2, 3, \ldots\ldots N_W \tag{14}$$

$$Q_{i\ min} \leq Q_i \leq Q_{i\ max}, \qquad i = 1, 2, 3, \ldots\ldots N_W \tag{15}$$

$$k = \begin{cases} 1 & if \quad Q_i \neq 0 \\ 0 & if \quad Q_i = 0 \end{cases} \quad i = 1, 2, 3, \ldots\ldots N_W \tag{16}$$

$$P(h) = \begin{cases} h_{i\min} - h_i & if \quad h_i < h_{i\min} \\ 0 & if \quad h_i \geq h_{i\min} \end{cases} \quad i = 1, 2, 3, \ldots\ldots N_W \tag{17}$$

$$P(Q) = \begin{cases} Q_{req.} - \sum Q & if \quad \sum Q < Q_{req.} \\ 0 & if \quad \sum Q > Q_{req.} \end{cases} \quad i = 1, 2, 3, \ldots\ldots N_W \tag{18}$$

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 \$/m, a_2 = 0.12 \$/m^4, a_3 = 0.03 \$/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.

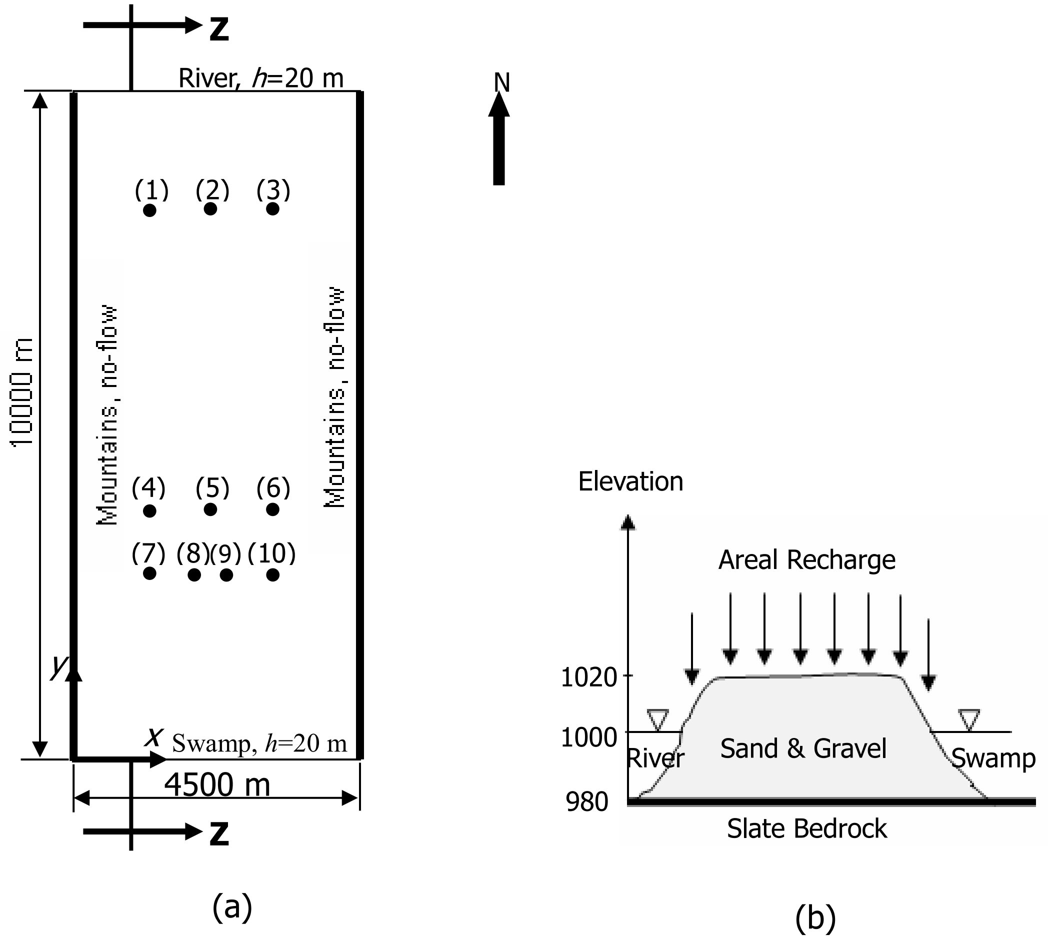

Fig. 7. The plan view and sectional elevation of the studied aquifer [24].

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].

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

in which, K: the hydraulic conductivity of aquifer; ϕ: 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:

$$[A]\times[\varphi]=[F] \tag{20}$$

in which, A: the conductance/stiffness matrix, φ: 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.

$$S_P = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left(\bar{d} - d_i\right)^2} \tag{21}$$

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.

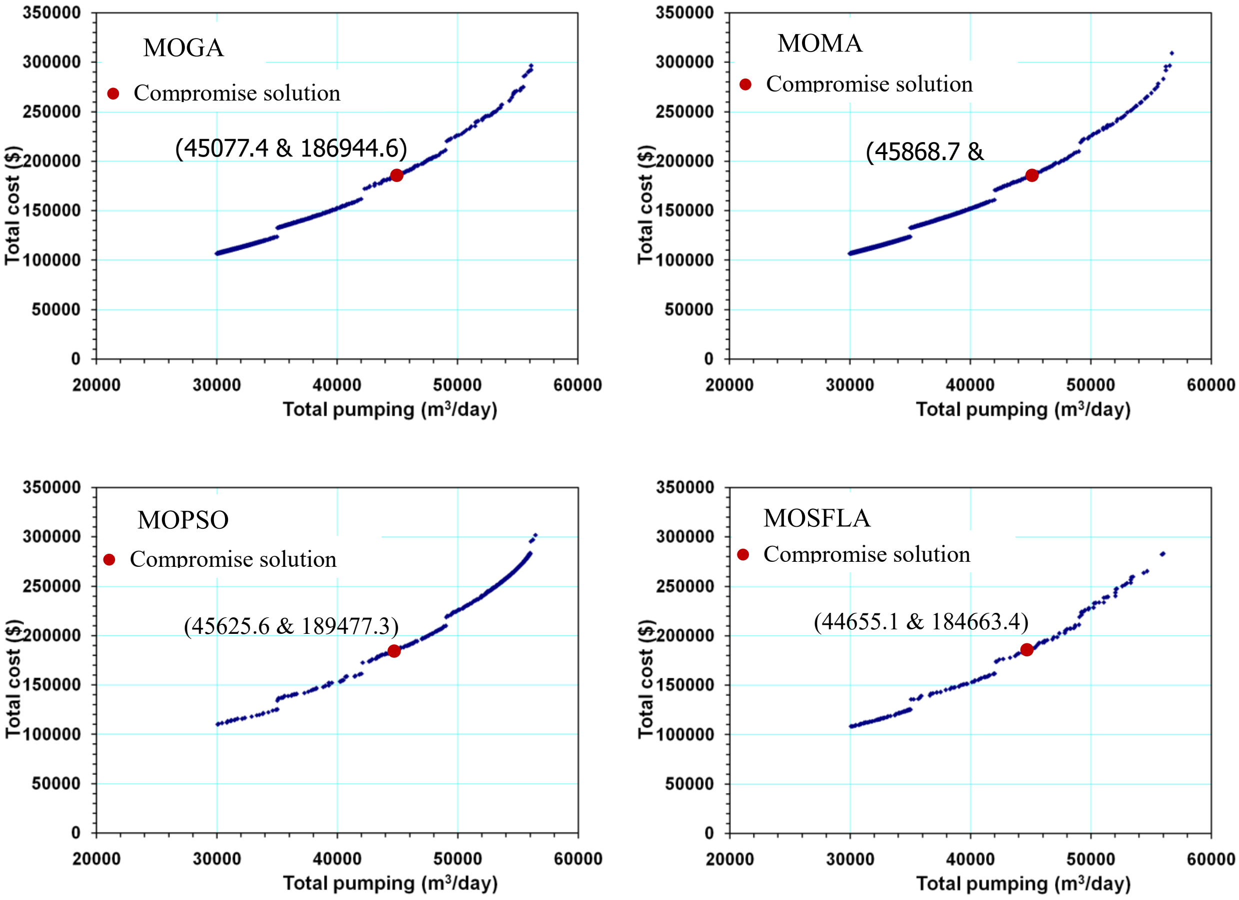

Fig. 8. PFs for the example application produced by the developed four algorithms.

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.

metric **Algorithm**	S_p	**Processing time (second)**	**Compromise solution**	**Best alternative**	***MSE***	**No. of *PF* solutions**
MOGA	231.6	174.408	(45094.6, 186891,0)	(45077.5, 186944.6)	0.000000114	1221
MOMA	2842.6	745.166	(45792.4, 189431.5)	(45868.7, 189665.6)	0.00000215	1551
MOPSO	129.3	138.965	(45582.2, 189013.0)	(45625.6, 189477.3)	0.00000347	3968
MOSFLA	479.9	593.794	(44672.6, 184592.2)	(44655.1, 184663.4)	0.000000151	186

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 ground water example application and worked effectively. Also, the models were experimented with two bench mark 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]:

$$f_1(x,y) = y - x^2 - p, \qquad f_2(x,y) = 0.5x + y + 1 - p, \qquad 0 \le x, y \le 7 \tag{A.1}$$

Subjected to:

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

$$P = \begin{cases} 12p_1 + p_2 + p_3 & if \quad p_1, p_2, p_3 > 0 \\ 0 & Otherwise \end{cases} \tag{A.3}$$

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].

$$f_1(x_i) = \sum_{i=1}^{n-1} \left[-10 \exp\left(-0.2\sqrt{x_i^2 + x_{i+1}^2} \right) \right] \tag{A.4}$$

$$f_2(x_i) = \sum_{i=1}^{n} \left[|x_i|^{0.8} + 5\sin(x_i)^3 \right] \tag{A.5}$$

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

References

[1] McKinney DC, Lin M-D. Genetic algorithm solution of groundwater management models. Water Resour Res 1994;30:1897–906. doi:10.1029/94WR00554.

[2] Wang M, Zheng C. GROUND WATER MANAGEMENT OPTIMIZATION USING GENETIC ALGORITHMS AND SIMULATED ANNEALING: FORMULATION AND COMPARISON. J Am Water Resour Assoc 1998;34:519–30. doi:10.1111/j.1752-1688.1998.tb00951.x.

[3] Gaur S, Ch S, Graillot D, Chahar BR, Kumar DN. Application of Artificial Neural Networks and Particle Swarm Optimization for the Management of Groundwater Resources. Water Resour Manag 2013;27:927–41. doi:10.1007/s11269-012-0226-7.

[4] Pramada SK, Mohan S, Sreejith PK. Application of genetic algorithm for the groundwater management of a coastal aquifer. ISH J Hydraul Eng 2018;24:124–30. doi:10.1080/09715010.2017.1378597.

[5] Boddula S, T. I. E. Groundwater management using a new coupled model of meshless local Petrov-Galerkin method and modified artificial bee colony algorithm. Comput Geosci 2018;22:657–75. doi:10.1007/s10596-018-9718-8.

[6] Nassery HR, Adinehvand R, Salavitabar A, Barati R. Water Management Using System Dynamics Modeling in Semi-arid Regions. Civ Eng J 2017;3:766–78. doi:10.21859/cej-030913.

[7] Alizadeh MJ, Shahheydari H, Kavianpour MR, Shamloo H, Barati R. Prediction of longitudinal dispersion coefficient in natural rivers using a cluster-based Bayesian network. Environ Earth Sci 2017;76:86. doi:10.1007/s12665-016-6379-6.

[8] Barati R. Parameter Estimation of Nonlinear Muskingum Models Using Nelder-Mead Simplex Algorithm. J Hydrol Eng 2011;16:946–54. doi:10.1061/(ASCE)HE.1943-5584.0000379.

[9] Barati R. Application of excel solver for parameter estimation of the nonlinear Muskingum models. KSCE J Civ Eng 2013;17:1139–48. doi:10.1007/s12205-013-0037-2.

[10] Haddad OB, Mariño MA. Optimum operation of wells in coastal aquifers. Proc Inst Civ Eng - Water Manag 2011;164:135–46. doi:10.1680/wama.1000037.

[11] Hosseini K, Nodoushan EJ, Barati R, Shahheydari H. Optimal design of labyrinth spillways using meta-heuristic algorithms. KSCE J Civ Eng 2016;20:468–77. doi:10.1007/s12205-015-0462-5.

[12] Wu J, Zhu X, Liu J. Using genetic algorithm based simulated annealing penalty function to solve groundwater management model. Sci China Ser E Technol Sci 1999;42:521–9. doi:10.1007/BF02917406.

[13] Wu J, Zhu X. Using the Shuffled Complex Evolution Global Optimization Method to Solve Groundwater Management Models, 2006, p. 986–95. doi:10.1007/11610113_105.

[14] Zhu X, Wu J, Wu J. Application of SCE-UA to Optimize the Management Model of Groundwater Resources in Deep Aquifers of the Yangtze Delta. First Int Multi-Symposiums Comput Comput Sci, IEEE; 2006, p. 303–8. doi:10.1109/IMSCCS.2006.192.

[15] Tamer Ayvaz M. Application of Harmony Search algorithm to the solution of groundwater management models. Adv Water Resour 2009;32:916–24. doi:10.1016/j.advwatres.2009.03.003.

[16] Gaur S, Chahar BR, Graillot D. Analytic elements method and particle swarm optimization based simulation–optimization model for groundwater management. J Hydrol 2011;402:217–27. doi:10.1016/j.jhydrol.2011.03.016.

[17] Gaur S, Mimoun D, Graillot D. Advantages of the analytic element method for the solution of groundwater management problems. Hydrol Process 2011;25:3426–36. doi:10.1002/hyp.8071.

[18] Mategaonkar M, Eldho TI. Groundwater remediation optimization using a point collocation method and particle swarm optimization. Environ Model Softw 2012;32:37–48. doi:10.1016/j.envsoft.2012.01.003.

[19] El-Ghandour HA, Elsaid A. Groundwater management using a new coupled model of flow analytical solution and particle swarm optimization. Int J Water Resour Environ Eng 2013;5:1–11.

[20] Park C-H, Aral MM. Multi-objective optimization of pumping rates and well placement in coastal aquifers. J Hydrol 2004;290:80–99. doi:10.1016/j.jhydrol.2003.11.025.

[21] Abdel-Gawad HA. Multi-objective management of heterogeneous coastal aquifers. Mansoura Eng J 2004;29:C1–14.

[22] Siegfried T, Bleuler S, Laumanns M, Zitzler E, Kinzelbach W. Multiobjective Groundwater Management Using Evolutionary Algorithms. IEEE Trans Evol Comput 2009;13:229–42. doi:10.1109/TEVC.2008.923391.

[23] Saafan TA, Moharram SH, Gad MI, KhalafAllah S. A multi-objective optimization approach to groundwater management using genetic algorithm. Int J Water Resour Environ Eng 2011;3:139–49.

[24] El-Ghandour HA, Elbeltagi E. Optimal Groundwater Management Using Multiobjective Particle Swarm with a New Evolution Strategy. J Hydrol Eng 2014;19:1141–9. doi:10.1061/(ASCE)HE.1943-5584.0000910.

[25] El-Ghandour HA, Elabd SM. Studying the reliability in multi-objective management of groundwater under uncertainty of hydraulic conductivity values. Mansoura Eng J 2015;40:C58–72.

[26] Wanakule N, Mays LW, Lasdon LS. Optimal Management of Large-Scale Aquifers: Methodology and Applications. Water Resour Res 1986;22:447–65. doi:10.1029/WR022i004p00447.

[27] Willis R. A planning model for the management of groundwater quality. Water Resour Res 1979;15:1305–12. doi:10.1029/WR015i006p01305.

[28] Aguado E, Remson I, Pikul MF, Thomas WA. Optimum pumping to prevent dewatering. J Hydraul Div 1974;100:860–77.

[29] Andricevic R. A Real-Time Approach to Management and Monitoring of Groundwater Hydraulics. Water Resour Res 1990;26:2747–55. doi:10.1029/WR026i011p02747.

[30] Gorelick SM, Remson I, Cottle RW. Management model of a groundwater system with a transient pollutant source. Water Resour Res 1979;15:1243–9. doi:10.1029/WR015i005p01243.

[31] Gorelick SM, Voss CI, Gill PE, Murray W, Saunders MA, Wright MH. Aquifer Reclamation Design: The Use of Contaminant Transport Simulation Combined With Nonlinear Programing. Water Resour Res 1984;20:415–27. doi:10.1029/WR020i004p00415.

[32] Jones L, Willis R, Yeh WW-G. Optimal control of nonlinear groundwater hydraulics using differential dynamic programming. Water Resour Res 1987;23:2097–106. doi:10.1029/WR023i011p02097.

[33] Lee S-I, Kitanidis PK. Optimal Estimation and Scheduling in Aquifer Remediation With Incomplete Information. Water Resour Res 1991;27:2203–17. doi:10.1029/91WR01307.

[34] Mora-Melia D, Iglesias-Rey P, Martínez-Solano F, Muñoz-Velasco P. The Efficiency of Setting Parameters in a Modified Shuffled Frog Leaping Algorithm Applied to Optimizing Water Distribution Networks. Water 2016;8:182. doi:10.3390/w8050182.

[35] El-Ghandour HA, Elbeltagi E. Comparison of Five Evolutionary Algorithms for Optimization of Water Distribution Networks. J Comput Civ Eng 2018;32:04017066. doi:10.1061/(ASCE)CP.1943-5487.0000717.

[36] Grierson DE. Pareto multi-criteria decision making. Adv Eng Informatics 2008;22:371–84. doi:10.1016/j.aei.2008.03.001.

[37] Elbeltagi E, Hegazy T, Grierson D. A new evolutionary strategy for pareto multi-objective optimization. Proc 7th Int Conf Eng Comput Technol Civil-Comp Press Scotl, 2010.

[38] Elbeltagi E, Hegazy T, Grierson D. Comparison among five evolutionary-based optimization algorithms. Adv Eng Informatics 2005;19:43–53. doi:10.1016/j.aei.2005.01.004.

[39] Kennedy J, Eberhart R. Particle swarm optimization. Proc IEEE Int Conf Neural Netw IV, 1942–1948, vol. 4, IEEE; 1995, p. 1942–8. doi:10.1109/ICNN.1995.488968.

[40] Shi Y, Eberhart R. A modified particle swarm optimizer. 1998 IEEE Int Conf Evol Comput Proceedings IEEE World Congr Comput Intell (Cat No98TH8360), IEEE; n.d., p. 69–73. doi:10.1109/ICEC.1998.699146.

[41] Zhang H, Li H. Multi-objective particle swarm optimization for construction time-cost tradeoff problems. Constr Manag Econ 2010;28:75–88. doi:10.1080/01446190903406170.

[42] Baltar AM, Fontane DG. A generalized multiobjective particle swarm optimization solver for spreadsheet models: application to water quality. Hydrol Days 2006:1–12.

[43] Elbeltagi E, Hegazy T, Grierson D. A modified shuffled frog-leaping optimization algorithm: applications to project management. Struct Infrastruct Eng 2007;3:53–60. doi:10.1080/15732470500254535.

[44] Coello CAC, Pulido GT, Lechuga MS. Handling multiple objectives with particle swarm optimization. IEEE Trans Evol Comput 2004;8:256–79. doi:10.1109/TEVC.2004.826067.