Self-adaptive multi-population-based Jaya algorithm to optimize the cropping pattern under a constraint environment

Vijendra Kumar and S. M. Yadav

ABSTRACT

Increasing population around the world, especially in India and China, has resulted in a drastic increase in water intake in both domestic and agricultural sectors. This, therefore, requires that water resources be planned and controlled wisely and effectively. With this consideration, the aim of the study is to achieve an optimal cropping pattern under a constrained environment. The objective is to maximize the net benefits with an optimum use of water. For optimization, a self-adaptive multi-population Jaya algorithm (SAMP-JA) has been used. For the Karjan reservoir in Gujarat State, India, two different models, i.e. maximum and average cropping patterns, were formulated based on the 75 per cent dependable inflow criteria. These two model scenarios are developed in such a way that either model can be selected by the farmer based on the crop area and its respective net benefits. Invasive weed optimization (IWO), particle swarm optimization (PSO), differential evolution (DE) and the firefly algorithm (FA) were compared to the results. The results show that the SAMP-JA obtained the maximum net benefit for both the models. The findings of the research are also compared with the actual cropping pattern. A significant increase has been noted in the cultivation of sugarcane, groundnut, wheat, millet, banana and castor. SAMP-JA has been noted to converge faster and outperforms PSO, DE, IWO, FA, teaching–learning-based optimization (TLBO), the Jaya algorithm (JA), elitist-JA and elitist-TLBO.

Key words | Jaya algorithm, optimization models, self-adaptive, teaching–learning-based optimization

Vijendra Kumar (corresponding author)
S. M. Yadav
Civil Engineering Department,
Sardar Vallabhbhai National Institute of Technology,
Surat, Gujarat,
India
E-mail: *vij100000@gmail.com*

INTRODUCTION

Water resources have been very restricted due to the growing population and demand from the agricultural sector. It has become essential to wisely and effectively allocate water resources, particularly in areas facing an acute water shortage. In order to maximize the agricultural yield, adequate water supply is essential for the growth of the crop. Many irrigation projects have been constructed over the past few centuries. With the growing demand, however, owing to high investment and land acquisition issues, it is not practical to build more and more water resources projects (Adeyemo & Otieno 2010). Optimizing the existing irrigation project is an alternative solution (Kumar & Yadav 2018). The conventional methods of optimization are linear programming (LP), non-linear programming (NLP) and dynamic programming (DP). LP has been the best method of optimization commonly used to obtain the optimum cropping pattern to increase the net benefits (Paul *et al.* 2000; Singh *et al.* 2001; Moradi-Jalal *et al.* 2007). Ghahraman & Sepaskhah (2002) used the system based on NLP and DP to find the optimum water allocation for various crop models. The NLP-based model was used by Georgiou & Papamichail (2008) to optimize water allocation and crop planning. Mainuddin *et al.* (1997) and Sethi *et al.* (2002) used deterministic linear programming

doi: 10.2166/hydro.2019.087

(DLP) to achieve the optimum cropping pattern. Azaiez *et al.* (2005) applied change constrained linear programming (CCLP) to optimize the stochastic nature of the cropping pattern. Sethi *et al.* (2006) used DLP and CCLP to study the optimum cropping pattern. Jothiprakash *et al.* (2011) used CCLP to solve different dependable inflows in order to obtain an optimal cropping pattern based on the monthly and annual probability distribution models.

Although all of these methods have been widely used, they have certain limitations. LP can solve only linear equations and cannot cope with the uncertainty problem (Kumar & Yadav 2018). On the other side, NLP has not been commonly used due to the high computational time and laborious mathematics involved in its development (Singh 2012), whereas DP has a dimensionality curse (Hossain & El-shafie 2013). In addition, many times these techniques cannot provide an optimal global solution. Heuristic and metaheuristic optimization techniques were created to overcome these problems. Initially, heuristic and metaheuristic methods were created to address issues, such as nonlinearity, discontinuity discrete, multi-objective and uncertainty (Fayaed *et al.* 2013). Researchers have used these methods to solve optimal cropping pattern models, for example: the genetic algorithm (GA) (Yang *et al.* 2009; Fallah-Mehdipour *et al.* 2013), goal programming (Fasakhodi *et al.* 2010), non-dominated sorting GA (NSGAII) (Lalehzari *et al.* 2016), fuzzy programming (Srivastava & Singh 2015), particle swarm optimization (PSO) (Noory *et al.* 2012; Habibi Davijani *et al.* 2016), fuzzy stochastic GA (Dutta *et al.* 2016), differential evolution (DE) (Reddy & Kumar 2008) and fuzzy LP (Raju & Kumar 2000; Sahoo *et al.* 2006; Zeng *et al.* 2010; Regulwar & Gurav 2011).

There are some limitations to existing optimization algorithms. The main limitation is algorithm-specific parameters other than the common control parameters, i.e. the number of iterations and population size. GA and NSGAII are the most widely used methods of optimization, but these require proper tuning of the probability of mutation and crossover, as well as reproduction. PSO needs the inertia weights, social and cognitive parameters to be tuned. DE requires tuning of crossover rate and scaling factor. The artificial bee colony algorithm needs the number of bees, such as onlooker, scout and employed bees. The biogeography-based optimization algorithm requires the emigration rate and immigration rate. Similarly, ant colony optimization, the shark algorithm and improved bat algorithm involve tuning of their own parameters. These parameters are called the algorithm-specific parameters and need to be tuned. If proper tuning is not achieved, this will affect the algorithm's overall performance and will increase the computational cost and the tendency to trap in a local optimal solution (Rao & Patel 2013). To overcome these problems, the teaching–learning-based optimization (TLBO) and Jaya algorithm (JA) have been newly developed free of the algorithm-specific parameters (Venkata Rao 2016), thus reducing the user's burden. Rao *et al.* (2011) proposed the TLBO algorithm, which is one of the most newly developed, efficient and simple to understand. It is based on the idea of teaching in the classroom, where the teacher teaches and the student learns. JA, conversely, has been developed very recently and works on the principle that the solution should move toward the best solutions and avoid the worst solution (Venkata Rao 2016).

A multi-population-based JA has been used in the present study which was proposed by Venkata Rao & Saroj (2017). Multi-population helps to divide the entire population into different sub-population groups in order to improve the diversity of search space. The idea is to split the search space into separate fields by allocating distinct sub-population groups. Each population in the sub-population will either intensify or diversify the search space (Venkata Rao & Saroj 2017). The algorithm's performance depends largely on the sub-population number choice. To solve this issue, a self-adaptive multi-population-JA (SAMP-JA) has been used (Venkata Rao & Saroj 2017). The self-adaptive helps to monitor the problem effectively and to change the number of sub-populations based on the landscape changes in the problem, i.e. the increased or decreased number of sub-populations.

The objective of the study is to introduce the SAMP-JA, which adjusts the number of sub-populations based on the landscape changes in the problem. An optimal cropping pattern problem was chosen to explore the efficiency of the algorithm. This research seeks to create an optimum cropping pattern and area to maximize net benefits using 75 per cent dependable inflow for the Karjan irrigation scheme. The SAMP-JA performance is compared to particle swarm optimization (PSO), differential evolution (DE), Invasive weed optimization (IWO) and the firefly algorithm (FA). In addition, to check accuracy, it is compared with TLBO, JA, the elite version of JA and TLBO, i.e. elitist-JA (EJA) and elitist teaching–learning-based optimization (ETLBO) algorithms, and the actual cropping pattern from the literature (Kumar & Yadav 2019). The following section describes the methods and materials used.

MATERIALS AND METHODS

Self-adaptive multi-population Jaya algorithm

JA works on the principle to get closer to a better solution by avoiding failure. The following are the steps to run the SAMP-JA:

- Step 1: The first step is to decide the population size and the iteration size.
- Step 2: Generate the initial solutions between the upper and lower bound of the variables.
- Step 3: The population is grouped into m (i.e. the number of sub-populations) groups based on the quality of the solution (initially considered as $m = 2$). m is updated at each iteration (bigger or smaller depending on the objective function value).
- Step 4: The next step is to identify the best and worst solution from the population list. If $Y_{j,k,i}$ is the jth variable (i.e. $j = 1, 2, \ldots\ldots, D$, D is the number of design variables) for the kth candidate (i.e. population size, $k = 1, 2, \ldots..n$, n is the number of candidate solution) during the ith iteration.
- Step 5: Each sub-population modifies the solution using the following equation:

$$Y_{\text{new},j,k,i} = Y_{j,k,i} + r_{1,j,i}\,(Y_{j,\text{best},i} - |Y_{j,k,i}|) - r_{2,j,i}\,(Y_{j,\text{worst},i} - |Y_{j,k,i}|) \quad (1)$$

  where $Y_{j,\text{best},i}$ and $Y_{j,\text{worst},i}$ are the best and worst solutions, and $Y_{\text{new},j,k,i}$ is the newly updated solution. The term $r_{2,j,i}\,(Y_{j,\text{worst},i} - |Y_{j,k,i}|)$ helps the algorithm to avoid the worst solution and the term $r_{1,j,i}\,(Y_{j,best,i} - |Y_{j,k,i}|)$ helps the algorithm to move toward the best solution. $r_{1,j,i}$ and $r_{2,j,i}$ are the two random numbers between [0,1].
- Step 6: The objective function obtained by $Y_{j,k,i}$, i.e. O(best-before) and $Y_{\text{new},j,k,i}$, i.e. O(best-after) are compared. The modified solution is only accepted if it is better than the old solution, then m is increased by 1, i.e. $m = m + 1$, it helps to increase the search space exploration. If m is decreased by 1, i.e. $m = m - 1$, it helps to exploit the search space, here, $m > 1$.
- Step 7: This completes an iteration. The cycle stops when the maximum number of generation is reached; otherwise, it repeats itself. To maintain diversity, duplicate solutions are replaced by the newly obtained solution. The SAMP-JA flowchart is presented in Figure 1.

Differential evolution

Storn & Price (1996) proposed DE as an efficient heuristic technique used for solving optimization problems. The algorithm works as follows:

- Step 1: Decide the internal parameters and generate the initial population vector solution, i.e. $Y^j_{k,G} = \{y^1_{k,G}, y^2_{k,G}, \ldots\ldots, y^D_{k,G}\}$.

$$y^j_{k,0} = y^j_{\min} + R_{k,j} * (y^j_{\max} - y^j_{\min}) \quad (2)$$

  where $y^j_{k,0}$ is the initial population at the kth component for the jth vector at the $G = 0$ generation, n is the population size, D is the dimensional parameter vector, $y^j_{\min}$ and $y^j_{\max}$ are the lower and upper limit and $R_{k,j}$ is the random number.
- Step 2: The second step mutation operator is used to produce the mutation vector, i.e. $V^j_{k,G} = \{v^1_{k,G}, v^2_{k,G}, \ldots\ldots, v^D_{k,G}\}$ for each target vector $Y^j_{k,G}$.

$$V^j_{k,G} = Y_{r^k_1,G} + F * (Y_{r^k_2,G} - Y_{r^k_3,G}) \quad (3)$$

  where F is the scaling factor parameter that ranges between [0, 2], r^k_1, r^k_2, and r^k_3 are the randomly generated integers between [1, n].
- Step 3: The next crossover operation is implemented in which each target vector $Y^j_{k,G}$ and its respective mutation vector $V^j_{k,G}$ produce a trail vector $U^j_{k,G} = \{u^1_{k,G}, u^2_{k,G}, .., u^D_{k,G}\}$. The basic binomial crossover

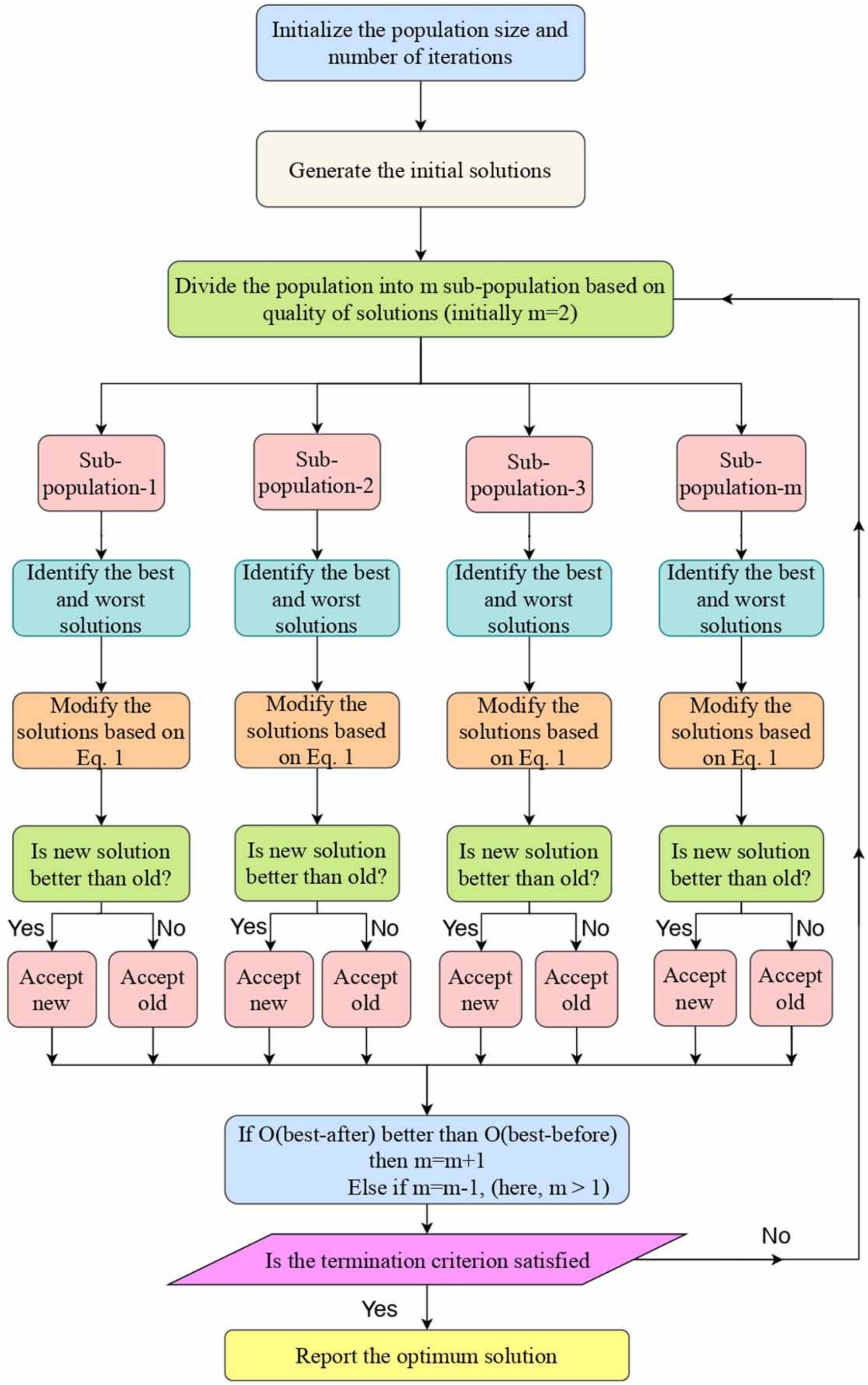

Figure 1 | Flowchart for the SAMP-JA algorithm.

is defined as follows:

$$U^j_{k,G} = \begin{cases} V^j_{k,G} & \text{if (rand } [0,1] \le CR) \text{ or } (j = j_{\text{rand}}) \\ Y^j_{k,G} & \text{otherwise} \end{cases} \quad (4)$$

where j_{rand} is the randomly generated integers between [1, n] and CR is the crossover rate parameter varies between [0, 1].

- Step 4: Selection operation helps to select the best member for the next generation based on the vector's fitness value and its trial vector.

$$Y^j_{k,G+1} = \begin{cases} U^j_{k,G} & \text{if } f(U^j_{k,G}) \le f(Y^j_{k,G}) \\ Y^j_{k,G} & \text{otherwise} \end{cases} \quad (5)$$

- Step 5: Increase the generation to $G = G + 1$. The algorithm reiterates until the maximum iterations are accomplished.

Particle swarm optimization

PSO was proposed by Eberhart & Kennedy (1995). It is a method of optimization based on metaheuristic swarm intelligence and works similar to natural groupings like a flock of birds and a fish school. It provides the optimum solution based on individual and social behavior. The algorithm works as follows:

- Step 1: Decide the internal parameters and generate the initial group of random particles; for each particle, generate position Y_k and velocity v_k.
- Step 2: Evaluate the fitness function of each particle $F(Y_k)$ and obtain the best position using Equation (6):

$$P\text{best}_k = \begin{cases} P\text{best}_k(i-1) & \text{if } F(P\text{best}_k(i)) \le F(P\text{best}_k(i-1)) \\ P\text{best}_k(i) & \text{if } F(P\text{best}_k(i)) > F(P\text{best}_k(i-1)) \end{cases} \quad (6)$$

where F represents the fitness function, $P\text{best}_k$ represents the individual's best position and I represents the iteration, i.e. $i = 1, 2, \ldots\ldots I$, for the kth individual, i.e. population size $k = 1, 2, \ldots n$.

- Step 3: Next, find the global best position from the swarm, i.e.

$$G\text{best}_k(I) = \max\{F(G\text{best}_1(i), F(G\text{best}_2(i), \ldots, F(G\text{best}_n(i)\} \quad (7)$$

- Step 4: Update the velocity using Equation (8) if $v_k > v_{\max}$ then $v_k = v_{\max}$; if $v_k < v_{\min}$ then $v_k = v_{\min}$:

$$v_k(i+1) = w(i)*v_k(i) + c_1 * r_1(i)*(P\text{best}_k(i) - Y_k(i)) + c_2*r_2(i)*(G\text{best}_k(i) - Y_k(i)) \quad (8)$$

where v_k and Y_k are the velocity and position of the kth individual, respectively. The internal parameters are $w(i)$ internal weight, c_1 cognitive parameters and c_2 social parameters.

- Step 5: The next step is to update the velocity using Equation (9).

$$Y_k(i+1) = Y_k(i) + v_k(i+1) \quad (9)$$

This completes one cycle; the algorithm reiterates until the maximum iterations are completed.

Invasive weed optimization algorithm

IWO was proposed by Mehrabian & Lucas (2006) inspired by the concept of colonizing weeds. The algorithm works as follows:

- Step 1: Define the N-dimensions solution space for each variable.
- Step 2: Initialize a population, a finite number of seeds are dispersed over the search space with an initial solution.
- Step 3: Next, evaluate the fitness function and rank the population based on fitness values.
- Step 4: Based on the colony ranking, reproduce new seeds from their minimum to maximum possible seeds production, i.e. $\text{Seed}_{\min}$ and $\text{Seed}_{\max}$. The number of seeds is reproduced using Equation (10):

$$\text{Seed}_n = \text{Seed}_{\min} + \frac{\text{Seed}_{\max} - \text{Seed}_{\min}}{F_b - F_w}(F_p - F_w) \quad (10)$$

where S_n, $\text{Seed}_{\max}$ and $\text{Seed}_{\min}$ are the allowable reproduced, maximum and minimum numbers of seeds, respectively. F_w and F_b are the worst and best fitness function values and F_p is the fitness function of the seed considered.

- Step 5: Disperse the new seed over the search space by varying standard deviations (SDs) using Equation (11):

$$\sigma_{\text{iter}} = \frac{(\text{iter}_{\max} - \text{iter})^a}{(\text{iter}_{\max})^a}(\sigma_{\text{initial}} - \sigma_{\text{final}}) + \sigma_{\text{final}} \quad (11)$$

where $\sigma_{initial}$ and σ_{final} are the initial and final SD, $iter_{max}$ is the maximum number of iteration and a is the non-linear modulation index that controls IWO performance and convergence (Mehrabian & Lucas 2006).

- Step 6: Evaluate the fitness function and rank the new and the parent's seeds. The plants with low rank are eliminated from the colony.
- Step 7: The process is repeated for the next iteration from step 3 and runs until the maximum iteration is achieved or the fitness criterion is obtained.

Firefly algorithm

FA is a swarm-based metaheuristic algorithm introduced by Yang (2008), inspired by the fireflies. The insects have the ability to emit light through the biochemical process called bioluminescence. Nonetheless, its purpose is still uncertain. The two fundamental functions are attracting prospective preys and mating partners. The flashing light also protects the fireflies from other predators. Three basic assumptions or rules were idealized in the development of the FA (Yang 2008). First of all, all the fireflies are unisex. Second, the flash's brightness is directly proportional to the fly's attractiveness. Third, the brightness decreases as the distance increases. The algorithm works as follows:

- Step 1: Decide the internal parameters and generate the initial firefly population n $(k = 1, 2, \ldots n)$ within the N-dimensions solution space $(d = 1, 2, \ldots N)$ for each variable.
- Step 2: Evaluate the fitness function of each $F(Y_k)$ firefly.
- Step 3: The firefly's attractiveness can be expressed using the following equation:

$$\beta(r) = \beta_0 e^{-\gamma r^2} \tag{12}$$

where $\beta(r)$ is the firefly attractiveness, β_0 is the attractiveness at distance $r = 0$ and γ is the light absorption coefficient.

- Step 4: The distance (r_{ij}) between any two fireflies at y_i and y_j can be represented as follows:

$$r_{ij} = ||y_i - y_j|| = \sqrt{\sum_{d=1}^{N} (y_{i,n} - y_{j,n})^2} \tag{13}$$

where $y_{i,n}$ and $y_{j,n}$ are the dth dimension of the ith and jth firefly's position coordinates, respectively.

- Step 5: The i firefly movement is attracted to another more attractive (brighter) firefly j is expressed as follows:

$$y_{new,i} = y_i + \beta_0 e^{-\gamma r^2} (y_{i,n} - y_{j,n}) + \alpha\, \varepsilon_i \tag{14}$$

where α is a randomization coefficient generated between [0, 1] and ε_i is a random number vector.

- Step 6: Evaluate the new fitness function, rank the fireflies and find the current best. This completes one cycle; the algorithm reiterates until the maximum iterations are completed.

STUDY AREA DESCRIPTION

The Karjan River originates from the Satpura hill near Mangrol Taluka's Kalvipura village in Gujarat State's Surat district, India. It has an overall length of 96.55 km. The Karjan Dam is situated at a latitude of 21°49′N and a longitude of 73°32′E. The main purpose of the dam is irrigation, but later micro-hydropower generation was added. The total length and height of the dam are 911.11 and 119.70 m, with a catchment area of 1,403.78 km^2. The dam is equipped with nine radial gates. Table 1 shows Karjan Dam's salient features. Figure 2 depicts the index map of the Karjan irrigation project with its cultivable command area. The irrigation project has

Table 1 | The salient features of the Karjan Dam

General item	Details
Location of the dam site	Latitude 21°49′ N, Longitude 73°32′ E
Total catchment area	1,403.78 km^2
Average annual rainfall	1,209.15 mm
Main purpose	Irrigation
Hydropower generation	6 MW (2 turbines of 3 MW capacity)
Type	Concrete gravity dam
Type of spillway	Gated spillway
Shape of crest	Ogee
Numbers, size and type of gates	9 numbers, size 15.55 × 14.02 m and (51′ × 46′) radial gates
Low water level	Reduced level 78.00 m
Top of dam	Reduced level 119.70 m
Number and size of the opening	3 numbers (1.83 × 2.75 m), 2 numbers (1.21 × 1.524 m)

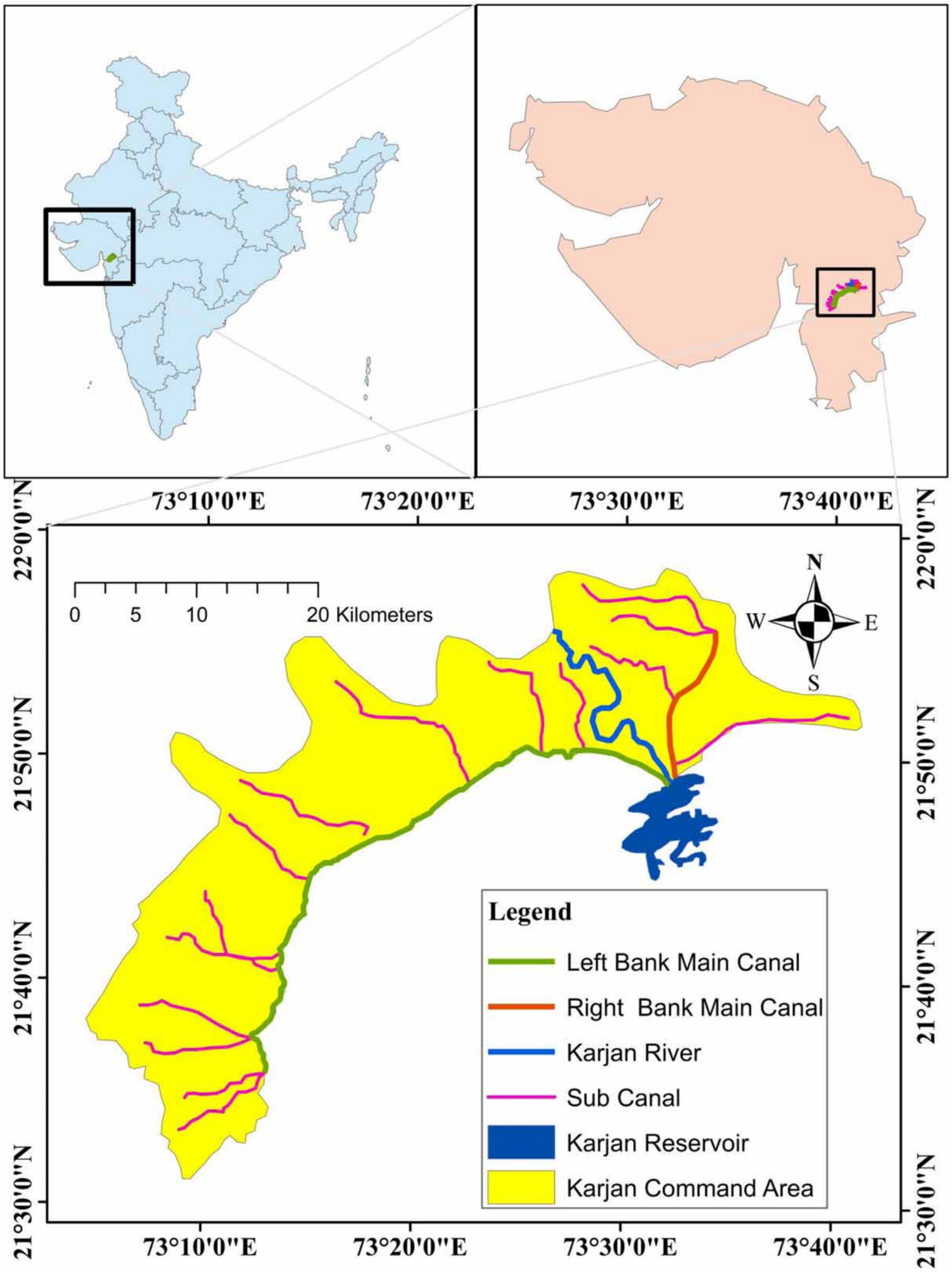

Figure 2 | Index map of the study area.

two canal systems, i.e. the left bank main canal (LBMC) and the right bank main canal (RBMC). The RBMC and LBMC systems are allocated in 4 and 12 zones, respectively. The total length of the RBMC is 12.6 km and LBMC is 51 km. The irrigation project provides water for an area of 11,412 ha under the RBMC and an area of 39,585 ha under the LBMC.

The following data were collected from the state water data center, Gandhinagar, for the Karjan River; monthly reservoir storage capacity for 18 years, monthly reservoir inflow data for 18 years, monthly evaporation losses from the reservoir's free surface for 18 years and monthly overflow data for 18 years. The various crops produced in the command area are groundnut, wheat, sugarcane, banana, millet grains/corn, pulses, vegetables, cotton and castor. The reservoir's evaporation loss is 32 (Mm^3). An average inflow to the reservoir is 989 Mm^3. The minimum and maximum reservoir storage are 49 and 630 Mm^3. Available water for irrigation is 520 Mm^3. The actual cropping pattern of the RBMC and LBMC has been collected for 12 years from 1999 to 2011 and the canal carrying capacity. In addition to these net irrigation, water demands and surface water charges are also collected for each crop. The subsequent section describes the mathematical model.

MATHEMATICAL MODEL FORMULATION

The mathematical formulation of the objective function and the constraints are given in the sequential section.

Objective function

The aim of the present study is to maximize the net benefit in two cropping seasons over a year and is expressed as follows:

$$\text{Max } NB = (C_1\, Rk_1 + C_2\, Rr_2 + \Sigma_{c=3}^{7}\, C_c\, Rp_c + \Sigma_{c=8}^{9}\, C_c\, Rt_c) + (C_1 Lk_1 + C_2\, Lr_2 + \Sigma_{c=3}^{7}\, C_c\, Lp_c + \Sigma_{c=8}^{9}\, C_c\, Lt_c) \quad (15)$$

where Max NB = maximum net benefit in Indian rupees (Rs); c = number of crops, $c = 1, 2 \ldots 9$; Rk, Rr, Rp and Rt are the crop area in (ha), allocated in zones 1–4 for the duration of the Kharif season, Rabi season, annual crops and biannual crops in the RBMC. Similarly, Lk, Lr, Lp and Lt are the crop area in (ha), allocated in zones 5–16 for the duration of the Kharif season, Rabi season, annual crops and biannual crops in the LBMC, respectively. Here, Rk_1 and Lk_1 are the groundnut cropping areas, Rr_2 and Lr_2 are the wheat cropping areas, Rp_3 and Lp_3 are the sugarcane cropping areas, Rp_4 and Lp_4 are the banana cropping areas, Rp_5 and Lp_5 are the millet grains/corn cropping areas, Rp_6 and Lp_6 are the pulses cropping areas, Rp_7 and Lp_7 are the vegetable cropping areas, Rt_8 and Lt_8 are the cotton cropping areas and Rt_9 and Lt_9 is the castor cropping areas, in the RBMC and LBMC, respectively.

With respect to the areas, the cultivation region for the biannual and annual crops is the length for both the seasons. The cropping areas under biannual and annual crops are equal in both the seasons of the years. C_c is the cost coefficient of the crop, $c = 1, 2 \ldots 9$, assuming full irrigation proposed on deducting the fertilizer and labor cost of the total benefits. The net benefit is calculated given that the job is performed by the farmer's family members.

Constraints

The target function is subjected to a set of constraints. These constraints are given under:

Crop area constraints

The cropping area should be greater equal than the minimum cultivable area available for irrigation. Moreover, it should be lower equal than the maximum cultivable area available for irrigation:

$$R_{z,1,\min} \leq Rk_1 \leq R_{z,1,\max}, \quad z = 1, 2 \ldots 4 \quad (16)$$

$$L_{z,1,\min} \leq Lk_1 \leq L_{z,1,\max}, \quad z = 5, 2 \ldots 16 \quad (17)$$

where z = number of zones, $z = 1, 2 \ldots .16$ and $R_{z,1,\min}$ and $R_{z,1,\max}$ are the minimum and maximum cultivable command areas for groundnut crops in the RBMC. Likewise, $L_{z,1,\min}$ and $L_{z,1,\max}$ are the minimum and maximum cultivable command areas for groundnut crops in the LBMC. Similarly, other crops' cultivable command areas are:

$$R_{z,r,\min} \leq Rk_r \leq R_{z,r,\max}, \quad z = 1, 2 \ldots 4 \quad (18)$$

$$L_{z,r,\min} \leq Lk_r \leq L_{z,r,\max}, \quad z = 5, 2 \ldots 16 \quad (19)$$

$$R_{z,p,\min} \leq Rk_p \leq R_{z,p,\max}, \quad z = 1, 2 \ldots 4 \quad (20)$$

$$L_{z,p,\min} \leq Lk_p \leq L_{z,p,\max}, \quad z = 5, 2 \ldots 16 \quad (21)$$

$$R_{z,t,\min} \leq Rk_t \leq R_{z,t,\max}, \quad z = 1, 2 \ldots 4 \quad (22)$$

$$L_{z,t,\min} \leq Lk_t \leq L_{z,t,\max}, \quad z = 5, 2 \ldots 16 \quad (23)$$

Water allocation

The releases from the reservoir in the channel should be more than the water required for the crops using the expression as per Equations (24)–(27):

$$\sum_{t=1}^{6} \mathrm{NIR}_{1,t} Rk_1 \leq \eta_s C_{R,z,t}, \quad z = 1, 2 \ldots 4 \tag{24}$$

$$\sum_{t=1}^{6} \mathrm{NIR}_{1,t} Lk_1 \leq \eta_s C_{L,z,t}, \quad z = 5, 2 \ldots 16 \tag{25}$$

$$\sum_{t=7}^{12} \mathrm{NIR}_{2,t} Rr_2 \leq \eta_s C_{R,z,t}, \quad z = 1, 2 \ldots 4 \tag{26}$$

$$\sum_{t=7}^{12} \mathrm{NIR}_{2,t} Rr_2 \leq \eta_s C_{L,z,t}, \quad z = 5, 2 \ldots 16 \tag{27}$$

where $\mathrm{NIR}_{c,t}$ is the net irrigation requirement of crop c in the month $t = 1, 2, \ldots 12$; time $t = 1$ refers to the month of June and $t = 2$ refers to the month of July. $C_{L,z,t}$ and $C_{R,z,t}$ are irrigation releases to the LBMC and RBMC for a month t in Mm^3. η_s is the efficiency of the surface water system. Similarly, the other crops are presented in a similar manner in two cropping seasons.

Canal capacity constraints

At any time, discharge from the dam needs to be lower equal than the maximum canal carrying capacity, i.e.

$$C_{R,z,t} \leq \mathrm{RCC}_t, \quad z = 1, 2 \ldots 4 \tag{28}$$

$$C_{L,z,t} \leq \mathrm{LCC}_t, \quad z = 5, 2 \ldots 16 \tag{29}$$

where RCC_t and LCC_t are the maximum canal carrying capacity for month t in Mm^3 for the RBMC and LBMC, respectively.

Irrigation demands

$$D_{R,t,\min} \leq C_{R,z,t} \leq D_{R,t,\max} \tag{30}$$

$$D_{L,t,\min} \leq C_{L,z,t} \leq D_{L,t,\max} \tag{31}$$

where $D_{R,t,\min}$ and $D_{R,t,\max}$ are the minimum and maximum irrigation demands for the RBMC, respectively; and $D_{L,t,\min}$ and $D_{L,t,\max}$ are the minimum and maximum irrigation demands for the LBMC, respectively.

Storage constraints

The reservoir storage should be lower equal than the maximum storage capacity of the reservoir, and it should be greater equal than the minimum storage capacity:

$$S_{\min} \leq S_t \leq S_{\max} \tag{32}$$

where $S_{\max}$ and $S_{\min}$ are the maximum and minimum active storages of the reservoir in Mm^3, and S_t is the reservoir storage at the beginning of the period t in Mm^3.

Evaporation constraints

The evaporation losses are estimated using the relationship between the initial and final reservoir storage, and the correlation coefficients are obtained. The evaporation constraints expression is represented below (Jothiprakash *et al.* 2011):

$$E_t = a_t + b_t * \frac{(\mathrm{S_t} + \mathrm{S_{t+1}})}{2} \tag{33}$$

The monthly linear models were prepared from the actual evaporation and average storage. Table 2 (Kumar & Yadav 2019) shows the observed regression coefficients with the statistical parameters r for each month, where S_t is the reservoir storage at the beginning of period t in

Table 2 | Summary of the evaporation regression model (Kumar & Yadav 2019)

Month	a_t	b_t	r
January	1.61	0.00303	0.82
February	1.85	0.00399	0.80
March	2.91	0.00322	0.73
April	2.97	0.00496	0.85
May	1.59	0.01440	0.63
June	2.20	0.00540	0.65
July	−0.09	0.01110	0.96
August	1.24	0.00548	0.64
September	0.69	0.00751	0.75
October	10.09	−0.01030	0.58
November	4.33	−0.00136	0.68
December	1.79	0.00240	0.65

Note: a_t and b_t are the regression coefficients during the time period '*t*'.
r: regression coefficients for various months with the statistical parameter '*r*'.

Mm^3, E_t is the evaporation losses from the reservoir in Mm^3, S_{t+1} is the reservoir storage at the end of period $(t+1)$ in Mm^3, and a_t and b_t are the regression coefficients during time period t.

Dependable inflow

Inflow at any percentage probability (P_p) represents the flow magnitude in an average year that can be expected to be equal to or exceed (P_p) per cent of the time and is termed as P_p% dependable inflow I_t (Subramanya 2013). Eighteen years of available historical inflow data were used to generate the nine different dependable inflows. Frequency analysis is obtained to determine the relationship between the magnitude of the rainfall and its probability of exceedance. The concept of the probability of the exceedance of inflow is estimated using the Weibull method (Subramanya 2013):

$$P_p = \frac{m}{N+1} \tag{34}$$

where N is the number of a record year, m is the number of order in the descending order of magnitude and P_p is the probability of an event.

Continuity constraints

Continuity constraints express the relationship between inflow and outflow; this is mathematically given by

$$S_{t+1} = S_t + I_t - C_{R,z,t} - C_{L,z,t} - E_t - \mathrm{Ovf}_t \tag{35}$$

where I_t = dependable inflow into the reservoir during month t in Mm^3 and Ovf_t = overflow from the reservoir at any month t in Mm^3.

Overflow constraints

A condition of overflow should be provided so that additional water can spill off. If no overflow condition is provided, the models will result in spilling water, even if there is less storage capacity:

$$\mathrm{Ovf}_t \geq S_t + I_t - \sum (C_{R,z,t} + C_{L,z,t}) - E_t - S_{\max} \tag{36}$$

where $\mathrm{Ovf}_t > 0$, $t = 1, 2, \ldots 12$.

MODEL APPLICATION

The dependable annual inflows were calculated from the annual probability distribution. Nine different rates of inflow were computed using the Weibull formula based on available historical monthly reservoir inflow data, namely 50%, 55%, 60%, 65%, 75%, 80%, 85% and 90%. Figure 3 depicts the annual inflow for various rates of inflow (Kumar & Yadav 2019). It was observed that 95% of rainfall occurs from June to October. Therefore, in the operation of the reservoir, the probability of June to October is more crucial than the annual probability. Of these, 75% dependable inflow was used for the study (Vasan & Raju 2009; Jothiprakash & Arunkumar 2013). Based on the actual condition of the Karjan irrigation scheme, two distinct models were created, namely (1) maximum cropping pattern, i.e. model-1 and (2) average cropping pattern, i.e. model-2. Model-1 has been prepared at 75% dependable inflow with the maximum crop area sown over the past 12 years. Model-2 has been prepared at 75% dependable inflow with an average crop area sown over the past 12 years. Here, the aim behind the maximum and average sown crop area is to check the maximum and average benefits obtained. These two model scenarios are formulated in such a manner that the farmer can choose either model depending on the cultivation area and its corresponding net benefits. To optimize both models, a SAMP-JA was used. SAMP-JA's performance has been compared to PSO, DE, IWO and FA. In addition, JA, TLBO, EJA and ETLBO algorithms and the actual cropping pattern from the literature (Kumar & Yadav 2019) have been compared to verify the accuracy. All algorithms were coded with MATLAB R2014b software, and Origin-Pro 8.5 software was utilized for plots and statistical parameters. Via a 64-bit system type, Intel® core™ i7-7700, @3.60 GHz system configuration with an 8.00 GB ram.

MODEL PARAMETERS

Various combinations of common controlling parameters, such as the number of iterations (i.e. 10, 20, 35, 50, 80 and 100) and population size (i.e. 10, 20, 30, 50, 75 and 100), were used to check algorithms' performance over 30

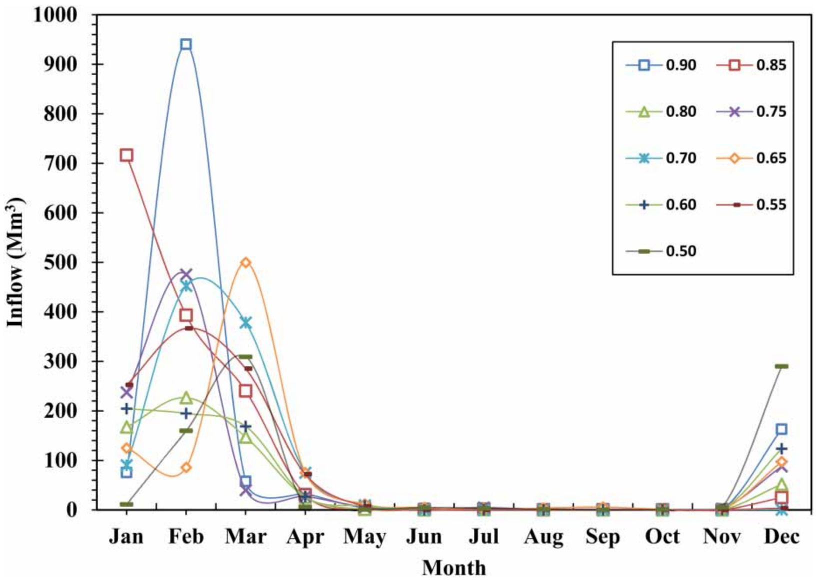

Figure 3 | Annual inflow values for different dependability levels.

independent runs. It was observed that for a population size of 100, all algorithms performed better. The internal parameters in PSO, i.e. the cognitive parameter (c_1) and social parameter (c_2), were taken as 1.3 and 2.0, respectively and the inertia weight $w(i)$ as 1. The internal parameters in DE, i.e. the lower and upper scaling factors, were taken as 0.2 and 0.8, respectively, and the *CR* crossover probability was taken as 0.2. IWO's internal parameters were the minimum and maximum number of seeds, i.e. $S_{\min}$ and $S_{\max}$ were taken as 0 and 3, the non-linear modulation index (a) was taken as 2, and initial SD (σ_{initial}) and final SD (σ_{final}) were taken as 0.6 and 0.001. The internal parameters in FA were the base coefficient of attraction (β_0) taken as 2, the light absorption coefficient (γ) was taken as 1 and the randomized parameter (α) was taken as 0.2. The following section describes the results and the discussion.

RESULTS AND DISCUSSION

Analysis of net benefits

Two models were prepared using the objective function and its constraints as stated earlier. The algorithms were named as SAMP-JA1, PSO1, DE1, IWO1 and FA1 for model-1. Similarly, the algorithms were named as SAMP-JA2, PSO2, DE2, IWO2 and FA2 for model-2. The comparison was done between (SAMP-JA1, PSO1, DE1, IWO1 and FA1) for model-1 and (SAMP-JA2, PSO2, DE2, IWO2 and FA2) for model-2. Table 3 shows model-1 comparative results for the various number of iterations for different algorithms over 30 independent runs and past literature studies (Kumar & Yadav 2019). It can be noted that for all the comparative iterations, SAMP-JA1 performed better and was able to attend very fast convergence. Similarly, Table 4 demonstrates model-2 comparison outcomes for the various number of iterations for different algorithms over 30 independent runs and previous literature studies (Kumar & Yadav 2019). Similar findings were also noted here, where SAMP-JA2 performed better for all the comparative iterations and was able to undergo very rapid convergence. The net benefits were in Indian rupees (1 US$ = 63.96 Rs).

The convergence rates of the various algorithms for model-1 and 2 for 100 iterations are shown in Figures 4 and 5. It was noted that for both models, SAMP-JA converges quickly to an optimum solution. PSO and DE converge slower than SAMP-JA and FA, and IWOs performed poorly. Table 5 shows the comparison of net benefits derived from different algorithms and real income for both the models. In model-1, SAMP-JA1 was 15.93%, 0.59%, 0.05%, 0.06%, 0.05%, 0.04% and 0.003% better

Table 3 | Comparison of net benefits (Rs × 10^7) obtained for model-1 over 30 independent runs

Model-1	Algorithm	10 iterations	20 iterations	35 iterations	50 iterations	80 iterations	100 iterations
Kumar & Yadav (2019)	TLBO1	197.789	198.743	199.081	199.153	199.226	199.226
	ETLBO1	198.529	199.295	199.749	199.956	200.319	200.319
	JA1	199.620	200.263	200.389	200.392	200.395	200.395
	EJA1	200.005	200.266	200.383	200.397	200.400	200.4000
Present study	PSO1	192.074	193.523	194.588	194.684	199.317	200.288
	DE1	185.196	197.027	199.102	199.819	200.234	200.302
	IWO1	165.340	165.796	166.280	166.834	167.810	168.485
	FA1	199.986	200.046	200.239	200.241	200.243	200.303
	SAMP-JA1	**200.212**	**200.322**	**200.390**	**200.399**	**200.4001**	**200.4001**

Note: Bold values indicate the optimal results of different iterations.

Table 4 | Comparison of net benefits (Rs × 10^7) obtained for model-2 over 30 independent runs

Model-2	Algorithm	10 iterations	20 iterations	35 iterations	50 iterations	80 iterations	100 iterations
Kumar & Yadav (2019)	TLBO2	71.828	72.309	72.465	72.533	73.337	73.337
	ETLBO2	72.518	72.906	73.007	73.267	73.374	73.374
	JA2	73.091	73.270	73.326	73.431	73.440	73.440
	EJA2	73.111	73.298	73.424	73.44	73.443	73.443
Present study	PSO2	71.703	71.945	72.204	72.267	73.193	73.422
	DE2	68.880	71.994	72.882	73.178	73.337	73.421
	IWO2	60.095	60.375	60.921	61.372	62.134	62.658
	FA2	73.111	73.319	73.336	73.336	73.340	73.350
	SAMP-JA2	**73.323**	**73.323**	**73.438**	**73.461**	**73.469**	**73.483**

Note: Bold values indicate the optimal results of different iterations.

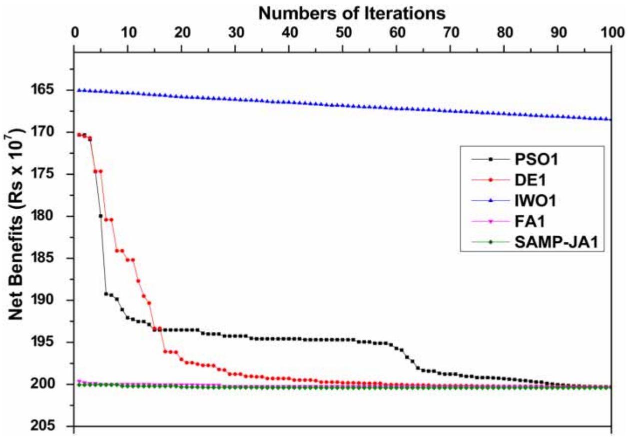

Figure 4 | Convergence plot of different algorithms for model-1 up to 100 iterations.

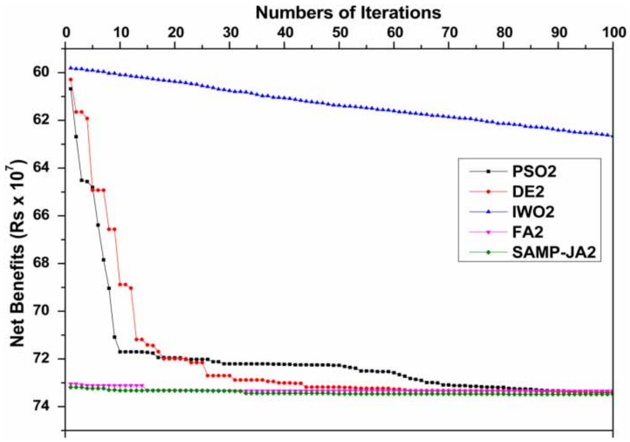

Figure 5 | Convergence plot of different algorithms for model-2 up to 100 iterations.

than IWO1, TLBO1, FA1, PSO1, DE1, ETLBO1 and JA1. The results of SAMP-JA1 and EJA1 were nearly the same, but as shown in Figure 6, SAMP-JA1 had a faster convergence rate. In model-2, SAMP-JA2 was 14.73%, 0.20%, 0.18%, 0.15%, 0.08%, 0.08%, 0.06% and 0.05% better than IWO2, TLBO2, FA2, ETLBO2, DE2, PSO2, JA2 and EJA2. Since the unit was in (Rs x 10^7) Rupees, even a small change in outcomes has resulted in a large change in net benefits. The average time to run the models were from 1 to 2 minutes for all algorithms.

Table 5 | Comparison of net benefits (Rs × 10^7) obtained from different algorithms and actual income for both models up to 100 iterations

Model		Algorithm	Net benefit (Rs × 10^7)
Model-1	Kumar & Yadav (2019)	LP1	183.694
		EJA1	200.400
		ETLBO1	200.319
		JA1	200.395
		TLBO1	199.226
		Real income of 2010	65.373
		Real income of 2011	68.694
	Present study	PSO1	200.288
		DE1	200.302
		IWO1	168.485
		FA1	200.303
		SAMP-JA1	**200.4001**
Model-2	Kumar & Yadav (2019)	LP2	72.533
		EJA2	73.443
		ETLBO2	73.374
		JA2	73.440
		TLBO2	73.337
		Real income of 2010	65.373
		Real income of 2011	68.694
	Present study	PSO2	73.422
		DE2	73.421
		IWO2	62.658
		FA2	73.350
		SAMP-JA2	**73.483**

Note: Bold values indicate the optimal results in model-1 and model-2.

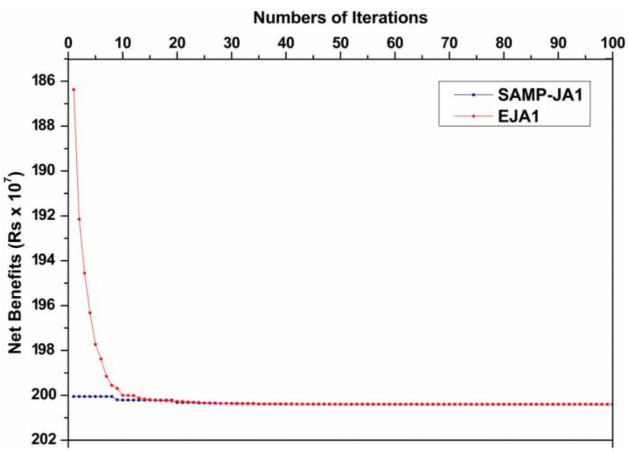

Figure 6 | Convergence plot of SAMP-JA1 and EJA1 up to 100 iterations.

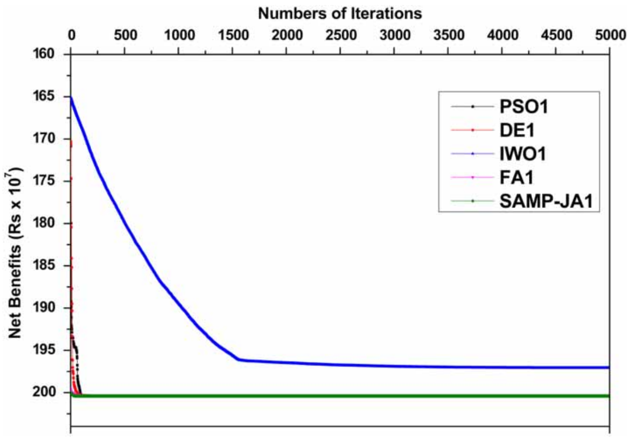

Figure 7 | Convergence plot of the different algorithms for model-1 up to 5,000 iterations.

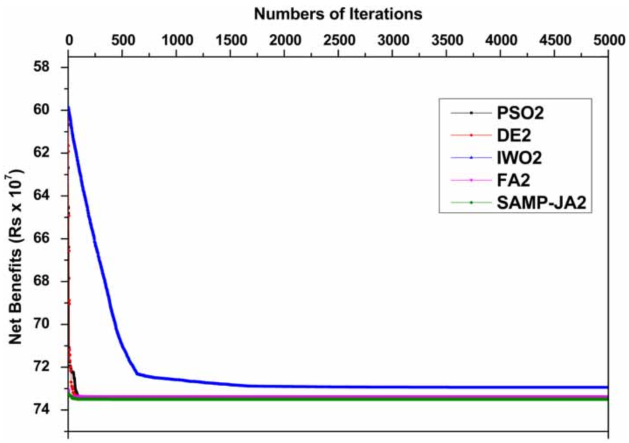

Figure 8 | Convergence plot of the different algorithms for model-2 up to 5,000 iterations.

Compared to the real income produced during 2010 and 2011, SAMP-JA1 was found to be 67.38% and 65.72% better, and SAMP-JA2 was 11.04% and 6.52% better than the real income generated in 2010 and 2011, respectively. Figures 7 and 8 demonstrate the convergence plot for model-1 and model-2 for all algorithms running up to 5,000 iterations. For model-1, it was observed that at 189 iterations SAMP-JA1 obtained the global optimum solution as 200.4017, and at 303 iterations PSO1 obtained the global optimum solution as 200.4017. At 5,000 iterations, DE1, FA1 and IWO1 obtained an optimal solutions as 200.4014, 200.307 and 197.0938, respectively. For model-2, it was observed that at 412 iterations SAMP-JA2 obtained the global optimum solution as 73.4872, and at 3,381 iterations PSO2 obtained the global optimum solution as 73.4872. At 5,000 iterations, DE2, FA2 and IWO2 obtained the optimum solution as 73.4869, 73.3535 and 72.9611, respectively. Model-1 can provide maximum net benefits, but more irrigation releases are needed compared to model-2. During less inflow, the dam officials are restrained to releases from the dam, and time model-2 was more suitable.

Total cropping area

The maximum cultivated areas proposed by SAMP-JA1, DE1, PSO1, FA1 and IWO1 for model-1 were 20,598, 20,579, 20,527, 20,527 and 14,685 ha, respectively. While the maximum cultivated areas proposed by SAMP-JA2, DE2, PSO2, FA2 and IWO2 for model-2 were 8,754, 8,722, 8,722, 8,712 and 7,546 ha, respectively. The optimized crop area obtained for model-1 using SAMP-JA1, DE1, PSO1, FA1 and IWO1 for both canals is shown in Table 6. The crop area acquired for groundnut by SAMP-JA1 in the RBMC was 1.12%, 5.54%, and 1.12% higher than PSO1, IWO1 and FA1, respectively, and 78.17% higher than IWO1 in the LBMC. The optimal cropping area for other crops are bolded values in Table 6. SAMP-JA1 clearly performed better than other algorithms.

Table 7 shows the optimized cropping area for model-2 for both canals using SAMP-JA2, DE2, PSO2, FA2 and IWO2. The cultivation areas obtained for groundnut by SAMP-JA2 in the RBMC were 1.64% and 3.77% more than PSO2 and IWO2, and in the LBMC 14.07% and 62.09% more than PSO2 and IWO2, respectively. The optimal cropping area for other crops are bolded values in Table 7. The cropping area has directly affected the net benefits for both the models resulting in higher net revenues for SAMP-JA compared to PSO, DE, FA and IWO.

Optimal cropping pattern

The cropping pattern acquired from SAMP-JA1, DE1, PSO1, FA1, IWO1, SAMP-JA2, DE2, PSO2, FA2 and IWO2 was compared with the actual cropping patterns for both RBMC and LBMC in 2010 and 2011. Tables 6 and 7 also show the comparison of optimal cropping patterns for model-1 and 2 with the actual cropping pattern for 2010 and 2011. It was found in the RBMC that SAMP-JA1 and DE1 had increased groundnut, sugarcane and vegetable areas by 4.21%, 35.13% and 88.20%, respectively. Cotton and castor areas were increased by 78.41% and 71.61% using SAMP-JA1, PSO1, DE1 and FA1. Millet grains and wheat areas were increased by 100% and 54.94% using SAMP-JA1. The percentage increase of SAMP-JA1 with the actual cropping pattern of 2010 and 2011 in the RBMC is presented in Table 8(a).

Table 6 | Comparison of the optimal cropping area for model-1 obtained from different algorithms

Canal	Variables	Crop (ha)	PSO1	DE1	SAMP-JA1	IWO1	FA1	2010	2011
RBMC	Rk1	Groundnut	202.81	**205.10**	**205.10**	193.73	202.81	187.91	205.04
	Rr2	Wheat	110.84	**137.25**	**138.77**	35.98	110.84	86.93	38.12
	Rp3	Sugarcane	2,033.78	2,035.49	**2,035.66**	1,586.59	2,033.78	1,194.45	1,446.65
	Rp4	Banana	3,060.59	**3,060.83**	**3,060.83**	2,789.33	3,060.59	1,299.94	844.86
	Rp5	Millet grains/corn	792.67	789.50	**805.30**	328.37	792.67	0	0
	Rp6	Pulses	179.50	178.52	**179.68**	39.28	179.50	124.01	146.98
	Rp7	Vegetables	29.91	**50.19**	**50.19**	48.86	29.91	7.61	4.23
	Rt8	Cotton	1,498.06	**1,498.08**	**1,498.08**	1,496.59	1,498.06	424.67	222.32
	Rt9	Castor	**113.99**	113.94	**113.99**	51.70	**113.99**	46.99	17.73
LBMC	Lk1	Groundnut	**125.87**	**125.87**	**125.87**	27.48	**125.87**	**125.87**	**125.87**
	Lr2	Wheat	**429.42**	**429.42**	**429.42**	375.90	**429.42**	153.37	230.04
	Lp3	Sugarcane	7,046.61	**7,047.80**	**7,047.80**	4,180.18	7,046.61	3,596.62	6,263.73
	Lp4	Banana	**1,283.92**	**1,283.92**	**1,283.92**	1,283.92	**1,283.92**	411.18	228.93
	Lp5	Millet grains/corn	**982.36**	**982.36**	**982.36**	161.16	**982.36**	0	90.8
	Lp6	Pulses	614.63	614.99	**615.67**	263.16	614.63	82.71	27
	Lp7	Vegetables	82.43	**86.36**	**86.36**	30.57	82.43	45.66	0
	Lt8	Cotton	**1,799.47**	**1,799.47**	**1,799.47**	1,652.78	**1,799.47**	183.14	182.13
	Lt9	Castor	**139.79**	**139.79**	**139.79**	**139.79**	**139.79**	87.96	87.64

Note: Bold values indicate the optimal results in model-1.

Table 7 | Comparison of the optimal cropping area for model-2 obtained from different algorithms

Canal	Variables	Crop (ha)	PSO2	DE2	SAMP-JA2	IWO2	FA2	2010	2011
RBMC	Rk1	Groundnut	111.84	113.70	113.70	109.41	113.69	187.91	**205.04**
	Rr2	Wheat	40.79	37.04	**44.28**	33.28	40.28	86.93	38.12
	Rp3	Sugarcane	1,002.55	1,001.52	1,002.82	804.72	1,001.82	1,194.45	**1,446.65**
	Rp4	Banana	1,125.58	1,125.73	1,125.79	1,066.95	1,124.79	**1,299.94**	844.86
	Rp5	Millet grains/corn	287.86	277.36	**288.60**	281.47	270.60	0	0
	Rp6	Pulses	179.54	179.26	**179.68**	72.23	178.68	124.01	146.98
	Rp7	Vegetables	7.85	3.38	**10.96**	8.98	10.87	7.61	4.23
	Rt8	Cotton	**489.07**	**489.07**	**489.07**	427.38	**489.07**	424.67	222.32
	Rt9	Castor	14.54	14.01	14.54	**14.55**	14.54	46.99	17.73
LBMC	Lk1	Groundnut	34.88	40.59	40.59	15.39	40.59	**125.87**	**125.87**
	Lr2	Wheat	179.73	179.73	179.73	157.72	179.73	153.37	**230.04**
	Lp3	Sugarcane	3,827.02	3,827.02	3,827.02	3,480.54	3,827.02	3,596.62	**6,263.73**
	Lp4	Banana	422.05	**422.12**	**422.12**	348.05	**422.12**	411.18	228.93
	Lp5	Millet grains/corn	325.53	326.20	**326.66**	231.53	325.66	0	90.8
	Lp6	Pulses	**163.79**	**163.79**	**163.79**	89.46	**163.79**	82.71	27
	Lp7	Vegetables	11.12	23.38	**25.56**	24.60	10.25	45.66	0
	Lt8	Cotton	451.35	**451.45**	**451.45**	355.11	**451.45**	183.14	182.13
	Lt9	Castor	47.40	46.85	47.59	24.61	46.59	**87.96**	**87.64**

Note: Bold values indicate the optimal results in model-2.

Table 8 | Percentage increase of SAMP-JA1 with the actual cropping pattern

(a) Percentage increase of SAMP-JA1 with the actual cropping pattern in the RBMC

Canal	Variables	Crop	2010 compared with SAMP-JA1	2011 compared with SAMP-JA1
RBMC	Rk1	Groundnut	8.38	0.03
	Rr2	Wheat	37.36	72.53
	Rp3	Sugarcane	41.32	28.93
	Rp4	Banana	57.53	72.40
	Rp5	Millet grains	100.00	100.00
	Rp6	Pulses	30.98	18.20
	Rp7	Vegetables	84.84	91.57
	Rt8	Cotton	71.65	85.16
	Rt9	Castor	58.78	84.45

(b) Percentage increase of SAMP-JA1 with the actual cropping pattern in the LBMC

Canal	Variables	Crop	2010 compared with SAMP-JA1	2011 compared with SAMP-JA1
LBMC	Lk1	Groundnut	0.00	0.00
	Lr2	Wheat	64.28	46.43
	Lp3	Sugarcane	48.97	11.13
	Lp4	Banana	67.97	82.17
	Lp5	Millet grains	100.00	90.76
	Lp6	Pulses	86.57	95.61
	Lp7	Vegetables	47.13	100.00
	Lt8	Cotton	89.82	89.88
	Lt9	Castor	37.08	37.31

In the LBMC, wheat, banana, cotton and castor areas were found to have increased by 55.36%, 75.07%, 89.85% and 37.19% using SAMP-JA1, PSO1, DE1 and FA1. Areas of cultivation of sugarcane and vegetables had increased by 30.05% and 73.56% using SAMP-JA1 and DE1. Table 8(b) presents the percentage increase of SAMP-JA1 with the actual cropping pattern in LBMC. Since the crop type in both the canals was the same, the results were the same except for the cultivated area. The main difference between SAMP-JA1, PSO1, DE1 and FA1 from IWO1 was that the cropping areas for SAMP-JA1, PSO1, DE1 and FA1 were more for groundnut, wheat, sugarcane, banana, millet grains and castor. As a result, the net benefits obtained by SAMP-JA1, PSO1, DE1 and FA1 were more compared with IWO1.

CONCLUSIONS

Most heuristic and evolutionary algorithms require good tuning of algorithm-specific parameters other than common control parameters. A slight change in the tuning parameters affects the overall performance of the algorithms. This paper suggested a multi-population-based JA that does not require algorithm-specific parameters. The multi-population enables us to divide the population into sub-population groups and helps to improve the search space diversity. The algorithm's performance is highly dependent on the selected numbers of the sub-population. To solve this problem, a self-adaptive model is used, i.e. SAMP-JA. In this research, SAMP-JA is used to optimize the optimum cropping pattern and area for the Karjan reservoir project in order to maximize the net benefit. To verify the precision, SAMP-JA's performance was compared with PSO, DE, IWO, FA, JA, TLBO, EJA and ETLBO algorithms. Following are the conclusions drawn from this study:

(i) In model-1, SAMP-JA1 had better net benefits than IWO1, LP1, TLBO1, FA1, PSO1, DE1, ETLBO1 and JA1. The results by SAMP-JA1 and EJA1 were nearly the same, but there was a faster convergence rate for SAMP-JA1. In model-2, SAMP-JA2 had better net benefits than IWO2, LP2, TLBO2, FA2, ETLBO2, DE2, PSO2, JA2 and EJA2. Since the unit was in (Rs $\times$ 10^7), even a small change in results led to a large change in net benefits. In addition, due to the internal parameters, the time it took to tune IWO, FA, PSO and DE was very high to get the optimum solution.
(ii) While comparing the optimized cropping area, it was clear that SAMP-JA1 performed better than other model-1 algorithms, and that SAMP-JA2 performed better for model-2.
(iii) The cropping area has a direct effect on net benefits. Thus, it leads to a higher net benefit for SAMP-JA1, PSO1, DE1 and FA1 as compared to IWO1.
(iv) The optimal cropping pattern has resulted in a substantial rise in groundnut, wheat, sugarcane, banana, millet grains and castor crops production by SAMP-JA1, PSO1, DE1 and FA1, which in turn generated more net benefits compared to IWO1, SAMP-JA2, DE2, PSO2, FA2, IWO2 and actual condition.

Based on the results of both the models, it is concluded that SAMP-JA converses faster and performs better than PSO, DE, IWO, FA, JA, TLBO, LP, EJA and ETLBO. In addition, to operate the models as a multi-objective reservoir, future studies are needed to consider the present and potential scenarios of power generation capability.

ACKNOWLEDGEMENTS

The authors acknowledge the helpful comments and suggestions made by anonymous reviewers to enhance the manuscript. They would like to thank the authorities of the Sardar Vallabhbhai National Institute of Technology (SVNIT) and the PG section of water resources engineering for their valuable support.

REFERENCES

Adeyemo, J. & Otieno, F. 2010 Differential evolution algorithm for solving multi-objective crop planning model. *Agric. Water Manag.* **97** (6), 848–856.
Azaiez, M. N., Hariga, M. & Al-Harkan, I. 2005 A chance-constrained multi-period model for a special multi-reservoir system. *Comput. Oper. Res.* **32** (5), 1337–1351.
Dutta, S., Sahoo, B. C., Mishra, R. & Acharya, S. 2016 Fuzzy stochastic genetic algorithm for obtaining optimum crops pattern and water balance in a farm. *Water Resour. Manag.* **30** (12), 4097–4123.
Eberhart, R. & Kennedy, J. 1995 A new optimizer using particle swarm theory. In: *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*. IEEE, Silver Spring, MD, pp. 39–43.
Fallah-Mehdipour, E., Bozorg Haddad, O. & Mariño, M. A. 2013 Extraction of optimal operation rules in an aquifer-dam system: genetic programming approach. *J. Irrig. Drain. Eng.* **139** (10), 872–879.
Fasakhodi, A. A., Nouri, S. H. & Amini, M. 2010 Water resources sustainability and optimal cropping pattern in farming systems; a multi-objective fractional goal programming approach. *Water Resour. Manag.* **24** (15), 4639–4657.
Fayaed, S. S., El-Shafie, A. & Jaafar, O. 2013 Reservoir-system simulation and optimization techniques. *Stoch. Environ. Res. Risk Assess.* **27** (7), 1751–1772.
Georgiou, P. E. & Papamichail, D. M. 2008 Optimization model of an irrigation reservoir for water allocation and crop planning under various weather conditions. *Irrig. Sci.* **26** (6), 487–504.
Ghahraman, B. & Sepaskhah, A. R. 2002 Optimal allocation of water from a single purpose reservoir to an irrigation project with pre-determined multiple cropping patterns. *Irrig. Sci.* **21** (3), 127–137.
Habibi Davijani, M., Banihabib, M. E., Nadjafzadeh Anvar, A. & Hashemi, S. R. 2016 Multi-objective optimization model for the allocation of water resources in arid regions based on the

maximization of socioeconomic efficiency. *Water Resour. Manag.* **30** (3), 927–946.
Hossain, M. S. & El-shafie, A. 2013 Intelligent systems in optimizing reservoir operation policy: a review. *Water Resour. Manag.* **27** (9), 3387–3407.
Jothiprakash, V. & Arunkumar, R. 2013 Optimization of hydropower reservoir using evolutionary algorithms coupled with chaos. *Water Resour. Manag.* **27** (7), 1963–1979.
Jothiprakash, V., Arunkumar, R. & Ashok Rajan, A. 2011 Optimal crop planning using a chance constrained linear programming model. *Water Policy* **13** (5), 734–749.
Kumar, V. & Yadav, S. M. 2018 Optimization of reservoir operation with a new approach in evolutionary computation using TLBO algorithm and Jaya algorithm. *Water Resour. Manag.* **32** (13), 4375–4391.
Kumar, V. & Yadav, S. M. 2019 Optimization of cropping patterns using elitist-Jaya and elitist-TLBO algorithms. *Water Resour. Manag.* **33** (5), 1817–1833.
Lalehzari, R., Boroomand Nasab, S., Moazed, H. & Haghighi, A. 2016 Multiobjective management of water allocation to sustainable irrigation planning and optimal cropping pattern. *J. Irrig. Drain. Eng.* **142** (1), 05015008.
Mainuddin, M., Das Gupta, A. & Onta, P. R. 1997 Optimal crop planning model for an existing groundwater irrigation project in Thailand. *Agric. Water Manag.* **33** (1), 43–62.
Mehrabian, A. R. & Lucas, C. 2006 A novel numerical optimization algorithm inspired from weed colonization. *Ecol. Inform.* **1** (4), 355–366.
Moradi-Jalal, M., Bozorg Haddad, O., Karney, B. W. & Mariño, M. A. 2007 Reservoir operation in assigning optimal multi-crop irrigation areas. *Agric. Water Manag.* **90** (1–2), 149–159.
Noory, H., Liaghat, A. M., Parsinejad, M. & Haddad, O. B. 2012 Optimizing irrigation water allocation and multicrop planning using discrete PSO algorithm. *J. Irrig. Drain. Eng.* **138** (5), 437–444.
Paul, S., Panda, S. N. & Kumar, D. N. 2000 Optimal irrigation allocation: a multilevel approach. *J. Irrig. Drain. Eng.* **126** (3), 149–156.
Raju, K. S. & Kumar, D. N. 2000 Irrigation planning of sri ram sagar project using multiobjective fuzzy linear programming. *ISH J. Hydraul. Eng.* **6** (1), 55–63.
Rao, R. V. & Patel, V. 2013 An improved teaching-learning-based optimization algorithm for solving unconstrained optimization problems. *Sci. Iran.* **20** (3), 710–720.
Rao, R. V., Savsani, V. J. & Vakharia, D. P. 2011 Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. *Comput. Des.* **43** (3), 303–315.
Reddy, M. J. & Kumar, D. N. 2008 Evolving strategies for crop planning and operation of irrigation reservoir system using multi-objective differential evolution. *Irrig. Sci.* **26** (2), 177–190.
Regulwar, D. G. & Gurav, J. B. 2011 Irrigation planning under uncertainty – a multi objective fuzzy linear programming approach. *Water Resour. Manag.* **25** (5), 1387–1416.
Sahoo, B., Lohani, A. K. & Sahu, R. K. 2006 Fuzzy multiobjective and linear programming based management models for optimal land-water-crop system planning. *Water Resour. Manag.* **20** (6), 931–948.
Sethi, L. N., Kumar, D. N., Panda, S. N. & Mal, B. C. 2002 Optimal crop planning and conjunctive use of water resources in a coastal river basin. *Water Resour. Manag.* **16** (2), 145–169.
Sethi, L. N., Panda, S. N. & Nayak, M. K. 2006 Optimal crop planning and water resources allocation in a coastal groundwater basin, Orissa, India. *Agric. Water Manag.* **83** (3), 209–220.
Singh, A. 2012 Optimal allocation of resources for the maximization of net agricultural return. *J. Irrig. Drain. Eng.* **138** (9), 830–836.
Singh, D. K., Jaiswal, C. S., Reddy, K. S., Singh, R. M. & Bhandarkar, D. M. 2001 Optimal cropping pattern in a canal command area. *Agric. Water Manag.* **50** (1), 1–8.
Srivastava, P. & Singh, R. M. 2015 Optimization of cropping pattern in a canal command area using fuzzy programming approach. *Water Resour. Manag.* **29** (12), 4481–4500.
Storn, R. & Price, K. 1996 Minimizing the real functions of the ICEC'96 contest by differential evolution. In: *Proceedings of IEEE International Conference on Evolutionary Computation*. IEEE, Silver Spring, MD, pp. 842–844.
Subramanya, K. 2013 *Engineering Hydrology*. Tata McGraw-Hill Education, Chennai, pp. 45–60.
Vasan, A. & Raju, K. S. 2009 Comparative analysis of simulated annealing, simulated quenching and genetic algorithms for optimal reservoir operation. *Appl. Soft Comput. J.* **9** (1), 274–281.
Venkata Rao, R. 2016 Jaya: a simple and new optimization algorithm for solving constrained and unconstrained optimization problems. *Int. J. Ind. Eng. Comput.* **7** (1), 19–34.
Venkata Rao, R. & Saroj, A. 2017 A self-adaptive multi-population based Jaya algorithm for engineering optimization. *Swarm Evol. Comput.* **37** (4), 1–26.
Yang, X. S. 2008 *Nature-Inspired Metaheuristic Algorithms*, 1st edn. Luniver Press, UK.
Yang, C. C., Chang, L. C., Chen, C. S. & Yeh, M. S. 2009 Multi-objective planning for conjunctive use of surface and subsurface water using genetic algorithm and dynamics programming. *Water Resour. Manag.* **23** (3), 417–437.
Zeng, X., Kang, S., Li, F., Zhang, L. & Guo, P. 2010 Fuzzy multi-objective linear programming applying to crop area planning. *Agric. Water Manag.* **98** (1), 134–142.

First received 21 April 2019; accepted in revised form 9 October 2019. Available online 7 November 2019