Memetic Computing
https://doi.org/10.1007/s12293-021-00327-8

REGULAR RESEARCH PAPER

Large scale reservoir operation through integrated meta-heuristic approach

Bilal[1] · Millie Pant[1] · Deepti Rani[2]

Received: 17 July 2020 / Accepted: 30 January 2021

Abstract
Reservoir optimization models are often large-scale, having a complex, nonlinear, multi-dimensional structure, which poses a challenge for classical methods for their solution. This has encouraged the researchers to focus on Meta-heuristic which due to their flexible and adaptive nature have been successful in solving a plethora of real-life optimization problems. This study brings forward an implementation and comparison of six well-known Meta-heuristics namely: Simulated Annealing, Genetic Algorithms, Particle Swarm Optimization, Differential Evolution, Artificial Bee Colony, and Cuckoo Search and an integrated version of these algorithms with dynamic programming for optimizing the reservoir operations policy. In addition, two adaptive variants of DE named: FCADE2 and SaDE are also considered for the comparison. The case study considered for Mula reservoir supplying water to Major Irrigation Project on River Mula (Godavari basin), Ahmednagar district, Maharashtra, India. The objective is to determine the optimum release policy for Mula reservoir. Performance of the algorithms is analysed on two data sets (1) single year and (2) 30-years.

Keywords Meta-heuristic algorithms · Reservoir operation · Release · Demand · Storage

1 Introduction

Reservoir operation rules may be defined by a function where the release of water from a reservoir for the given time interval is computed through the values of current reservoir storage, and current and expected demands and inflows. Common objectives for reservoir systems are optimization of water supply for different purposes like irrigation, industries, and domestic; generation of hydropower; water quality improvement, preservation, and enhancement of aquatic animals; flood control; navigation, etc. Associated constraints are due to the continuity equation, maximum and minimum storages in the reservoirs, maximum and minimum releases from the reservoirs, and certain case-specific obligations. As per the literature, two commonly exploited classical methods for reservoir systems are Linear Programming (LP) and Dynamic Programming (DP) methods [1, 2].

However, with the passage of time and because of the growing complexity of the real-life models, researchers have also started focusing on Meta-heuristics algorithms, because of their flexibility and their natural adaptability to cope up with different types of problems through minor modifications in the structure of the algorithm [2]. Several Meta-heuristics have been proposed so far for the optimization of reservoir operations. Popular choices include Differential Evolution (DE), Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC), Simulated Annealing (SA), and Cuckoo Search (CS).

Besides classical and Meta-heuristics, the solution methodology that has gained attention are hybrid methods, integration of classical and Meta-heuristics, where the algorithms are designed to synergize the features of the two. Successful application of classical, Meta-heuristics, and hybrid variants is shown on hypothetical as well as real-life examples.

The present study proposes an integration of Dynamic Programming (DP) with selected Meta-heuristics namely

✉ Bilal
bilal25iitr@gmail.com

Millie Pant
millifpt@iitr.ac.in

Deepti Rani
deeptinatyan@yahoo.com

1 Department of Applied Science and Engineering, Indian Institute of Technology, Roorkee 247667, India

2 National Institute of Hydrology, Roorkee, India

Published online: 02 March 2021

PSO, DE, GA, ABC, CS, SA and also with adaptive DE variants FCADE2 and SaDE. The proposed hybrid variants are employed to obtain the optimal release policy for Mula reservoir, Maharashtra, India. The present article is segregated into six sections. Section 1 gives a brief literature review on reservoir operations. Section 2, describes the area of study. In Sect. 3, the mathematical formulation of the problem is discussed along with the solution methodology. Section 4, provides experimental settings and numerical results and analysis are given in Sect. 5. The final concluding remarks are in Sect. 6.

1.1 DP for reservoir operation

Literature suggests an extensive use of dynamic programming (DP) for reservoir operation studies and it is perhaps the most popular optimization algorithm besides LP for such problems [17, 23]. The application of DP for single and multi-reservoir systems has been widely discussed by researchers. One of the initial studies on discrete DE for multi-reservoir problems is given in Chow et al. [10], where the authors discussed the computational requirements of DP. Since DP suffers from an inherent drawback of the "curse of dimensionality", researchers have proposed enhanced versions for reducing the computational complexities of DP. Selected variants are state increment DP (SIDP) by Larson [5] in this study author uses the concept of state incrimination with dynamic programming, to find the optimum solution of fire power in two reservoir Hall et al. [6] applied the incremental DP (IDP) approach, In 1971 Heidari et al. [7] proposed a discrete differential DP (DDDP) technique for optimization of a water resources system, DP with successive approximation (DPSA) by Bellman and Dreyfus [3], and a multi reservoir system is optimized by Trott and Yeh [9], a variants of DP is used to solve the reservoir operation by Giles and Wunderlich [12]. Literature also reveals that DP/DDDP has been a popular choice among researchers for such type of optimization problems [8, 11, 18]. DPSA approach, proposed by Larson [5, 14] has also been studied by researchers for solving multi-reservoir operation problems [14, 15]. Labadie [13], introduced CSUDP, a generalized DP software, employing a combination of DPSA and IDP/DDDP techniques for multi-reservoir operation. Some other enhanced DP variants available in the literature are as follows: Kumar and Baliarsingh [16] introduced a new iterative algorithm named folded DP (FDP). Castelletti et al. [19] proposed Neuro-DP, a hybrid of Artificial Neural Networks (ANN) and DP, Tilmant, and Kelman [20] suggested stochastic dual DP (SDDP) and Srivastava and Awchi [22] provided two DP variants CODP and CIDP for reservoir optimization. Interested readers may also consult Nandlal and Bogardi [21] for DP models applied to various reservoir models around the world. Rani et. al. [23] provides a review on the simulation and optimization model. Some recent examples of the application of modified DP variants can be found in Xiang Li et al. [24], Shaokun He et al. [25], and Xiang Zeng et al. [26]. The region of study with corresponding DP references is provided in Table 1

1.2 Meta-heuristic for reservoir operation

Genetic Algorithms (GA) have been the most popular choice of Meta-heuristics for reservoir operations. Different researchers have shown the successful application of GA on test cases around the world. Jothiprakash and Shanthi [29] proposed the usage of GA for optimized reservoir operation for a single reservoir system and applied it on Pechiparai reservoir in Tamil Nadu, India. Kumar et al. [30], proposed GA for optimal allocation of water for crops and implemented it on a case study of Malaprabha reservoir, Karnataka State, India. Kerachian and Karamouz [32, 33] tackled the reservoir optimization model in Iran and proposed a blend of a simulation model and GA for optimizing the reservoir operation rules. Malekmohammadi et al. [40] also considered reservoirs in Iran on Bakhtiari and Dez River and proposed a combination of GA and K-nearest neighbor algorithm for optimizing the reservoir operation models. Hinçal et al. [43] proposed a real coded Genetic Algorithm on a reservoir over the Colorado River. A hybrid approach integrating DP with GA was proposed by [47, 52, 53] for determining the Optimum solution for reservoir operation problem. Researchers observed that hybridization helps in improving the solution quality. Besides GA, other Meta-heuristics applied by researchers include Simulated Annealing (SA), Differential Evolution, Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC), Cuckoo Search (CS), etc. Lakshmi Narasimhan et al. [31] proposed a modified differential evolution (MODE) and applied it on the Short-term scheduling of hydrothermal power systems with cascaded reservoirs. Reddy et al. [34] proposed a multi-objective variant of Differential Evolution for reservoir problems. Later the multi-objective variant of differential evolution was applied for Evolving strategies for crop planning and operation of irrigation reservoir system by Reddy et al. [36], Mandal and Chakraborty [38] proposed DE for Short-term combined economic emission scheduling of hydrothermal power systems with cascaded reservoirs. Regulwaret. al. [42] used DE, while Qin et al. [41], proposed Multi-objective cultured differential evolution. A review paper on multi-objective evolutionary algorithms is given by Adeyemo [44]. Use of Simulated Annealing for reservoir systems is shown by Khodabakhshi et al. [39] and Georgiou et al. [28], Teegavarapu, and Simonovic [27] and Kangrang and Compliew [46]. Baltar and Fontane [37] and Mehdipour et al. [45] proposed multi-objective PSO (MOPSO). Afshar [48] proposed constrained variants of PSO and Afshar et al. [49]

Table 1 Dynamic Programming and its variants on reservoir operation

References	DP variant	Region of study
Bellman and Dreyfus [3]	Dynamic programming	Test Problems
Larson [4]	Dynamic programming with successive approximations (DPSA)	Sample test system
Larson [5]	State increment dynamic programming	Test problems
Hall et al. [6]	Incremental dynamic programming (IDP)	Shasta and Folsom Reservoir, California
Heidari et al. [7]	Discrete differential dynamic programming	Test Problems
Fults and Hancock [8]	Incremental dynamic programming (IDP)	Shasta-Trinity system, California
Trott and Yeh [9]	Dynamic programming	Karun and Dez reservoir, Iran
Chow et al. [10]	Discrete differential dynamic programming(DDDP)	Test Problems
Fults et al. [11]	Incremental dynamic programming (IDP) with RecIP	Sample Test Example
Giles and Wunderlich [12]	Dynamic programming	TVA Reservoir
Labadie [13]	Generalized DP software package CSUDP	Sample test system
Shim et al. [14], Yi et al. [15]	Dynamic programming with successive approximations (DPSA)	Han River Basin in Korea
Kumar and Baliarsingh [16]	Iterative DP variant named Folded DP or FDP	hypothetical reservoir system
Labadie [17]	Various DP variants and its application	State of the art review
Yurtal et al. [18]	Dynamic programming	Lower Seyhan Basin, Turkey
Castelletti et al. [19]	Neuro-dynamic programming	Piave reservoir network, Northern Italy
Tilmant and Kelman [20]	Stochastic dual dynamic programming (SDDP)	South-eastern Anatolia Development Project, Turkey
Nandalal and Bogardi [21]	Dynamic Programming	Case studies around the world
Srivastava and Awchi [22]	Controlled Output Dynamic programing, Controlled Inventory Dynamic Programming	Mula Reservoir, India
Rani and Moreira [23]	Review of simulation–optimization modelling to reservoir system operation	Review paper
Li et al. [24]	Improved dynamic programming	Reservoir system, North-eastern China
He et al. [25]	A parallel dynamic programming algorithm	Upper Yangtze River, China
Zeng et al. [26]	A parallel dynamic programming algorithm	Yangtze River Basin, China

proposed a Multi-objective particle swarm optimization (MOPSO). The use of PSO is also shown by Chenari et al. [54]. Niu [60] proposed a parallel multi-objective particle swarm optimization while Zhang [50] proposed adaptive particle swarm optimization. Choong et al. [55] and Shimei [51] proposed the application of ABC for water reservoir systems. Perez et al. [58] proposed multi-objective ABC. Nieto et al. [59] proposed wavelet kernel SVM based with ABC algorithm. Ahmad et al. [56] used Artificial Bee Colony and Gravitational Search Algorithm. Chiu et al. [35] proposed a hybrid genetic algorithm–simulated annealing algorithm fuzzy programming for reservoir operation. Yasar [57] proposed a Cuckoo search for Optimization of Reservoir Operation. References related to the implementation of popular Meta-heuristic for reservoir operation studies are provided in Table 2.

2 Region of study: Mula project

The test case considered in the present study is that of Mula River project. It is a tributary of river Pravara and a subtributary of river Godavari, the third largest river of India. The Mula Project serves as a significant multi-usage project, fulfilling the irrigation as well as the industrial requirements of Ahmednagar district.

The geographic location of the Mula project is depicted in Fig. 1 (online source: (https://www.indianetzone.com/34/mula_dam_dnyaneshwarsagar_dam_maharashtra.htm). Monthly demand vs. monthly inflow are given in Fig. 2.

Selected noteworthy features of this project are listed below.

1. The yearly water requirement is 748×10^6 m^3. This includes an annual irrigation demand of 674×10^6 m^3 and an annual water supply–demand of 74×10^6 m^3.
2. Average yearly and the 75% water year dependable inflows to the reservoir are 825×10^6 m^3 and 688×10^6 m^3, respectively.
3. As per historic data, the estimates of the average and the 75% water year dependable inflows to the reservoir are obtained through the flow duration curve analysis.
4. The gross reservoir capacity is 736×10^6 m^3, out of which active storage capacity is 608×10^6 m^3, and dead storage capacity is 127×10^6 m^3.

Table 2 Meta-heuristic for Reservoir problems and their region of study in Chronological Order

References	Meta-heuristic used	Description	Region of Study
Teegavarapuand and. Simonovic [27]	Simulated Annealing	Optimal Operation of Reservoir Systems using	Four hydropower generating reservoirs, Manitoba, Canada
Georgiou et al. [28]	Simulated Annealing	Reservoir optimization for multi-crop cultivation area selection	Havrias river, Greece
Jothiprakash and Shanthi [29]	Genetic Algorithm	Optimized reservoir operation for a single reservoir system	Pechiparai reservoir in Tamil Nadu, India
Kumar et al. [30]	Genetic Algorithm	Optimized reservoir system for optimal allocation of water for crops	Malaprabha reservoir, Karnataka State, India
Lakshminarasimman [31]	Modified differential evolution	Reservoir optimization	Sample test system
Kerachian and Karamouz [32, 33]	Simulation modal + GA	Reservoir optimization	15-Khordad Reservoir, Iran
Reddy et al. [34]	Multi-objective Differential Evolution	Reservoir problem	Hirakud Reservoir, Orissa state, India
Chiu et al. [35]	Hybrid genetic algorithm–simulated annealing algorithm fuzzy programming	Reservoir operation	Shihmen Reservoir, Taiwan
Reddy et al. [36]	Multi-objective differential evolution	Evolving strategies for crop planning and operation of the irrigation reservoir system	Malaprabha reservoir, Karnataka State, India
Baltar and Fontane [37]	Multi-objective PSO (MOPSO)	Multipurpose reservoir operations	Sample test system
Mandal and Chakraborty [38]	Differential evolution	Short-term combined economic emission scheduling of hydrothermal power systems with cascaded reservoirs	Sample test system
Khodabakhshi et al. [39]	Simulated Annealing	Reservoir optimization	Sirvan River basin, Iran
Malekmohammadi et al. [40]	combination of GA and KNN	Optimization models for reservoir operation	Bakhtiari and Dez River-Reservoir, Iran
Hui et al. [41]	Multi-objective cultured differential evolution	For generating optimal trade-offs in reservoir flood control operation	Yangtze River, Yangtze city, China
Regulwar et al. [42]	Differential Evolution (DE)	FOR the optimal operation of multipurpose reservoir	Jayakwadi project stage-I, river Godavari, in Maharashtra State, India
Hinçal et al. [43]	GA	To maximize the total energy production of a reservoir	Colorado River, North America
Adeyemo [44]	Multi-objective evolutionary algorithms	Review paper	Vaalharts irrigation scheme (VIS), South Africa
Mehdipour et al. [45]	MOPSO	Optimization of multipurpose multi-reservoir operations	Sample test system
Kangrang et al. [46]	Simulated annealing	Optimal reservoir rule curves	Bhumibol and Sirikit reservoirs, Thailand
Li et al. [47]	GA with DP	Reservoir optimization	Mula reservoir, India
Afshar [48]	PSO	A iran based Reservoir optimization	Dez reservoir, Iran
Afshar et al. [49]	MOPSO	To optimize the water quality of Karkheh river in iran	Karkheh River, Iran
Zhang [50]	Adaptive particle swarm optimization	Reservoir operation optimization	Sample test system
Choong and shafie [51]	ABC	Optimizing release policy of reservoir operations	Review
Rani and Srivastava [52], Rani et al. [53]	DP with GA	The optimum solution for reservoir operation problem	Mula reservoir, India
Chenari et al. [54]	PSO	Normal and drought conditions are considered in this study	Mahabad reservoir dam, Iran

Table 2 (continued)

References	Meta-heuristic used	Description	Region of Study
Shi-Mei Choong et al. [55]	ABC	Optimizing release policy of reservoir operations:	Chenderoh Dam, Malaysia
Asmadi Ahmad et al. [56]	Artificial Bee Colony (ACO) and Gravitational Search Algorithm (GSA)	Reservoir Optimization	TimahTasoh Dam, Malaysia
Yasar [57]	Cuckoo search	Optimization of Reservoir Operation	Adiguzel dam, Denizli, Turkey
Perez et al. [58]	Multi-objective ABC	In river basins, water quality monitoring networks	Great Fish River basin, South Africa
Nieto et al. [59]	Wavelet kernel SVM based with ABC algorithm	Predicting the cyanotoxin content in a reservoir	Trasona reservoir, Northern Spain
Niu [60]	Parallel multi-objective particle swarm optimization	Cascade hydropower reservoir operation	Lancang cascaded hydropower system, China

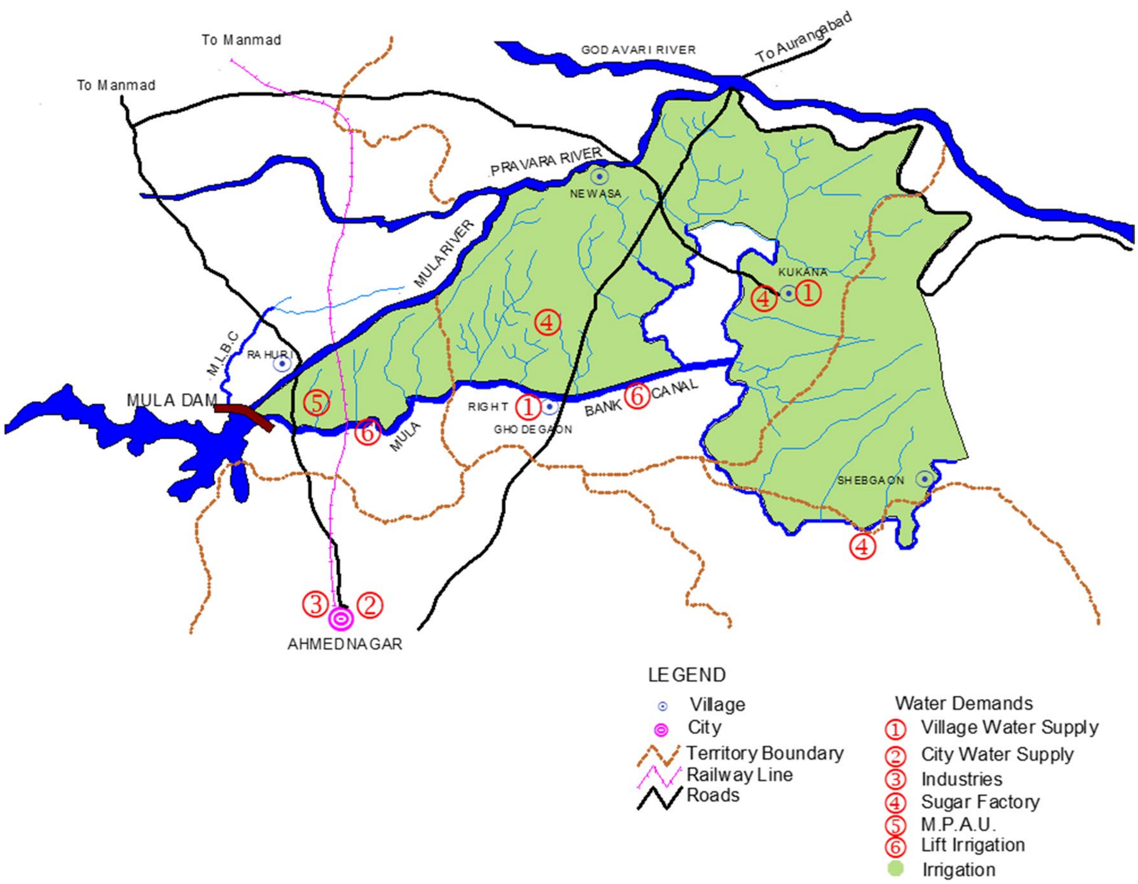


Fig. 1 Geographic location of Mula project

5. The Mula project does not have arrangements for flood storage

2.1 Reservoir policy

Reservoir Policy, relates to a collection of rules that define, under certain conditions, the amount of water to be collected, released or removed from a reservoir or a reservoir system. In the present study, the decision variables for the operating policy are presumed to be amount of water released from the reservoir at each point. The inflow data is given in "Appendix A". It shows that the reservoir would generally fill up during July to October and the quantity of water would start depleting during November to May. Therefore, reservoir is

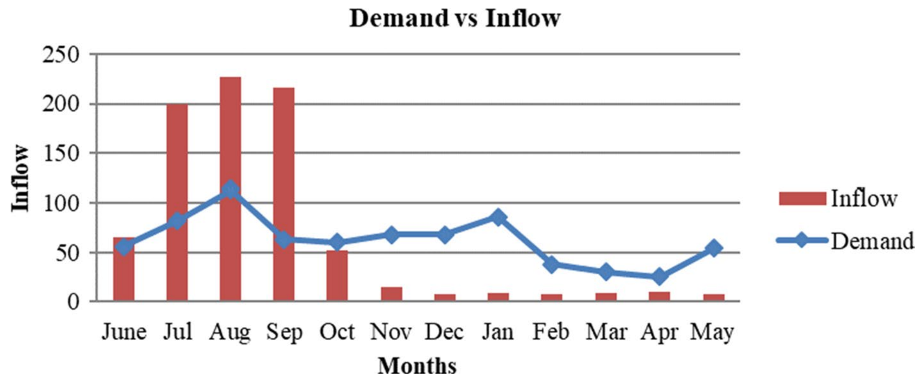


Fig. 2 Average monthly inflow and demand for 1 year

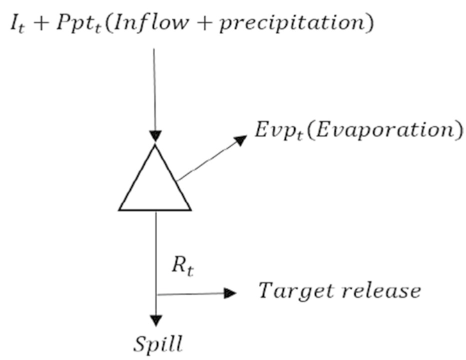


Fig. 3 Variables linked with a reservoir operation problem

considered to be empty in the beginning of June in the reservoir operation model. In this paper all the calculations are done from June onwards, keeping the above fact in mind.

3 Mathematical formulation and solution methodology

The optimization model determines the best and most efficient release combination, which minimises the squaredeviation from the release goal. The mathematical model consists of bound constraints in the form of upper and lower release and storage limits.

Further, the model is accompanied by a set of constraints that should be satisfied for each period owing to the continuity equation. The variables linked with a single reservoir can be seen in Fig. 3.

3.1 Optimization model

The objective function is given through Eq. (1). It represents the minimization of the sum of squared deviation from the target demands:

$$g_t(R_t) = Min \sum_{t=1}^{N} (Q_t - D_t)^2 \tag{1}$$

where t represents the time-period; g_t (R_t) indicates the release at t; Q_t gives the release for period t and D_t provides the target demand for time period t.

The objective function given through Eq. (1) is assisted with certain constraints described below:

Continuity equation:

$$S_{t+1} = S_t + I_t - Q_t - Evp_t \quad \forall t = 1, \dots, N \tag{2}$$

where S_t, I_t and R_t the storage, inflow, and releases respectively for the given reservoir at time period t; N, the number of time steps used in the model; Evp_t, evaporation from reservoir surface during time period t.

Bound constraints:

Limits on storage:

$$S_{\min} \le S_t \le S_{\max} \quad \forall t = 1, \dots, N \tag{3}$$

Limits on the release:

$$Q_{\min} \le Q_t \le Q_{\max} \quad \forall t = 1, \dots, N \tag{4}$$

Equations (3) and (4) ensure that storage and release are within the specified limits.

Solution Methodology

Three approaches are considered for solving the above mathematical model. First is the DP or the classical approach; the second is the Meta-heuristic approach, and the third is the hybrid approach by integrating DP with -heuristic. All the approaches are described briefly in the subsequent paragraphs.

Dynamic Programming: Dynamic programming (DP) is a special class of optimization technique, established by Richard Bellman [2] in the 1950s. The unique characteristic feature of DP is its multi-stage behavior, which transforms a complex problem into a sequence of simpler problems. This feature of DP is exploited to obtain the initial range of decision variables for the DE algorithm. The conventional DP reservoir operation model estimates the optimal monthly releases using monthly inflow to set a proper operation policy for the reservoir. In the present study, optimal releases obtained using DP were used as initial range of decision variables in metaheuristic algorithms used in this study.

Initially, the DP model estimates the optimal monthly releases using monthly inflow to set a proper operation policy for the reservoir. Two assumptions made before applying DP are:

(1) Current inflow will be treated as the input to the reservoir.
(2) The releases of the reservoir will include spill.

Recursive equation for the DP models is shown through Eq. (5):

$$f_r(S_r) = \min_{Q_r} \left[g_r(S_r, Q_r) + f_{r-1}(S_{r-1}) \right] \quad \forall r \tag{5}$$

Definition of return function is given through Eq. (6):

$$g_r(S_r, Q_r) = \left(Q_r - D_r\right)^2 \tag{6}$$

Remaining set of constraints are given through Eqs. (7)–(10):

1. Reservoir continuity (state transformation) equation

$$S_{r-1} = S_r + I_r - Q_r - EV_r, \quad \forall r \tag{7}$$

2. Spill from reservoir

$$Sp_r = Q_r - D_r \quad \text{if} \quad Q_r > D_r \quad \text{else} \quad Sp_r = 0 \quad \forall r \tag{8}$$

3. Non-negativity restrictions on state and decision variables and upper bound constraints on reservoir storages (states) limited to live storage capacity (Y_a).

$$Q_r \geq 0 \quad \forall r \tag{9}$$

$$0 \leq S_{r-1} \leq Ya \quad \forall r \tag{10}$$

where $fr(Sr)$, minimum optimal return function for the operation of the reservoir from all the r stages to go, given that reservoir storage (state) in r stages to go is Sr; g_r (Sr), return function at r stages togo; Sr, initial reservoir storage (state); S_{r-1}, resulting reservoir storage (state); Ir, Inflow to the reservoir; EVr, evaporation losses from the reservoir in r stages to go; Qr, total reservoir release including spill. This is also the decision variable for the DP model in r stages to go, Dr, Release target in r stages to go; Spr, reservoir spill in r stages to go.

Starting from the known initial storage, possible release decisions $\left\{Q_1, Q_1, \ldots, Q_N\right\}$ at each stage should be such that the final storage lies within the set of discrete states $\{S_{min} = 0, \delta, 2\,\delta, 3\,\delta, \ldots, m\,\delta = S_{max} = Y_a\}$.

If ‘δ’ is taken as a small quantity, it will lead to the increase in the number of states for a large-sized reservoir, resulting in larger computational requirements for DP. Contrarily, for coarse increment size, the accuracy of the results is likely to reduce, although the solution will be the true optimal solution for the given increment.

DP model solution provides a set of optimal releases $\{Q_1^*, Q_2^*, \ldots, Q_r^*, \ldots Q_N^*\}$ representing optimal policy for discrete increment ‘δ’ used to solve the DP model.

4 Meta-heuristic algorithms

Six well-known meta-heuristic algorithms are used in this study. These are: Genetic Algorithms (GA) [61], Simulated Annealing (SA) [62], Particle Swarm Optimization PSO [63], Differential Evolution DE [64], Artificial Bee Colony (ABC) [65], and Cuckoo Search CS [66]. All the considered Meta-heuristic algorithms are stochastic and iterative in nature and are population-based except for SA, which is a single solution-based. Characteristics of these algorithms are given in Table 3 and their Pseudo codes are given in “Appendix B”.

Table 3 List of Meta-heuristic and their characteristics

Meta-heuristic	References	Operators	Population-based	Control parameters
Genetic Algorithm	[61]	Crossover, Mutation, Selection	✓	Crossover rate, Mutation rate
Simulated Annealing	[62]	Temperature value		Temperature
Particle Swarm Optimization	[63]	Position, Velocity	✓	Inertia weight, Acceleration coefficient
Differential Evolution	[64]	Mutation, Crossover, Selection	✓	Scaling factor, Crossover rate
Artificial Bee Colony	[65]	Position of food	✓	Random number, Limit
Cuckoo Search	[66]	Position	✓	Step size, Switching parameter

5 Adaptive variants

In addition, to reach the best possible optimum solution of the reservoir problem, two adaptive variants of DE named FCADE2 [67] and SaDE [68] are also considered for the comparison. The considered algorithms populations based, stochastic and adaptive in nature.

6 Integrated approach

DP with a suitable discrete interval size, when applied to the given problem, provides an approximate optimal solution for the discrete values of the chosen storages. However, the actual optimal solution may fall somewhere within these discrete values of storage. The idea of the proposed integrated approach is to define a search space around this solution for the application of the Meta-heuristics algorithm.

The reduced potential region of search space expedites the search procedure of the Meta-heuristic algorithm thereby increasing overall computation efficiency, in terms of deriving optimal reservoir operating policy and the corresponding time. The probability of getting trapped into a local optimum, a common phenomenon for Meta-heuristic is also reduced because of the smaller search region. The objective function and constraints here are the same as in the mathematical model, except for the limits on releases, which are defined on the basis of the search space obtained through the DP model. The population is generated using real coding by randomly generating numbers within the upper and lower limits of the releases (can be seen in Table 10).

7 Experimental settings

a. **Computational environment**
   All the algorithms implemented in this study, are executed on a 64-bit operating system with window 10, Intel(R), Xeon(R) CPU E5-1650 v3 @ 3.50 GHz processor, 16.00 GB RAM.
   Coding of algorithms is done in MATLAB.
b. **Data**
   Two test cases are considered for this study. Initially, the analysis is done on data taken for 1 year i.e. 12 months. Later, the study is extended for 30 years (360 months). Data for both cases is taken from [52].
c. **Performance metrics**
   Algorithms used in this study are compared on the basis of the following performance metrics: objective function value, number of function evaluations (NFE), rank-based statistical analysis, computational time, and percent improvement in the objective function value. In addition, the results are shown graphically to interpret the behavior of algorithms.
d. **Control Parameters' settings for Algorithms:** Table 4 shows the parameter setting of control parameters for each algorithm.

For case 1, population size is kept as 20 for all population-based Meta-heuristics, and stopping criteria is the number of function evaluations set as 500. For case 2, population size is varied as 20, 50, and 100, and the stopping criteria are kept as 5000 and 50,000 number of function evaluations for all population sizes. For adaptive variants, population size is varied as 20, 50 and the stopping criteria are kept as 10,000, 50,000 and 100,000 number of function evaluations for all population sizes.

8 Numerical results and analysis

8.1 Test Case 1: reservoir policy evaluated for 1 year

Table 5, provides the limits on release based on the monthly demand for 1 year, which is highest in the month of August and is lowest in the month of April. Figure 4 shows the corresponding search space for the release policy. The simulation results are recorded in Table 6. For all the population-based Meta-heuristics, population size is kept as 20, and stopping criteria is the number of iterations set as 500.

Table 4 Parameters setting

Meta-heuristic	Control parameter
GA	Crossover rate = 0.7, Mutation rate = 0.3
SA	Temperature = 0.025
PSO	Inertia weight = 0.72, Acceleration coefficients = 1.494
DE	Scaling factor = 0.8, Crossover rate = 0.5
ABC	Random number = 1, limit = 50
CS	Step size = 1, switching parameter = 0.25
FCADE2	Initial crossover Rate = 0.1(adaptive), Scaling factor = 0.75
SaDE	Crossover Rate = 0.2(adaptive), Scaling factor = 0.5 (adaptive)

Table 5 Monthly demand policy for 1-year data

Months	June	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May
Lower limit	0	0	0	0	0	0	0	0	0	0	0	0
Upper limit	56.45	82.33	113.5	63.33	60.83	68.39	68.39	86.6	38.21	30.45	25.72	54.39

Fig. 4 Entire search region for releases for meta-heuristics

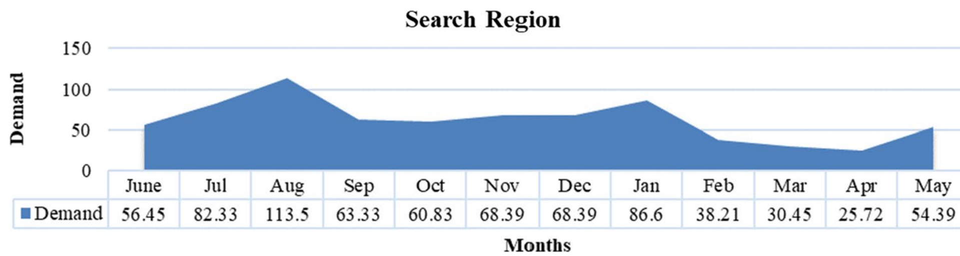


Table 6 Release Policies using Algorithms used in this study for 1-year data

Months\Algorithms	PSO	DE	ABC	GA	CS	SA
June	56.45	56.45	56.45	56.45	56.45	35.50
Jul	82.32988	82.33	82.33	82.33	82.33	76.95
Aug	113.5	113.5	113.5	113.5	113.5	100.63
Sep	63.33	63.33	63.33	63.33	63.33	48.24
Oct	60.82911	60.83	60.83	60.83	60.83	59.53
Nov	68.3901	68.39	68.39	68.39	68.39	65.33
Dec	68.38995	68.39	68.39	68.39	68.39	49.77
Jan	86.59995	86.6	86.6	86.6	86.6	67.71
Feb	38.21	38.21	38.21	38.21	38.21	31.10
Mar	30.44931	30.45	30.45	30.45	30.45	19.27
Apr	25.72	25.72	25.72	25.72	25.72	9.46
May	54.39003	54.39	54.39	54.39	54.39	37.98

Fig. 5 Comparison of Release policies for algorithms used in this study for 1-year data

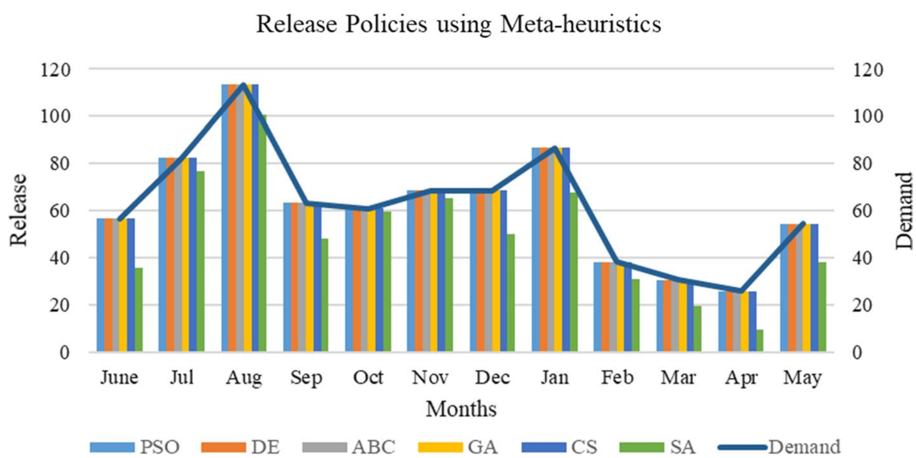


Figure 5 shows the convergence of Meta-heuristic algorithms selected for this study.

Table 6 shows that almost all the algorithms provide similar results fulfilling the release policy for each month. This however is not true for SA, which was not able to generate a release policy meeting the demand. For example, in the month of June, the release obtained from PSO, DE, ABC, GA, CS are 56.45 which is equal to demand while the release obtained from SA is 35.50 only, which is much lesser than the required demand in June. Similarly, in the month of December, the release policies obtained from each algorithm except SA are almost equal to the demand while for SA the release is quite less than the demand in that month. SA is a single solution based algorithm, while other meta-heuristics are population based means work on multiple solutions simultaneously. This inherent feature of Metaheuristics used in this study naturally give them an edge over SA. Moreover, although SA can explore the search space but is not good at exploitation, which sometimes leads to SA getting trapped in local minimum solution. The behavior of algorithms in terms of release policies is shown in Fig. 5. Table 7 shows the objective function values obtained through each algorithm while Fig. 6 shows the convergence behavior of the Meta-heuristics.

Analysis for case 1. Case 1 is a small-sized problem as the data considered is only for 1 year with only 12 decision variables. As expected, for such small size problems each of the population-based Meta-heuristic methods performed quite well except for SA, which did not perform adequately. Graphical results given in Fig. 6 indicate ABC to converge faster in comparison to other algorithms when data is considered for a single year.

8.2 Test Case 2: Reservoir policy for 30 years

To extend the study further, data set of 30 years varying from June 1972–73 to 2001–02 is taken. Consequently, the number of decision variables are increased considerably to 360. For this case, the objective function is first optimized through the six selected algorithms and then hybridized variants are employed for obtaining the optimal solution. Corresponding results are recorded separately in Tables 8 and 9 respectively.

Table 8 presents the results in terms of mean, maximum, minimum, standard deviation and time (in seconds) for each Meta-heuristic algorithm while keeping the NFE as 5000 and 50,000 and varying the population size as 20, 50, and 100 for both NFE. The best cost obtained out of 10 independent runs of each algorithm.

From this Table, it is observed that for NFE = 5000, DE obtained the best mean value = 130,880 for population size = 20. The standard deviation for DE was also the smallest in comparison to other Meta-heuristics. PSO gave the second-best mean value for the same settings of NFE and population size but the standard deviation for PSO was highest in comparison to other algorithms. On incrementing the population from 20 to 50 and then to 100, while keeping the NFE as 5000, it was observed that the mean best value deteriorated for both DE and PSO and while the standard deviation for DE increased on increasing the population

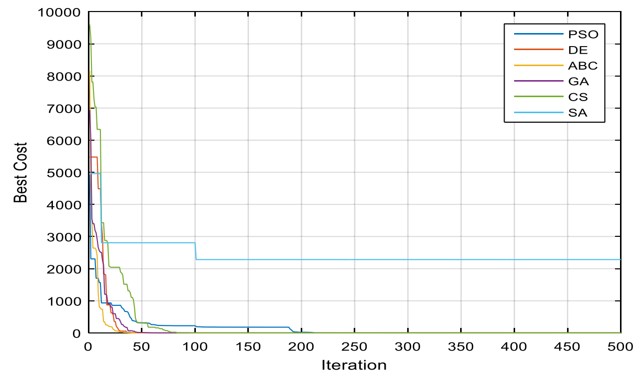


Fig. 6 Convergence graph for Algorithms used in this study for 1-year data

size, for PSO the standard deviation decreased on increasing the population size. Interestingly, the minimum value was obtained by PSO for these settings.

On increasing the NFE from 5000 to 50,000 and varying the population size as 20, 50, and 100, the best value (= 125,825.22) was obtained once again by DE for the smallest population size = 20, giving an improvement of around 4% in comparison to the mean best value obtained for smaller NFE of 5000 and same population size. The standard deviation recorded for DE was the smallest (= 0.07) and the best minimum value (= 129,825.13) was also obtained by DE for this setting (NFE = 5000 and population size = 20). For other algorithms, there appeared to be no definite pattern in the relation between NFE and population size. Also, Table 8 shows the time taken by each algorithm to attained optimum value. Figures 7(a), (b), (c) and 8(a), (b), (c) compares graphically, the performance of algorithms in terms of convergence, for all algorithms used in this study. Figure 7(a), (b), and (c) illustrates the results after setting the number of function evaluations as 5000 and by varying the population size as 20, 50, and 100. While Fig. 8(a), (b), and (c) shows the convergence when function evaluations are set as 50000 for population size 20, 50, and 100 respectively. These results are interpreted graphically through Fig. 9.

8.3 Statistical analysis

8.3.1 Confidence interval

Figure 10 depicts the confidence level for the mean objective function value with 95% confidence for each algorithm.

Table 7 Objective function valuefor 1-year data

Algorithms	PSO	DE	ABC	GA	CS	SA
Fitness value	1.3059e-06	0	0	0	0	2283.6553

Table 8 Comparison of algorithms based on Mean, Minimum, Maximum, Standard Deviation and computational time for 30-year data

Number of function evaluation	Population Size	Algorithms	Mean	Standard deviation	Minimum	Maximum	Time (s)
5000	20	ABC	134,091.86	176.66	133,690.61	134,342.49	2.05
		DE	130,880.69	161.90	130,689.35	131,161.28	1.31
		GA	135,631.43	276.98	135,285.11	136,049.64	0.83
		PSO	131,558.92	1135.19	129,776.77	134,376.27	1.61
		SA	138,420.62	231.12	137,977.14	138,869.14	0.60
		CS	135,687.11	174.90	135,327.37	135,918.70	0.83
	50	ABC	135,557.77	158.53	135,317.50	135,803.07	2.18
		DE	132,823.38	199.04	132,505.63	133,152.50	1.33
		GA	134,051.96	215.71	133,712.07	134,359.69	1.50
		PSO	132,006.66	580.34	131,071.54	132,958.33	1.69
		SA	138,410.77	283.99	137,874.90	138,792.61	0.66
		CS	136,462.09	234.88	136,114.23	136,758.28	0.84
	100	ABC	136,697.57	204.79	136,411.58	137,068.96	2.20
		DE	134,308.41	311.75	133,783.78	134,732.09	1.38
		GA	134,130.12	161.96	133,919.72	134,348.96	1.88
		PSO	132,935.60	424.14	132,539.73	133,992.86	1.78
		SA	138,445.66	256.72	138,094.56	139,002.03	0.67
		CS	136,890.84	344.22	136,490.63	137,438.32	0.86
50,000	20	ABC	132,479.54	15.06	132,456.56	132,509.69	20.1
		DE	129,825.22	0.07	129,825.13	129,825.36	8.3
		GA	134,016.53	311.88	133,729.63	134,604.42	8.2
		PSO	130,111.80	189.95	129,990.81	130,620.11	8.7
		SA	137,978.48	170.73	137,763.16	138,235.69	5.9
		CS	132,451.16	22.71	132,430.95	132,503.44	7.8
	50	ABC	132,818.51	26.58	132,763.28	132,854.91	20.6
		DE	129,887.37	7.35	129,877.19	129,900.12	10.16
		GA	132,811.04	182.84	132,637.87	133,279.65	9.19
		PSO	130,243.01	300.06	129,901.04	130,617.13	9.8
		SA	138,083.44	201.82	137,731.90	138,314.37	6.0
		CS	133,190.59	74.74	133,092.28	133,310.86	8.5
	100	ABC	133,256.37	66.65	133,156.46	133,362.62	20.9
		DE	130,283.85	29.72	130,223.84	130,323.18	11.3
		GA	132,479.50	42.48	132,418.41	132,562.17	9.24
		PSO	129,973.23	760.67	128,006.41	130,817.36	10.9
		SA	137,987.33	218.27	137,554.65	138,201.04	6.4
		CS	134,308.22	109.96	134,085.42	134,483.60	8.9

8.3.2 Friedman ranking test

The algorithms are ranked according to their performances using a standard competition-ranking scheme, Friedman ranking test [69], where the same rank indicates that the performance of algorithms is also the same. The rank and average rank of all the six algorithms used in this scheme are provided in Table 9, for NFE equal to 5000 and 50,000 both. The lowest (best) rank is obtained by DE.

8.3.3 DP embedded hybrid Meta-heuristics

Hybridization of all the six Meta-heuristics with DP is done in a sequential manner. DP is activated initially to contract the search space, and then the respective Meta-heuristics algorithm is applied within the reduced search space. For ease of viewing, the changed limits after applying DP are shown for 1 year through Table 10 and the corresponding reduced search space is shown in Fig. 11.

Table 9 Friedman rank of each algorithm with varying population size and function evaluation

Algorithms	Ranks						Average rank	Rank
	Function Evaluation-5000			Function Evaluation-50000				
	Population size			Population size				
	20	50	100	20	50	100		
ABC	2.90	4.00	4.30	3.90	3.60	3.60	3.88	4
DE	1.20	1.90	2.70	1.00	1.00	1.00	1.52	1
GA	4.50	3.00	2.30	5.00	3.50	3.50	3.46	3
PSO	1.90	1.10	1.00	2.00	2.00	2.00	1.62	2
SA	6.00	6.00	6.00	6.00	6.00	6.00	6	6
CS	4.50	5.00	4.70	3.10	4.90	4.90	4.52	5

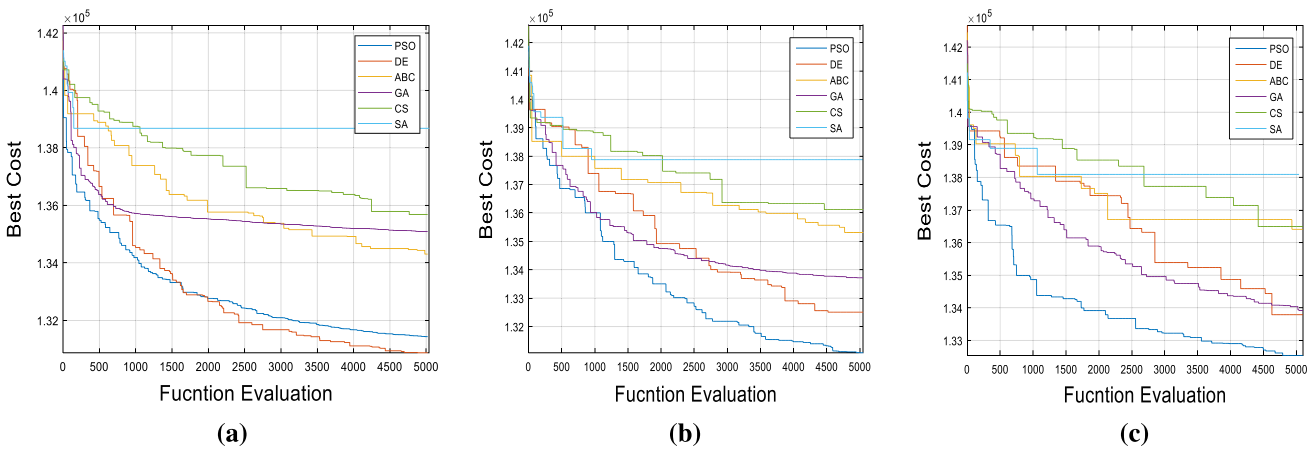


Fig. 7 Convergence of algorithms (**a**), (**b**) and (**c**) for 20, 50, 100 population size respectively in 5000NFE

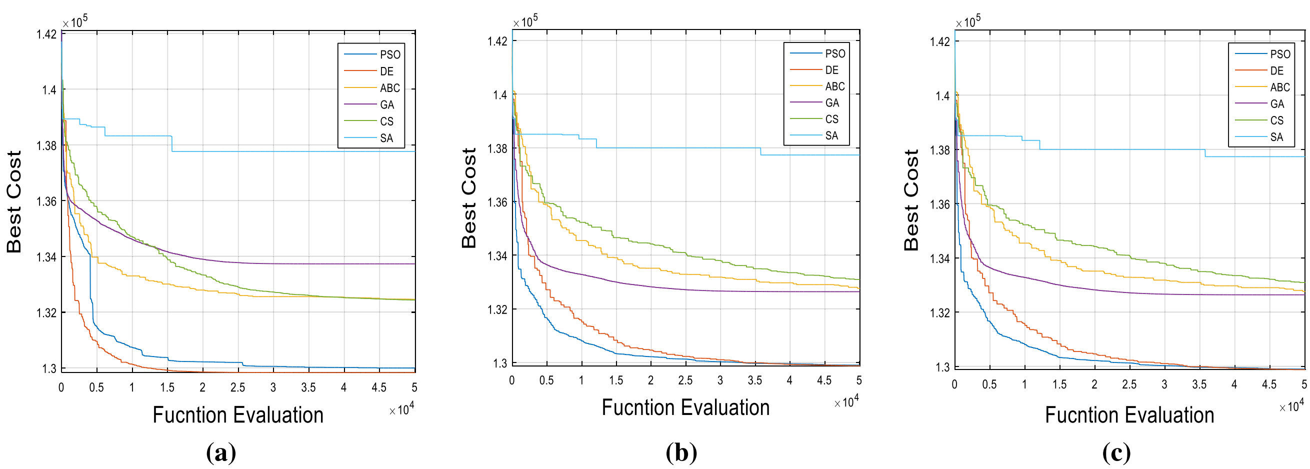


Fig. 8 Convergence of algorithms (**a**), (**b**) and (**c**) for 20, 50, 100 population size respectively in 50000NFE

Table 11 compares the performance of algorithms in terms of objective function value by providing the average, maximum and minimum values obtained by the algorithms. It also provides the standard deviation and time taken by all the algorithms. Results are evaluated in a similar fashion as that of Meta-heuristics without DP, with NFE = 5000

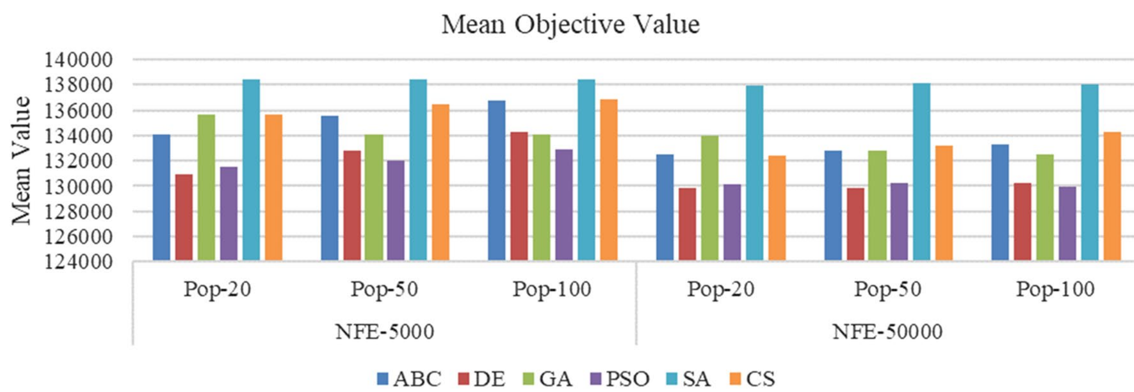


Fig. 9 Comparison of mean objective values with varying population size and function evaluation

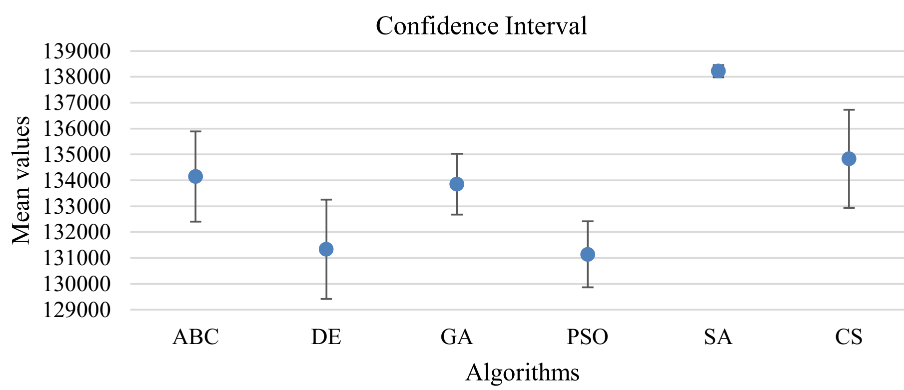


Fig. 10 Confidence interval graph of mean value

Table 10 Monthly demand before and after DP

	Months	June	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May
Before DP	Lower limit	0	0	0	0	0	0	0	0	0	0	0	0
	Upper limit	56.45	82.33	113.5	63.33	60.83	68.39	68.39	86.6	38.21	30.45	25.72	54.39
After DP	Lower limit	45.91	74.33	105.5	55.33	51.74	60.39	58.71	75.66	26.78	18.38	17.72	46.39
	Upper limit	56.45	82.33	113.5	63.33	60.83	68.39	68.39	86.6	38.21	30.45	25.72	54.39

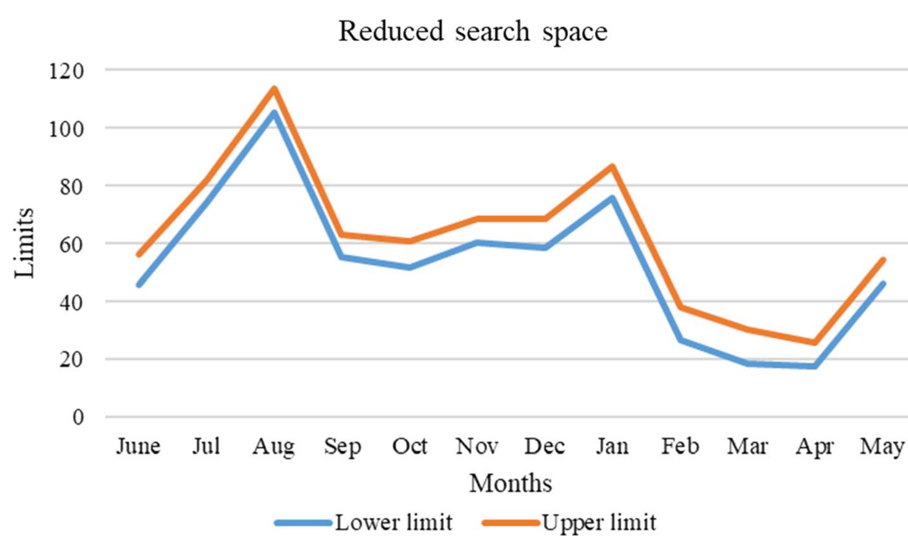


Fig. 11 Reduced search space after DP

Table 11 Comparison of proposed hybrid algorithms based on Mean, Minimum, Maximum, Standard Deviation and computational time for 30-year data

Number of function evaluation	Population size	Algorithms	Mean	Standard deviation	Minimum	Maximum	Time (s)
5000	20	DP-ABC	54,555.71	140.46	54,417.96	54,769.87	2.01
		DP-DE	47,077.40	999.19	45,330.08	48,110.59	0.94
		DP-GA	56,313.73	376.37	55,917.41	57,219.80	0.83
		DP-PSO	47,619.68	1095.95	45,539.77	49,070.30	1.11
		DP-SA	58,733.34	248.10	58,333.62	59,121.08	0.58
		DP-CS	56,557.27	264.25	56,030.94	56,848.90	0.71
	50	DP-ABC	56,270.31	236.52	55,831.63	56,567.31	2.03
		DP-DE	49,628.19	633.70	48,341.98	50,154.08	1.05
		DP-GA	54,680.36	391.57	54,128.42	55,246.36	0.82
		DP-PSO	47,754.76	887.62	46,334.69	49,025.27	1.08
		DP-SA	58,745.06	193.14	58,406.00	59,001.16	0.58
		DP-CS	57,503.07	258.66	57,186.35	58,038.22	0.75
	100	DP-ABC	57,226.88	165.19	56,971.60	57,475.71	2.04
		DP-DE	51,496.18	226.39	50,998.15	51,796.49	1.07
		DP-GA	54,714.13	230.71	54,401.27	55,116.42	0.86
		DP-PSO	49,690.79	1304.28	47,524.35	51,227.75	1.55
		DP-SA	58,822.35	132.45	58,555.56	59,034.38	0.59
		DP-CS	58,123.92	277.01	57,651.17	58,473.88	0.77
50,000	20	DP-ABC	52,735.98	14.13	52,716.59	52,756.07	19.5
		DP-DE	44,550.35	1146.49	42,859.59	45,274.87	8.1
		DP-GA	55,182.14	261.57	54,929.79	55,746.24	7.8
		DP-PSO	45,009.57	1380.94	43,472.24	47,126.43	8.5
		DP-SA	58,387.47	182.70	58,000.26	58,620.32	5.8
		DP-CS	52,713.68	19.09	52,686.75	52,740.91	7.6
	50	DP-ABC	53,131.45	37.60	53,075.64	53,180.78	19.6
		DP-DE	43,504.18	80.10	43,334.92	43,589.55	9.6
		DP-GA	53,202.02	186.80	52,972.64	53,623.17	8.1
		DP-PSO	44,227.25	827.04	43,369.44	45,962.24	8.9
		DP-SA	58,478.83	222.29	58,111.84	58,715.92	5.9
		DP-CS	53,664.19	97.78	53,517.95	53,833.81	7.8
	100	DP-ABC	53,601.69	39.60	53,518.54	53,655.15	19.9
		DP-DE	44,644.86	65.23	44,546.46	44,752.14	9.8
		DP-GA	52,757.50	32.47	52,713.35	52,807.39	8.5
		DP-PSO	44,092.28	827.79	43,347.12	46,217.36	10.6
		DP-SA	58,480.21	124.04	58,322.11	58,681.84	5.9
		DP-CS	54,970.58	194.61	54,764.82	55,359.29	7.9

and 50,000 and population size varying as 20, 50, and 100. A marked improvement was observed in the mean best value for all the algorithms. The trend in the deterioration of the mean best value on increasing the population size was observed for DP-DE and DP-PSO both for NFE = 5000. However, this was not the case when the NFE was incremented to 50,000. There was no definite relation between the mean best value and population size except for the fact that the mean value was better for NFE = 50,000, for all the population sizes in comparison to the mean best value for NFE = 5000. Standard deviation was highest for DP-PSO for all the cases.

Table 11, also shows the time taken by each algorithm to attained optimum value. If we compare it with the computational time taken by algorithms without being integrated with DE (Table 8), there is a visible improvement in most of the cases.

Figures 12a–c and 13a–c compares graphically, the performance of algorithms in terms of convergence, for all integrated algorithms used in this study. Figure 12a–c illustrates

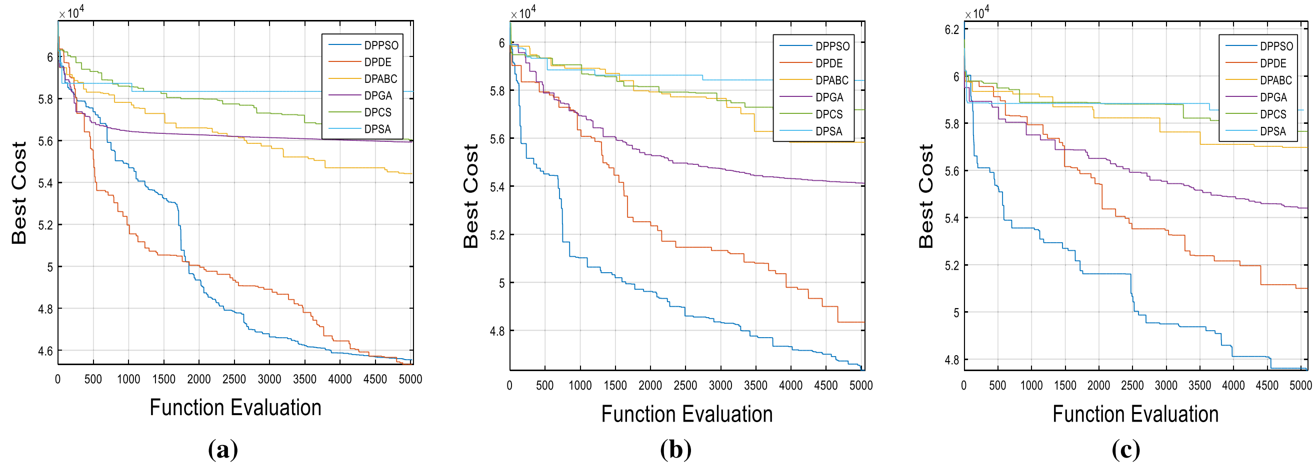


Fig. 12 Convergence of algorithms (**a**), (**b**) and (**c**) for 20, 50, 100 population size respectively in 5000NFE

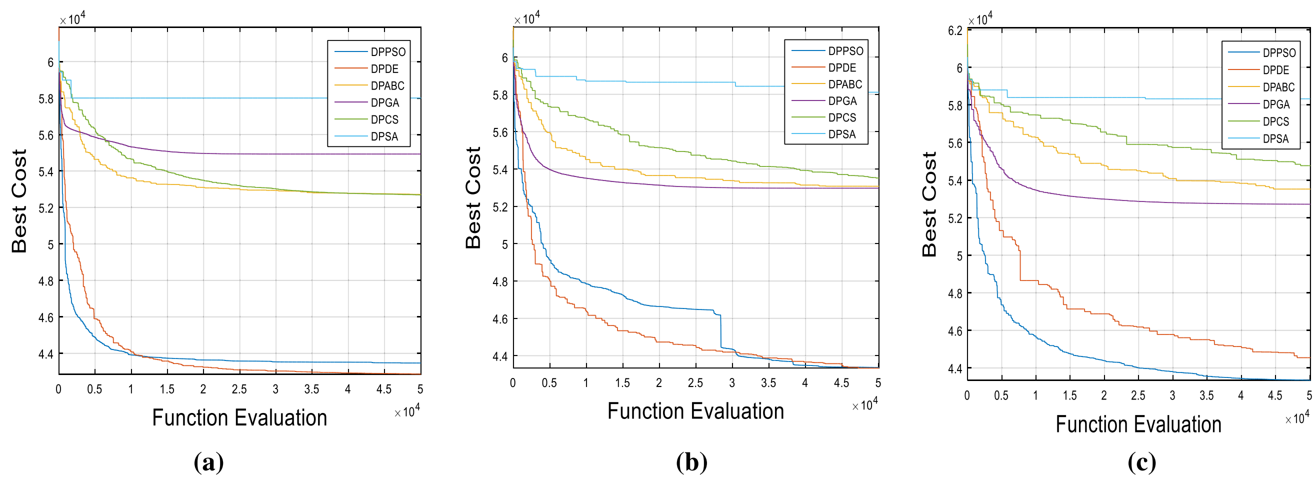


Fig. 13 Convergence of algorithms (**a**), (**b**) and (**c**) for 20, 50, 100 population size respectively in 5000NFE

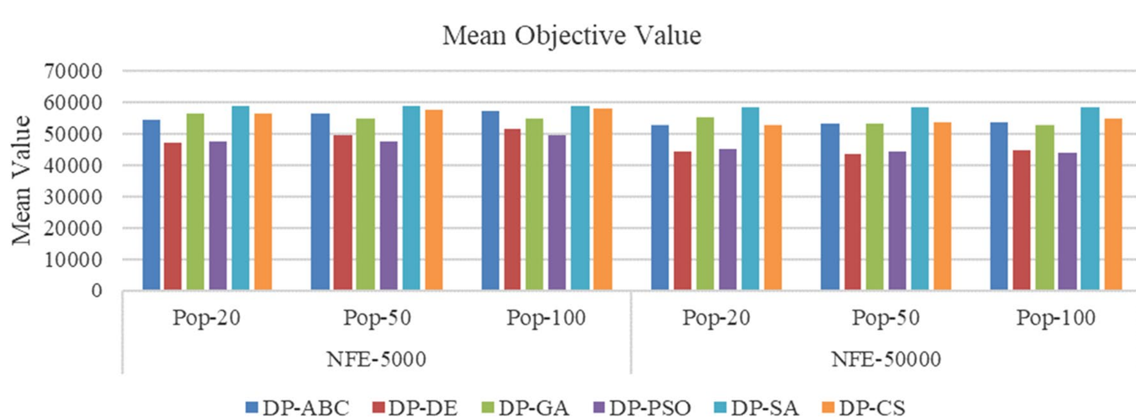


Fig. 14 Comparison of mean objective valueswith varying population size and function evaluation

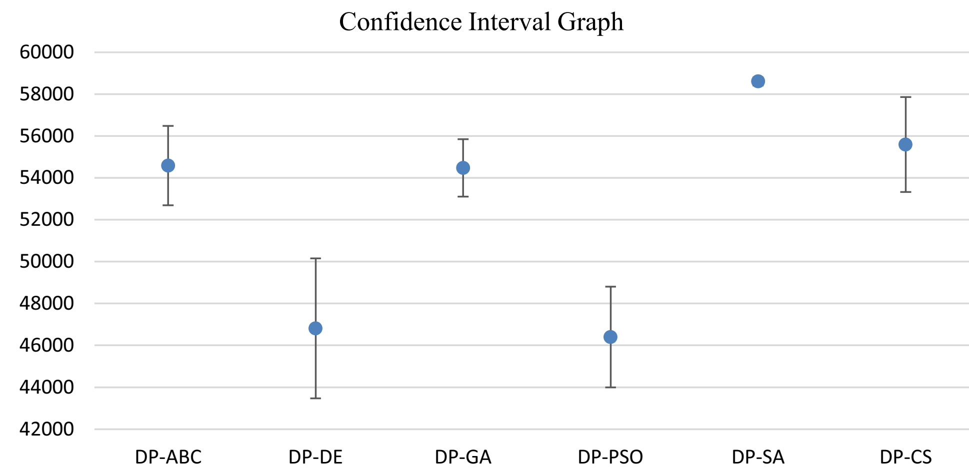


Fig. 15 Confidence interval graph of mean values for proposed algorithms

Table 12 Friedman rank of the hybrid algorithm with varying population size and function evaluation

Algorithms	Ranks						Average rank	Rank
	Function Evaluation-5000			Function Evaluation-50000				
	Population size			Population size				
	20	50	100	20	50	100		
DP-ABC	3.00	4.00	4.00	3.90	3.50	4.00	3.73	4
DP-DE	1.50	1.90	2.00	1.40	1.10	1.90	1.63	2
DP-GA	4.30	3.00	3.00	5.00	3.50	3.00	3.63	3
DP-PSO	1.50	1.10	1.00	1.60	1.90	1.10	1.36	1
DP-SA	6.00	6.00	6.00	6.00	6.00	6.00	6	6
DP-CS	4.70	5.00	5.00	3.10	5.00	5.00	4.63	5

the results after setting the number of function evaluations as 5000 and by varying the population size as 20, 50, and 100. While Fig. 13a–c shows the convergence when function evaluations are set as50000 for population size 20, 50, and 100 respectively.

These results are interpreted graphically through Fig. 14. Figure 15 depicts the confidence level for the mean objective function value with 95% confidence for the proposed algorithms.

8.4 Statistical analysis

Freidman ranking test is used to compare the algorithm on the basis of rank. Table 12 provides the ranks and the average rank for the hybrid algorithms, for NFE 5000 and 50,000 with population size varying as 20, 50, and 100. Here the results demonstrate that integration of DP with PSO outperformed all other algorithms by obtaining the smallest average rank as 1.36 and thereby securing the top position. DP-DE, the hybrid of DP with DE secured the second position by obtaining an average rank of 1.63.

Table 13 Percentage Improvement of each algorithm after introduced DP

Algorithms	Average percentage improvement (%)
DP-ABC	59.31
DP-DE	64.37
DP-GA	59.30
DP-PSO	64.62
DP-SA	57.59
DP-CS	58.78

8.5 Percentage improvement

Table 13, provides the percent improvement in the mean objective function value after integrating the Meta-heuristics with DP. For both DE and PSO, the improvement is more than 60% while for the remaining algorithms the improvement is more than 50%. The corresponding results can be interpreted graphically through Fig. 16.

Analysis for case 2: In this case, the total number of decision variables are 360. DE and PSO emerged as the best

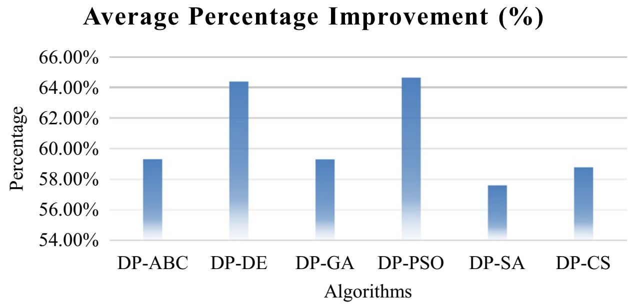


Fig. 16 Average percentage improvement in the objective value of proposed algorithms

performing algorithms without and with DP integration. This is evident through numerical, statistical, and graphical results. The effect of population size and NFE was also most visible on DE and PSO for both cases indicating that both are sensitive to these settings. Another observation, that can be made after analyzing Table 13 is that out of the six Meta-heuristics considered for this study, compatibility of DE and PSO with DP is higher in comparison to other algorithms. Wilcoxon's test applied to the mean values obtained using algorithms before and after applying DP shows the *s* statistical significance of the results with $p = 0.028$ indicating that the distribution is same for all the algorithms.

8.6 Comparison of FCADE2 and SaDE

After comparing the classical version of DE. Two adaptive variants are selected named as, FCADE2 [67] and SaDE [68] for the further comparison. In FCADE2 [67], the population is divided into a number of clusters using fuzzy C-means clustering. Adaptive mutation strategies are decided according to the distributed population points and controlling parameter in crossover is adaptive to help the algorithm in better exploration of search space during the evolution process. Qin and Suganthan [68] introduced a self-adaptive DE (SaDE), several mutation strategies recorded in a candidate pool and parameter settings are gradually self-adapted according to the learning experience during the evolution. The key thought of SaDE is the dynamic selection of strategies according to the performance of each strategy in each generation. Table 14 provides the comparison of FCADE2 and SaDE of mean values with varying population size and function evaluation for 30-year data of mula reservoir. In Table 14 it clearly shown that FCADE2 is outperform SaDE in all the combination of NFEs and population sizes (Fig. 17).

FCADE2 and SaDE are further hybridized with DP in a similar manner as discussed earlier in Sect. 4, for which the results are provided in Table 15 (Fig. 18).

8.6.1 Effect of integrating DP with Meta-heuristics

DP is a standard optimization tool, mostly applied in addition to LP techniques for reservoir operation problems. The optimum solution obtained, however, is for the discrete storage values selected. This however, is an approximate solution to the problem due to discreteness; while the true optimal solution may lie within the discrete values of storages. The underestimated or overestimated solutions obtained by DP, due to the increment used to discretize the storages proves to be helpful in narrowing down the search space for further calculations. Once the solution space is contracted, Meta-heuristics (GA, SA, PSO, DE, ABC, CS, FCADE2 and SaDE) are initiated for obtaining the optimal solution within the reduced search space.

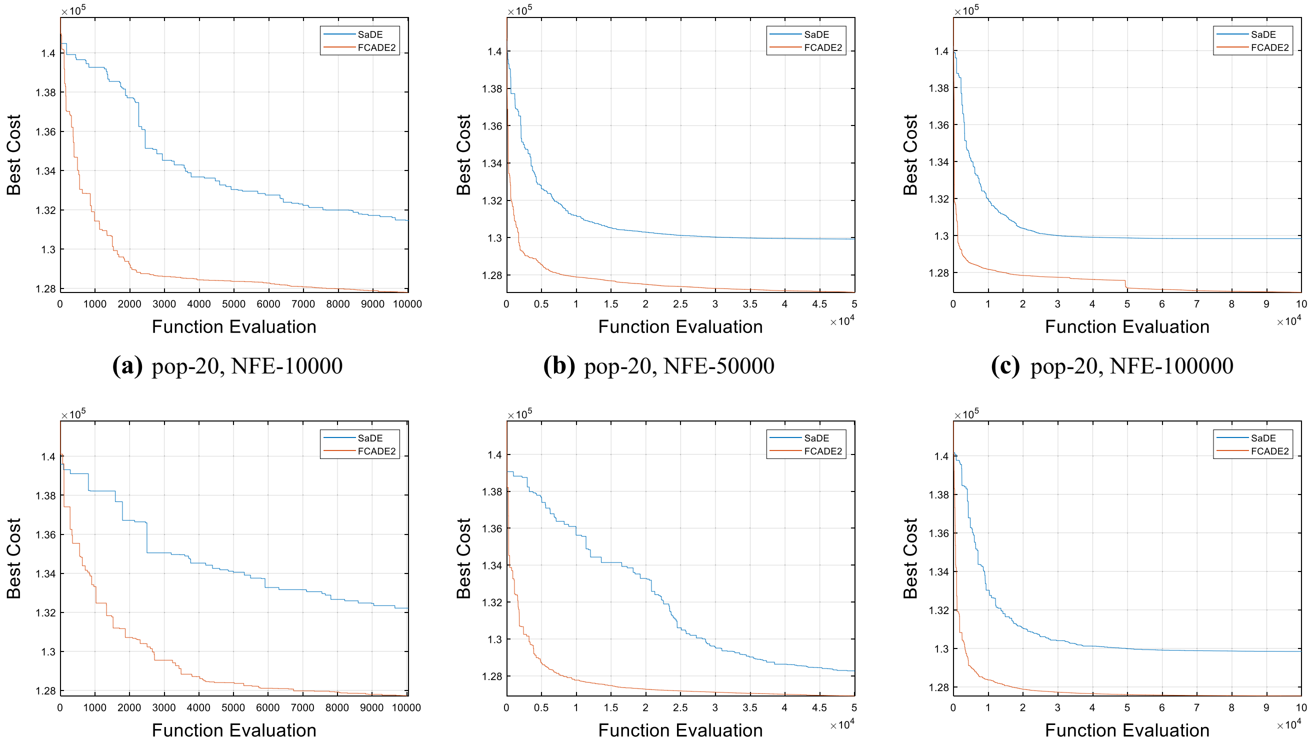
For example, let's assume that the initial search space is given as $\{Q_{\min} = 0, Q_{\max} = D_t\}$. once DP is implemented, the search space will be defined as $\{Q_{\min} = Q_t^* - \delta, Q_{\max} = Q_t^* + \delta\}$, where δ represents the increment used in DP. These new values are taken as the new range of search space for the metaheuristics used in this study. Metaheuristics in general are designed to search every region of the search space and are likely to work faster in a smaller region in comparison to a larger region.

9 Conclusion and future scope

The present article investigates the performance of Meta-heuristics for obtaining the optimal release policy for Mula reservoir, India. Two cases are considered. In the first case data set is taken for a single year and six Meta-heuristics GA, SA, PSO, DE, ABC, CS, are selected for solving it. In the second case, the dataset taken is for 30 years, and the solution is obtained through the six Meta-heuristics under

Table 14 Comparison of FCADE2, SaDE based on Mean, Minimum, Maximum, Standard Deviation and computational time for 30-year data

Algorithms	Function Evaluation-10000		Function Evaluation-50000		Function Evaluation-100000	
	Population size		Population size		Population size	
	20	50	20	50	20	50
FCADE2	127,798.24	127,713.46	127,066.9	126,916.09	126,928.51	127,533.75
SaDE	131,433.09	132,214.71	129,918.94	128,264.56	129,832.35	129,843.41



(a) pop-20, NFE-10000 **(b)** pop-20, NFE-50000 **(c)** pop-20, NFE-100000

(a) pop-50, NFE-10000 **(b)** pop-50, NFE-50000 **(c)** pop-50, NFE-100000

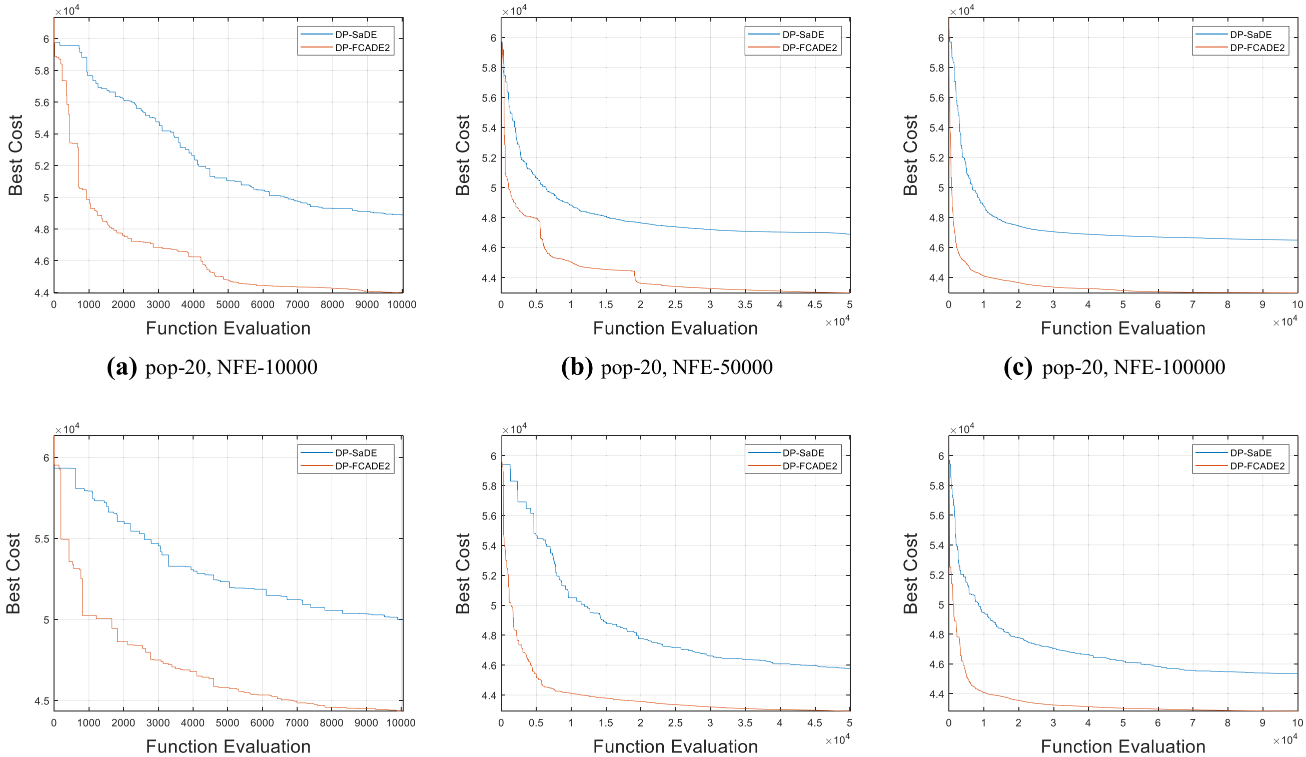
Fig. 17 Comparison of convergence graphs using FCADE2 and SaDE

Table 15 Comparison of integrated DP-FCADE2, DP-SaDE based on Mean values of objective function with varying population size and function evaluation for 30-years data

Algorithms	Function Evaluation-10000		Function Evaluation-50000		Function Evaluation-100000	
	Population size		Population size		Population size	
	20	50	20	50	20	50
DP-FCADE2	43,980.749	44,344.774	42,998.046	42,925.399	42,968.299	42,830.321
DP-SaDE	48,856.719	50,001.218	46,902.267	45,773.983	46,483.597	45,365.991

DP and non-DP environment. After comparison with classical meta-heuristics, two adaptive differential evolution variants FCADE2 and SaDE are selected for the comparison.

- As expected the performance of all the Meta-heuristics was at par with each other except for SA for the smaller data set (1 year), where the number of function evaluations are kept as 500.
- For the larger dataset of 30 years, DE gave the best performance in terms of mean function value in a non-DP environment, followed by PSO. It was also observed that the best performance is obtained for a small population of size 20 and NFE = 50,000.
- In a DP environment, the performance of all the algorithms improved significantly in terms of mean best value, with both DP-DE and DP-PSO giving an improvement of around 65% in comparison to their non-DP counterpart. This was an expected outcome because the search space was reduced to a potential region giving a better opportunity to Meta-heuristics for determining the optima. Numerically and statistically DE and PSO performed better than other algorithms. In a non-DP environment DE performed better than PSO, while in a DP environment, the performance of PSO was marginally better than DE. The above-mentioned points indicate the applicability of Meta-heuristics for larger size problems and also justifies the compatibility of Meta-heuristics with DP. It further indicates that the integration of Meta-heuristics with a classical method will enhance the performance of the latter.
- A huge deviation was observed in the maximum and minimum values of PSO in a non-DP environment and although the deviation was not as high in a DP-environment, it was still higher than other algorithms. For DE,



(a) pop-20, NFE-10000 **(b)** pop-20, NFE-50000 **(c)** pop-20, NFE-100000

(a) pop-50, NFE-10000 **(b)** pop-50, NFE-50000 **(c)** pop-50, NFE-100000

Fig. 18 Comparison of convergence graphs using DP-FCADE2 and DP-SaDE

the deviation was small in a non-DP environment, while in a DP environment it was comparatively higher. This behavior is apparently inexplicable and requires more study. Likewise, for other algorithms, no definite pattern was observed to establish a relationship between population size, NFE, and standard deviation.

- After integrated with DP, the time taken by each algorithm to reach the optimum is reduced. The average percentage improvement in time in ABC 5%, in DE 16%, in GA 20%, in PSO 15%, in SA 7% and in CS 10%.
- Two adaptive variants are selected named as, FCADE2 and SaDE for the further comparison. In which FCADE2 is outperform SaDE in all the combination of NFE and population size

Considering the pros and cons of the empirical results, at this point of time, it can be safely concluded that integration of classical method with Meta-heuristics with definitely yield good results, and secondly, smaller population size for sufficiently large NFE will give better results.

9.1 Future directions

- In this paper, DP was used as a local search method for contracting the search space to a potential region. It will be interesting to have a comparison of other classical methods and Meta-heuristics which remains a future part of the study. Secondly, uncertainty in parameters (inflow and outflow) is a common phenomenon in a real-life scenario. This study can be continued further by considering different scenarios of uncertainty. Also, from the mathematical point of view, the problem can be modeled as a multi-objective optimization problem and can technically be extended for cases where the number of reservoirs are more than one.

Appendix A

Inflow data

June	63.51	62.48	67.51	62.86	149.45	78.77	53.77	54.32	158.99	41.68	28.36	75.37	113.28	30.38	42.76	0
July	414.69	177.69	77.05	178.47	270.59	312.32	177.11	176.83	217.21	313.78	104.63	166.28	215.94	144.7	183.48	101.68
August	196.9	148.14	159.08	178.05	346.66	149.51	253.46	420.67	167.7	258.57	359.37	253.41	246.87	194.42	201.88	167.42
September	67.76	1185.58	100.24	151.23	352.19	122.87	110.67	391.38	334.7	217.26	55.65	203.7	23.54	39.74	33.38	50.41
October	2.51	32.71	98.35	229.69	56.13	0	20.49	88.53	37.88	0	19.87	67.82	77.1	21.17	4.44	41.49
November	6.95	8.14	35.82	18.75	23.83	41.12	18.48	28.23	16.04	0	27.52	12.77	4.86	1.39	0.45	8.08
December	0	4.81	49.46	0	12.01	9.53	16.62	23.56	0	0	2	7.04	7.78	0.23	3.79	0
January	0	3.59	37.39	6.56	15.68	11.05	8.12	10.14	42.97	1.71	2.04	6.58	0	3.65	2	0
February	2.75	4.91	33.19	6.86	13.3	12.52	2.09	0	0	21.5	0	8.55	0	23.01	0.96	1.65
March	6.62	15.28	53.52	6.81	14.91	5.13	0.65	16.05	9.37	0	0	4.84	7.12	5.02	0.15	3
April	4.81	8.14	56.6	20.53	23.63	15.74	3.83	6.12	7.08	6.31	3.31	0.42	0	1.99	3.79	6.69
May	0	12.42	44.99	8.72	0	8.45	3.04	7.79	16.44	0	0.21	3.97	7.59	7.34	5.17	5.58

June	81.59	19.82	66.31	244.51	11.75	24.04	79.88	8.04	5.69	69.08	26.01	80.27	60.78	74.45
July	271.59	150.1	296.31	516.06	95.92	280.47	628.05	164.33	295.52	227.97	183.52	234.15	206.55	287.28
August	165.1	210.18	351.67	372.92	185.91	163	313.77	85.31	267.95	398.63	155.96	186.13	128.16	168.84
September	210.73	240.15	97.5	64.64	87.58	80.55	172.99	94.89	103.14	62.51	214.44	102.39	88.31	44.24
October	81.64	65	250.06	40.82	4.65	97.55	5.05	20.76	182.76	336.45	214.58	133.24	19.15	115.7
November	7.11	8.08	31.19	13.98	0	15.06	23.65	3.73	31.68	28.69	59.25	6.81	8.38	9.42
December	4.47	6.49	6.34	9.74	2.08	18.04	2.61	8.45	5.46	30.96	7.9	6.81	8.48	8.7
January	0.3	10.1	18.63	9.43	0	6.35	12.65	4.09	4.33	7.41	8.5	5.39	90.8	7.63
February	3.85	15.26	11.94	2.75	0.33	4.3	5.86	1.03	5.35	66.12	12.27	5.53	3.13	6.59
March	4.12	4.82	38.52	12.25	0	5.62	4	5.53	5.85	6.79	9.11	8.98	4.2	18.51
April	5.72	4.33	14.29	9.56	2.95	3.64	4.36	1.51	5.7	6.17	9.14	8.33	7.1	5.33
May	0.11	22.29	15.37	6.47	3.81	3.96	4.69	4.84	288	9.67	12.06	7.97	2	6.43

Appendix B

List of algorithms and their Pseudocode.

A. **Genetic Algorithm (GA)**

Genetic Algorithms (GA) was developed by John Holland and his colleagues in 1973. It is inspired by the natural biological evolution process and the survival of the fittest policy and is used for solving complex optimization problems such as constrained and unconstrained problems. It is a robust technique that helps us to achieve a near-optimal solution for various complex problems.

GA Algorithm:

1. Initialize the population of genes using encoded chromosomes.
2. Calculate the solution for each and every member gene of the population.
3. Store the best solution of the entire population.
4. For each generation of population repeat the following steps until the population size is reached
5. Select parents using any of the selection process corresponding to the problem statement
6. Apply crossover to the selected parent's chromosomes. Crossover process used in this paper is the uniform crossover

   $$\textit{Uniform Crossover}: \; y_i = \alpha * x_i + (1 - \alpha) * x_j$$

   where y_i: offspring; x_i and x_j: parent chromosome; α: Random number.
7. Apply mutation to the children chromosomes obtained in the above step.
8. Calculate the solution for the new chromosomes created.
9. Replace parents with children in the population.
10. Store the best solution of this generation.
11. Stop when maximum iteration is reached or stopping condition is met.

B. Simulated Annealing (SA)

Simulated Annealing was proposed by Kirkpatrick in 1983. SA is a stochastic meta-heuristic algorithm and is one of the first and oldest techniques that do not get trapped in local optima. SA is inspired by the field of metallurgy and the process on which it is based is annealing.

SA Algorithm:

1. Generate random initial solution for the problem.
2. Calculate the fitness for initialized solution F_{i}.
3. Generate a new solution for the problem and calculate fitness for the new solution F_j.
4. Accept the new solution if F_j is strictly superior than F_i.
5. The new solution is accepted even if F_j is not strictly superior than F_i when comparison between 'acceptance probability' and random number gives a positive result. Acceptance probability for new solution is calculated as:

   $$P = e^{-\frac{\Delta E}{T}}$$

   where ΔE: Energy difference; T: Temperature; e: 2.71828.
6. Update temperature using the following formula

   $$T = T - \Delta T$$

   where ΔT: Temperature change.
7. Repeat steps 3–6 until maximum iterations are reaches or stopping criteria is met.

C. Particle Swarm Optimization (PSO)

PSO was researched and stated by Eberhart and Kennedy in 1995, and is based on the intelligent socio cooperative foraging behavior displayed by lower level organisms like fish school and bird convoys.

PSO Algorithm:

1. Define fitness function along with the search space range (lower bound and upper bound) and parameter length for every particle considered in the search space.
2. Declare population size for the particles and set inertia and acceleration factors.
3. Initialize all the parameters of every particle considered in the search space (also called as population). Then calculate best solution for every particle and the respective position where we get the best solution (denoted as personal best). Also calculate global best particle position for the entire population.
4. Update population in every iteration by:
   - Update particle velocity vector using the formula

     $$w * v_i(t) + c1 * rand() * \left(persBest - v_i(t+1) = x_i(t)\right) + c2 * rand() * (globalBest - (t))$$

     where $x_i(t)$: position of particle; $v_i(t)$: velocity of particle; w: inertia weight; $c1$ and $c2$: accelerating factors; rand (): random number generator function with input range [0,1].
   - Update particle position vector using the formula:

$x_i(t+1) = x_i(t) + v_i(t+1)$

- Update best particle position vector.
- Repeat the above steps for every particle of the population.
- Update global best position in that population generation.

5. Repeat until maximum iteration or stopping criteria is reached.

D. Differential Evolution (DE)

DE is an evolutionary optimization technique developed by Storn and Price in 1995. DE is Population-based search technique that helps to achieve the robust optimum solution for a given problem faster than other evolutionary problems. DE uses same operators i.e. mutation, crossover and selection as that of GA but the manner in which these operators are applied makes DE differ from GA.

DE Algorithm:

1. Initialize population with NP parameter vectors, making population size for each generation as NP.
2. Repeat the following step while maximum iteration is reached or stopping criteria is met.
3. For each individual in the population do
   - Choose three different individuals other the one being processed randomly.

     $X_{r1} \neq X_{r2} \neq X_{r3} \neq X_j$

   - Apply mutation by adding the weighted difference of the two vectors to the third vector.

     $V = X_{r1} + F * \left(X_{r2} - X_{r3}\right)$

     where F: scaling factor.
   - Apply Crossover by mixing mutated vector 'V' with target vector 'X'.

     $$\boldsymbol{U} = \begin{cases} X \ if \ rand_j \leq Cr \\ X \ otherwise \end{cases}$$

     where Cr: crossover rate.
   - Selection is performed between the trial vector 'U' and target vector 'X' depending upon which is strictly superior.

     $$\boldsymbol{X}(\boldsymbol{t}+1) = \begin{cases} U \ if \quad f(U) \leq f(X) \\ X \ otherwise \end{cases}$$

4. Update the population
5. Repeat until maximum iteration or stopping criteria is reached.

E. Artificial Bee Colony (ABC)

ABC proposed by Karaboga in 2007 is based on two necessary and sufficient properties for swarm intelligence, self-organization, and division of labor.ABC simulates the behavior of real bees for solving multi-dimensional and multimodal optimization problems.

ABC Algorithm

1. Initialize the bee population with food source position information.
2. Repeat the following steps until the maximum iteration is reached or stopping condition is met.
3. Employed bees are sent to the food source and nectar amount collected by the bees is calculated.
4. Onlooker bees select the food source position to collect nectar by using the following equation

   $$p_i = \frac{fit_i}{\sum_{n=1}^{SN} fit_n}$$

   where fit_i: fitness of the solution of the ith bee food source.
5. New food source position is calculated by both employed and onlooker bees by using the following equation

   $$V = X_{ij} + \varphi * \left(X_{ij} - X_{kj}\right)$$

   where V: neighborhood new food source position; X_i: ith solution position in the swarm; φ: Random number.
6. Food sources are abandoned by the employed and onlooker bees based value of the solution calculated from a new food source position. Further exploitation of these abandoned food source is stopped by the bees.
7. Scouts are sent to search for new food sources at random source position based on the abandoning limit factor 'L'.
8. Best food source found so far based on the amount of nectar collected is memorized.

F. Cuckoo search (CS)

CS was proposed by Yang and Deb in 2009. It is inspired by the parasitic behavior shown by cuckoo bird when the female cuckoo lays eggs in the nests of host birds of other species.

CS Algorithm

1. Initialize population of 'n' host nest where 'n' is the population size.
2. Repeat the following steps until maximum iterations are reached or stopping criteria is met
3. Get a cuckoo randomly using levy flight strategy and evaluate its fitness F_i.
4. Generate new solution by choosing a host nest position and calculate its fitness F_j using the formula

   $x(t+1) = x(t) + \alpha \oplus levy(\lambda)$

   where $\alpha > 0$ is the step size; $\oplus$ is the entry wise multiplication operator.
5. Replace the solution j for host nest position generated in the above step with randomly chosen cuckoo if $F_i > F_j$.
6. Abandon p_a fraction of host nests giving bad solutions and generate new host nests.
7. Store the solution of the host nest of the best quality.

References

1. Bellman R (1952) On the theory of dynamic programming. Proc Natl Acad Sci USA 38(8):716
2. Murty KG (1983) Linear programming. Wiley, Chichester
3. Bellman R, Dreyfus S (1962) Applied dynamic programming. Princeton University Press, Princeton
4. Larson RE (1968) A dynamic programming successive approximations technique. In: Joint automatic control conference, Univsity of Michigan
5. Larson RE (1968) State increment dynamic programming. Elsevier, New York
6. Hall WA et al (1969) Optimum firm power output from a two reservoir system by incremental dynamic programming. Water Research Centre, University of California, LA., Contribution 130.
7. Heidari M et al (1971) Discrete differential dynamic programing approach to water resources systems optimization. Water Resour Res 7(2):273–282. https://doi.org/10.1029/WR007i002p00273
8. Fults DM, Hancock LF (1972) Optimum operations model for Shasta-Trinity system. J Hydraul Div ASCE 98(HY9):1497–1514
9. Trott WJ, Yeh WW-G (1973) Optimization of multiple reservoir system. J Hydraul Div ASCE 99(10):1865–1884
10. Chow VT, Maidment DR, Tauxe GW (1975) Computer time and memory requirements for DP and DDDP in water-resource systems-analysis. Water Resour Res 11(5):621–628. https://doi.org/10.1029/WR011i005p00621
11. Fults DM, Hancock LF, Logan GR (1976) A practical monthly optimum operations model. J Water Resources Plan Manag Div ASCE 102(1):63–76
12. Giles JE, Wunderlich WO (1981) Weekly multipurpose planning model for TVA reservoir system. J Water Resources Plan Manag Div ASCE 107(2):495–511
13. Labadie JW (1999) Generalized dynamic programming package: CSUDP version 3.3 Documentation and User Guide. Department of Civil Engineering, Colorado State University, Ft. Collins
14. Shim K-C, Fontane DG, Labadie JW (2002) Spatial decision support system for integrated river basin flood control. J Water Resources Plan Manag ASCE 128(3):190–201. https://doi.org/10.1061/(ASCE)0733-9496(2002)128:3(190)
15. Yi J, Labadie JW, Stitt S (2003) Dynamic optimal unit commitment and loading in hydropower systems. J Water Resources Plan Manag ASCE 129(5):388–398. https://doi.org/10.1061/(ASCE)0733-9496(2003)129:5(388)
16. Kumar DN, Baliarsingh F (2003) Folded dynamic programming for optimal operation of multi-reservoir system. Water Resour Manag 17:337–353. https://doi.org/10.1023/A:1025894500491
17. Labadie JW (2004) Optimal operation of multi-reservoir systems: state-of-the-art review. J Water Resources Plan Manag ASCE 130(2):93–111
18. Yurtal R, Seckin G, Ardiclioglu M (2005) Hydropower optimization for the Lower Seyhan system in Turkey using dynamic programming. Water Int 30(4):522–529. https://doi.org/10.1080/02508060508691896
19. Castelletti A et al (2007) Neuro-dynamic programming for designing water reservoir network management policies. Control Eng Pract 15(8):1031–1038. https://doi.org/10.1016/j.conengprac.2006.02.011
20. Tilmant A, Kelman R (2007) A stochastic approach to analyze trade-offs and risks associated with large-scale water resources systems. Water Resour Res 43(6):W06425. https://doi.org/10.1029/2006WR005094
21. Nandalal KDW, Bogardi JJ (2007) Dynamic programming based operation of reservoirs: applicability and limits. Cambridge University Press, Cambridge
22. Srivastava DK, Awchi TA (2009) Storage-yield evaluation and operation of mula reservoir, India. J Water Resources Plan Manag 135(6):414–425. https://doi.org/10.1061/(ASCE)07339496(2009)135:6(414)
23. Rani D, Moreira M (2010) Simulation-optimization modelling: A survey and potential application in reservoir systems operation. Water Resour Manage 24(6):1107–1138. https://doi.org/10.1007/s11269-009-9488-0
24. Li X, Wei J, Li T, Wang G, Yeh WW-G (2014) A parallel dynamic programming algorithm for multi-reservoir system optimization. Adv Water Resour 67:1–15
25. He S, Guo S, Chen K, Deng L, Yin J (2019) Optimal impoundment operation for cascade reservoirs coupling parallel dynamic programming with importance sampling and successive approximation. Adv Water Resources 131, Article 103375
26. Zeng X, Hu T, Cai X, Zhou Y, Wang X (2019) Improved dynamic programming for parallel reservoir system operation optimization. Adv Water Resources 131, Article 103373
27. Teegavarapu RSV, Simonovic SP (2002) Optimal Operation of Reservoir Systems using Simulated Annealing. Water Resources Manag 16:401–428. https://doi.org/10.1023/A:1021993222371
28. Georgiou PE, Papamichail DM, Vougioukas SG (2006) Optimal irrigation reservoir operation and simultaneous multi-crop cultivation area selection using simulated annealing. Irrigat Drain J Int Commission Irrigat Drain 55(2):129–144
29. Jothiprakash V, Shanthi G (2006) Single reservoir operating policies using genetic algorithm. Water Resour Manag 20(6):917–929
30. Kumar DN, SrinivasaRaju K, Ashok B (2006) "Optimal reservoir operation for irrigation of multiple crops using genetic algorithms. J Irrigat Drain Eng 132(2):123–129
31. Lakshminarasimman L, Subramanian S (2006) Short-term scheduling of hydrothermal power system with cascaded reservoirs by using modified differential evolution. IEE Proc Gen Transm Distrib 153(6):693–700
32. Kerachian R, Karamouz M (2006) Optimal reservoir operation considering the water quality issues: A stochastic conflict resolution approach. Water Resour Res 42:W12401

33. Kerachian R, Karamouz M (2007) A stochastic conflict resolution model for water quality management in reservoir-river system. Adv Water Resour 30(4):866–882
34. Reddy MJ, Kumar DN (2007) Multiobjective differential evolution with application to reservoir system optimization. J Comput Civil Eng 21(2):136–146
35. Chiu Y-C, Chang L-C, Chang F-J (2007) Using a hybrid genetic algorithm–simulated annealing algorithm for fuzzy programming of reservoir operation. Hydrol Process Int J 21(23):3162–3172
36. Reddy MJ, Kumar DN (2008) Evolving strategies for crop planning and operation of irrigation reservoir system using multi-objective differential evolution. Irrigat Sci 26(2):177–190
37. Baltar AM, Fontane DG (2008) Use of multiobjective particle swarm optimization in water resources management. J Water Resources Plan Manag 134(3):257–265
38. Mandal KK, Chakraborty N (2009) Short-term combined economic emission scheduling of hydrothermal power systems with cascaded reservoirs using differential evolution. Energy Convers Manag 50(1):97–104
39. Khodabakhshi F, Ghirian AR, Khakzad N (2009) Applying simulated annealing for optimal operation of multi-reservoir systems. Am J Eng Appl Sci 1:80–87
40. Malekmohammadi B, Zahraie B, Kerachian R (2010) A real-time operation optimization model for flood management in river-reservoir systems. Nat Hazards 53(3):459–482
41. Qin H et al (2010) Multi-objective cultured differential evolution for generating optimal trade-offs in reservoir flood control operation. Water Resources Manag 24(11):2611–2632
42. Regulwar DG, Choudhari SA, Raj PA (2010) Differential evolution algorithm with application to optimal operation of multipurpose reservoir. J Water Resource Protect 2(06):560
43. Hınçal O, Altan-Sakarya AB, Ger AM (2011) Optimization of multireservoir systems by Genetic Algorithm. Water Resour Manag 25:1465–1487
44. Adeyemo J, Otieno F (2010) Differential evolution algorithm for solving multi-objective crop planning model. Agric Water Manag 97(6):848–856
45. Fallah-Mehdipour E, Bozorg Haddad O, Marino MA (2011) MOPSO algorithm and its application in multipurpose multireservoir operations. J Hydroinf 13(4):794–811
46. Kangrang A, Compliew S, Hormwichian R (2011) Optimal reservoir rule curves using simulated annealing. Proc Inst Civil Eng Water Manag 164(1):27–34
47. Li FF, Wei JH, Fu XD, Wan XY (2012) An effective approach to long-term optimal operation of large-scale reservoir systems: case study of the three gorges system. Water Resources Manag 26(14):4073–4090
48. Afshar MH (2012) Large scale reservoir operation by constrained particle swarm optimization algorithms. J Hydro-Environ Res 6(1):75–87
49. Afshar A, Shojaei N, Sagharjooghifarahani M (2013) Multiobjective calibration of reservoir water quality modeling using multi-objective particle swarm optimization (MOPSO). J Water Resour Manag 27(7):1931–1947
50. Zhang Z et al (2014) An adaptive particle swarm optimization algorithm for reservoir operation optimization. Appl Soft Comput 18:167–177
51. Mei CS, El-Shafie A (2014) Artificial Bee Colony (Abc) approach for reservoir operation
52. Rani D, Srivastava DK (2016) Optimal operation of Mula reservoir with combined use of dynamic programming and genetic algorithm. Sustain Water Resources Manag 2(1):1–12
53. Rani D, Srivastava DK, Gulati TR (2016) A set of linked optimization models for an inter-basin water transfer. Hydrol Sci J 61(2):371–392
54. SaberChenari K, Abghari H, Tabari H (2016) Application of PSO algorithm in short-term optimization of reservoir operation. Environ Monit Assess 188(12):667
55. Choong S-M, El-Shafie A (2016) Wan Mohtar WHM (2016) An application of artificial bee colony algorithm for reservoir optimization: a case study of Chenderoh Dam, Malaysia. Int J Adv Agric Environ Eng (IJAAEE) 3(2):5
56. Ahmad A et al (2016) The application of artificial bee Colony and gravitational search algorithm in reservoir optimization. Water Resour Manag 30(7):2497–2516
57. Yasar M (2016) Optimization of reservoir operation using cuckoo search algorithm: Example of adiguzel dam, Denizli, Turkey. Math Probl Eng 2016:1–7
58. Pérez CJ et al (2017) A Multi-Objective Artificial Bee Colony-based optimization approach to design water quality monitoring networks in river basins. J Cleaner Prod 166:579–589
59. Nieto PJG et al (2017) A hybrid wavelet kernel SVM-based method using artificial bee colony algorithm for predicting the cyanotoxin content from experimental cyanobacteria concentrations in the Trasona reservoir (Northern Spain). J Comput Appl Math 309:587–602
60. Niu W et al (2018) A parallel multi-objective particle swarm optimization for cascade hydropower reservoir operation in southwest China. Appl Soft Comput 70:562–575
61. Holland JH (1973) Genetic algorithms and the optimal allocation of trials. SIAM J Comput 2(2):88–105
62. Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220(4598):671–680
63. Eberhart R, Kennedy J (1995) A new optimizer using particle swarm theory. In: Proceedings of the sixth international symposium on micro machine and human science, 1995.MHS'95. IEEE, pp 39–43
64. Storn R (1996) On the usage of differential evolution for function optimization. In: Proceedings of North American Fuzzy Information Processing. IEEE, Berkeley, pp 519–523
65. Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Glob Optim 39(3):459–471
66. Yang XS, Deb S (2009) Cuckoo search via Lévy flights. In: World congress on nature & biologically inspired computing, 2009. NaBIC 2009. IEEE, pp 210–214
67. Bilal MP, Vig G (2020) Clustering based adaptive differential evolution for numerical optimization. In: 2020 IEEE congress on evolutionary computation (CEC), Glasgow, United Kingdom, pp 1–8. doi: https://doi.org/10.1109/CEC48606.2020.9185874.
68. Qin AK, Suganthan PN (2005) Self-adaptive differential evolution algorithm for numerical optimization. In: 2005 IEEE congress on evolutionary computation, Edinburgh, Scotland, vol 2, pp 1785–1791. doi: https://doi.org/10.1109/CEC.2005.1554904.
69. Bagdonavičius V, Julius K, Nikulin MS (2013) Non-parametric tests for complete data. Wiley, New York