Water Resour Manage (2009) 23:2119–2139
DOI 10.1007/s11269-008-9373-2

An Improved Continuous Ant Algorithm for Optimization of Water Resources Problems

S. Madadgar · A. Afshar

Received: 9 November 2007 / Accepted: 14 October 2008 /
Published online: 18 November 2008

Abstract Ant colony optimization was initially proposed for discrete search spaces while in continuous domains, discretization of the search space has been widely practiced. Attempts for direct extension of ant algorithms to continuous decision spaces are rapidly growing. This paper briefly reviews the central idea and mathematical representation of a recently proposed algorithm for continuous domains followed by further improvements in order to make the algorithm adaptive and more efficient in locating near optimal solutions. Performance of the proposed improved algorithm has been tested on few well-known benchmark problems as well as a real-world water resource optimization problem. The comparison of the results obtained by the present method with those of other ant-based algorithms emphasizes the robustness of the proposed algorithm in searching the continuous space more efficiently as locating the closest, among other ant methods, to the global optimal solution.

Keywords ACO · Continuous ant algorithm · Reservoir operation optimization · Explorer ants · Adaptation operator

1 Introduction

Most of meta-heuristic methods benefit from a population of intelligent swarms that search the decision space biasing toward an optimal solution. Of the most popular

S. Madadgar (✉)
Hydroinformatic Research Center, Department of Civil Engineering,
Iran University of Science and Technology, Tehran, Iran
e-mail: sh_madadgar@civileng.iust.ac.ir

A. Afshar
Center of Excellence for Fundamental Studies in Structural Mechanics,
Iran University of Science and Technology, Tehran, Iran
e-mail: a_afshar@iust.ac.ir

meta-heuristic methods, Genetic Algorithms (GAs) have been applied to some reservoir operation optimization problems (Esat and Hall 1994; Fahmy et al. 1994; Oliveira and Loucks 1997; Wardlaw and Sharif 1999). Labadie (2004) presented a state-of-the-art review over the application of mathematical and heuristic optimization algorithms such as genetic algorithms, artificial neural networks, and fuzzy-based approach to the multi-reservoir operation optimization problem. Bozorg Haddad and Afshar (2004) applied Honey Bees Mating Optimization (HBMO) meta-heuristic algorithm to a single reservoir operation problem with 60 operating periods.

Inspiration from ant colonies' foraging behavior led to a group of swarm-based optimization algorithms named Ant Colony Optimization (ACO) that was first introduced in the early 1990s to solve Combinatorial Optimization Problems (COPs) such as the traveling salesman and the quadratic assignment problems (Dorigo 1992; Dorigo et al. 1996). First application of ACO algorithms to water engineering problems was reported by Abbaspour et al. (2001). They employed ACO algorithms in estimating hydraulic parameters of unsaturated soils. Simpson et al. (2001) studied the selection of tuning parameters in ant algorithms for optimizing pipe network systems. Maier et al. (2003) applied ACO algorithms to find optimal design of water distribution systems. They concluded that ACO algorithms may form a competitive alternative to genetic algorithms. Zecchin et al. (2003) used the original ant system (Dorigo et al. 1996) for optimization of water distribution networks and compared their performance with the min–max ant system, a modified version of the ant system proposed by Stützle and Hoos (1997a, b). Jalali et al. (2005) proposed an improved version of the ACO algorithm in single water-reservoir operation optimization. They employed explorer ants as a local search technique in a standard ACO algorithm. Nagesh Kumar and Janga Reddy (2006) compared the performance of ACO algorithm with real coded *GA* to derive operating policies for a multi-purpose reservoir system. They emphasized superior performance of ACO, especially in long-time horizon operation models.

Ant colony optimization algorithms were originally proposed for discrete search spaces. In continuous domains, discretization of the search space has been successfully implemented. Jalali et al. (2007) proposed a multi-colony ant algorithm to discretize the continuous search space non-homogenously in order to focus on the area surrounding the optimum solution.

Early attempts for direct extension of ant algorithms to continuous decision space led to the Continuous ACO (CACO) algorithm which was initially proposed by Bilchev and Parmee (1995). Later, other ant-based methods for continuous domains like Asynchronous Parallel Implementation (API) and Continuous Interacting Ant Colony (CIAC) were proposed (Monmarche et al. 2000; Dreo and Siarry 2002). Conceptually, however, they are far from the original spirit of ant colony optimization. Mostly benefiting from the original concepts of ACO, Socha and Dorigo (2006) proposed a new algorithm called ACO_R. They tested the algorithm with standard benchmark problems and emphasized the better performance of ACO_R in comparison with other ant-related algorithms in continuous domains.

Benefiting from the general concepts of ACO_R, this paper presents an improved and adaptive ant colony algorithm which integrates both adaptation operator and explorer ants into the structure of the original ACO_R for optimization of water resources problems in continuous domains. Performance of the proposed model is discussed on some well-known benchmark problems. As real-world water resource problem, the optimization of a nonlinear and non-convex hydropower reservoir operation is considered. It is indicated that for the tested benchmark problems as well as the present case study, the proposed algorithmoutperforms the original ACO_Rin locating a good near optimum solutions.

2 Ant Colony Optimization Algorithms: Spirit and Original Concepts

Ant Colony Optimization (ACO) algorithms are inspired by real-ants foraging behavior. Ants start searching for food by exploring the area around their nest in a completely random manner. During the return trip from the food source to the nest, ants deposit some chemical liquid called pheromone on the ground. The amount of pheromone deposited by an ant may depend on the quantity and quality of the food. The amount of pheromone on the ground encourages, but not obligates, other ants to follow the trail. This behavior of real ant colonies has inspired the definition of artificial ant colonies to find approximate solutions of hard combinatorial optimization problems.

The central component of any ACO algorithm is the pheromone model, which is to probabilistically sample the search space. At each construction step, the solution components are selected through a probability rule which vary across different variants of ACO. The probability rule in the original Ant System (AS) is defined as follows (Dorigo et al. 1996):

$$P\left(c_{ij}\middle| s^p, t\right) = \frac{\left[\tau_{ij}(t)\right]^\alpha \times \left[\eta\left(c_{ij}\right)\right]^\beta}{\sum_{j=1}^{J} \left[\tau_{ij}(t)\right]^\alpha \times \left[\eta\left(c_{ij}\right)\right]^\beta}, \quad \forall c_{ij} \in \text{allowable set} \tag{1}$$

in which $p\left(c_{ij}\,|s^p, t\right)$ is the probability of adding the solution component to the current partial solution s^p at iteration t; $\tau_{ij}(t)$ is the pheromone value associated with component at iteration t; $\eta(\cdot)$ is a weighting function which assigns the heuristic value to represent the cost of choosing the solution component c_{ij}; α and β are two parameters to control the relative importance of the pheromone trail and heuristic value; and i is the current construction step including J component solutions in the allowable set. The heuristic value η is the same as providing the ants with sight and is sometimes called visibility.

The pheromone deposit is made once all ants have constructed their solutions. The pheromone is updated to increase the pheromone concentrations associated with good or promising solutions, and decrease those that are associated with less desirable ones. This is usually achieved by increasing the pheromone levels associated with chosen good solution, s_{ch} by a certain value, $\Delta\tau$, and decreasing all the pheromone values through pheromone evaporation:

$$\tau_{ij}(t+1) = \begin{cases} (1-\rho)\,\tau_{ij}(t) + \rho\Delta\tau & \text{if} \quad \tau_{ij}(t) \in s_{ch} \\ (1-\rho)\,\tau_{ij}(t) & \text{otherwise} \end{cases} \tag{2}$$

where $0 < \rho \leq 1$ is the evaporation coefficient. Pheromone evaporation is mainly included to avoid pre-mature convergence of the algorithm. It is in favor of the exploration of new areas in the search space. In order to increase the probability of the search by subsequent ants in the promising regions of the search space, good solutions found earlier by the ants are used to update the pheromone.

Different ACO algorithms, such as Ant Colony System (ACS) (Dorigo and Gambardella 1997) or MAX–MIN Ant System (MMAS) (Stützle and Hoos 2000) differ in the way of pheromone updating. In principle, algorithms update pheromone using either the iteration-best solution or the global-best solution. Combination of several solutions found by the ants has also been used. Updating by global-best solution results in a faster convergence, while the iteration-best update allows for more diversification of the search (Stützle and Dorigo 1999).

3 Continuous Ant Colony Algorithms: Concepts and Mathematical Presentation

Ant colony optimization algorithms were basically developed for discrete optimization problems. Therefore, the application of the original ACOs to continuous optimization problems is possible by discretization of the continuous decision space. Thus, the allowable continuous range of decision variables are usually discretized into discrete set of allowable values; and the search process is then conducted over the resulting discrete search space. Since search is restricted to a finite number of discrete values, missing the optimal solution is probable. If the optimal solution is embedded in ignored spaces between allowable values, there may not be any chance to catch it. Therefore, discretization of original space may result in relatively poor performance of the ACO algorithms in continuous problems.

Jalali et al. (2007) employed the inherent potential of multi-colony ant system to tackle a continuous optimization problem. They utilized a multi-colony system with heterogeneous discretization scheme and possibility of information exchange between the colonies to provide a non-homogeneous and dynamic discretization scheme in the search space. The non-homogenous discretized scheme helped them to search for the optimal solution in the continuous search space.

As stated, there have been some attempts to apply ACO meta-heuristic to the continuous domains without discretization requirement. Note that most of these approaches follow rather loosely the original concept of ACOs.

Continuous ACO (CACO) is one of the first attempts to develop and apply an ant-related algorithm to continuous optimization problems (Bilchev and Parmee 1995). The CACO employs a nest situated somewhere in the search space where the ants start searching from. The good solutions found are stored as a set of vectors, which initiate in the nest. At any iteration, ants choose one of the vectors probabilistically. Ants continue the search process from the end-point of the chosen vector by making some random moves from there. Then the vectors are updated with the best results found. Since the CACO does not construct incremental solutions, which is one of the main characteristics of the ACO meta-heuristic, it may not qualify to be an extension of ACO. API algorithm (Monmarche et al. 2000) is another ant-related approach to continuous optimization in which ants perform their search independently, starting from the same nest that is moved periodically. Ants use tandem running which is a type of recruitment strategy. It is an ant-related algorithm that may be used to tackle both discrete and continuous optimization problems. As an ant-based approach, Continuous Interacting Ant Colony (CIAC) was also introduced for continuous optimization (Dreo and Siarry 2002). In CIAC, ants communicate via pheromone deposition and also direct communication. The ants move through the search space where attracted by pheromone laid in spots, and guided by direct communication between the individuals. The CIAC claims to draw its original inspiration from ACO, however, existence of direct communication between ants and absence of constructive incremental solutions may disqualify it as an extension of ACO.

The most recent approach to continuous problems proposed by Socha and Dorigo (2006) is the closest to the spirit of ACO algorithms. As they stated, ACO_R enables one to tackle both continuous and mixed-variable optimization problems. The central idea in ACO_R is the incremental construction of solutions based on the biased (by pheromone) probabilistic choice of solution components. In discrete ACO algorithms, the problem formulation defines the set of available solution components. In order to construct a solution, ants make a probabilistic choice to select c_{ij} from the allowable set of solution components according to Eq. 1. In fact, an ant samples from a discrete probability distribution (Fig. 1a) in order to

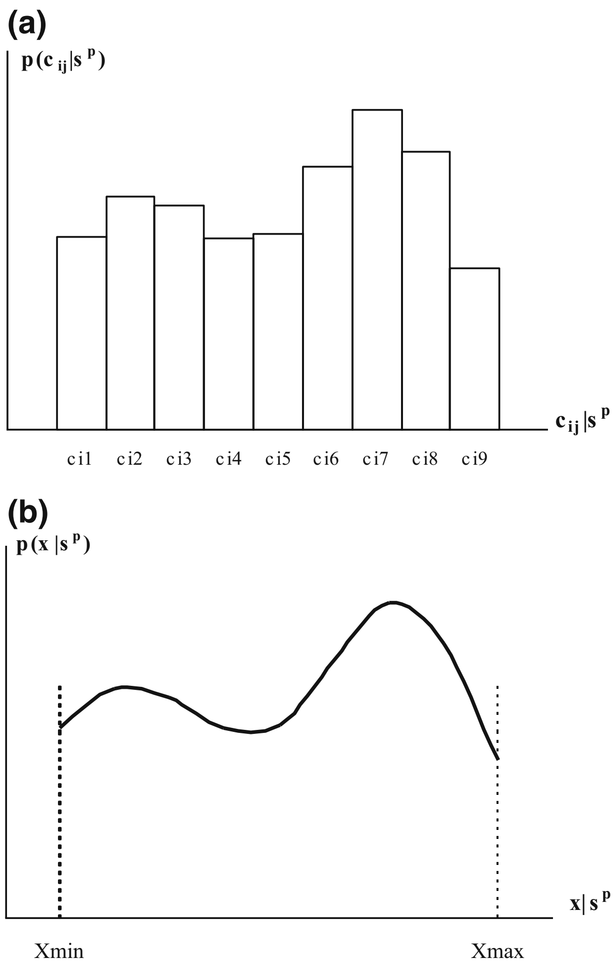

Fig. 1 Schematic of **a** discrete probability distribution of a set of allowable components $[c_{i1},...,c_{i9}]$ in construction step i, **b** continuous probability density function with a possible range of $x \in [x_{\min}, x_{\max}]$ (adopted from Socha and Dorigo 2006)

choose a component to be added to the current partial solution, s^p. The probabilities associated with the elements of the available components in each construction step form such probability distribution (Socha and Dorigo 2006).

As the number of available options at each construction step increases, the discrete probability distribution approaches to a probability density function (Fig. 1b). This concept forms the fundamental idea underlying the algorithm proposed by Socha and Dorigo (2006). In this case, an ant samples the solution components from a continuous probability density function (pdf) instead of choosing a solution component from an allowable set according to Eq. 1.

Gaussian function, as the most popular pdf, has some clear advantages such as reasonably easy way of sampling, although it is not able to describe a situation where two disjoint areas of the search space are promising (i.e. bi-modal pdf). In order to overcome this problem, a Gaussian kernel pdf has been suggested (Socha and Dorigo 2006).

Weighted sum of several one-dimensional Gaussian functions form a Gaussian kernel pdf, $G^i(x)$ as, (Socha and Dorigo 2006):

$$G^i(x) = \sum_{l=1}^{k} \omega_l g_l^i(x) = \sum_{l=1}^{k} \omega_l \frac{1}{\sigma_l^i \sqrt{2\pi}} e^{-\frac{\left(x-\mu_l^i\right)^2}{2\sigma_l^{i2}}} \tag{3}$$

in which k is the number of single pdfs, $g_l^i(x)$, contributing to Gaussian kernel pdf at ith construction step. $G^i(x)$ is parameterized with three vectors of parameters $\left(\boldsymbol{\omega}, \boldsymbol{\mu}^i, \boldsymbol{\sigma}^i\right)$. In any function, $\boldsymbol{\omega}$ defines the vector of weights of the individual Gaussian functions, $\boldsymbol{\mu}^i$ represents the vector of means, and $\boldsymbol{\sigma}^i$ defines the vector of standard deviations. The size of all these vectors is equal to the number of Gaussian functions constituting the Gaussian kernel (k), hence $|\boldsymbol{\omega}| = \left|\boldsymbol{\mu}^i\right| = \left|\boldsymbol{\sigma}^i\right| = k$.

Gaussian kernel pdf provides more flexible sampling shape, compared to a single Gaussian function and allows a relatively easy sampling (Fig. 2).

To define the three noted vectors in Gaussian kernel pdf, a solution archive (T) is formed. To update the pheromone concentration, a certain number (k) of the solutions are kept in the solution archive. In an n-dimensional problem, the archive stores the values of n variables associated with any selected solution s_l. The ith

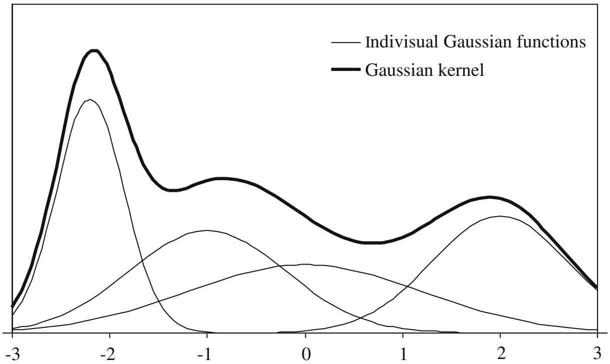

Fig. 2 Example of a Gaussian kernel pdf consisting of four Gaussian functions (illustration is limited to the range $x \in [-3, 3]$) (adopted from Socha and Dorigo 2006)

component of the ith solution is hereby denoted by s_l^i. Figure 3 shows the structure of the solution archive. The size of an archive is equal to k which is a parameter of the algorithm. At the end of any iteration, pheromone is updated by adding the set of new superior solutions to the solution archive. Therefore, only the best solutions remain in the archive, and will be used to guide the ants in the search process.

To update the archive at the end of any iteration, all current solutions (newly produced ones and those already in the archive) are evaluated and ranked according to their fitness values. Afterward, the k number of superior solutions are sorted and stored in the archive according to their ranks (i.e., solution s_l has rank l). The remaining solutions are discarded and the next iteration of the algorithm launches. Clearly, the archive size does not change through updating process; as the entering of superior solutions to the archive continues until the archive fills up.

The solutions in the archive are used to determine the vectors ω, μ^i, and σ^i which define the final shape of the Gaussian kernel pdf. As illustrated in Fig. 3, in each dimension $i = 1, \ldots, n$ of the problem, a different G^i is included. Each G^i is formed by the ith variable values of the archived solutions.

For any Gaussian kernel pdf, G^i, Socha and Dorigo (2006) considered the values of the ith variable of all the solutions in the archive become the elements of the vector $\boldsymbol{\mu}^i$:

$$\mu_l^i = s_l^i \tag{4}$$

The weight, ω_l, of the solution s_l, and corresponding single pdf, is a value of the Gaussian function with argument l, mean 1.0, and standard deviation qk, as (Socha and Dorigo 2006):

$$\omega_l = \frac{1}{qk\sqrt{2\pi}} e^{-\frac{(l-1)^2}{2q^2k^2}} \tag{5}$$

where q is a tuning parameter of the algorithm which must be specified. The influence of q is similar to adjusting the balance between the iteration-best and the global-best

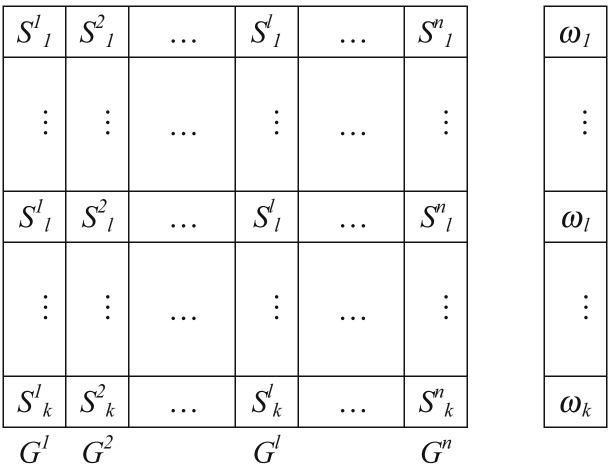

Fig. 3 The solution archive used in ACO_R. The solutions are stored according to their rank (s_1 has the best quality). ω is the weight of each Gaussian function proportional to the rank of associated solution. Hence, $\omega_1 \geq \ldots \geq \omega_l \geq \ldots \geq \omega_k$. G^i is the kernel pdf at ith construction step using only the ith component of all k solutions in the archive (adopted from Socha and Dorigo 2006)

pheromone updating in ACO. For small values of q, the best-ranked solutions are strongly preferred. When q approaches zero, only the Gaussian function of the best solution found so far is used for constructing new solutions. For larger values of q, the probability becomes more uniform; and the algorithm samples the search space based on a larger number of reasonably good solutions. In this case, the search is more diversified and the final optimal solution is more reliable. However, the higher robustness usually means slower convergence speed.

Sampling the Gaussian kernel pdf as defined by Eq. 3 is accomplished with the following procedure. Before starting a solution construction, each ant chooses one of the individual Gaussian functions, $g_l(x)$, for all n construction steps which allows exploiting the possibly existing correlation between the decision variables. Then, a Gaussian function with rank l is chosen with the probability distribution expressed as (Socha and Dorigo 2006):

$$p_l = \frac{\omega_l}{\sum_{j=1}^{k} \omega_j} \quad \forall l = 1, \ldots, k \tag{6}$$

where ω_j is the weight of the jth Gaussian function.

It should be noticed that each ant samples the Gaussian functions of the same rank at any construction step. However, the shapes of the chosen Gaussian functions differ from one step to another. For step i, $\mu_l^i = s_l^i$, and σ_l^i is calculated dynamically as (Socha and Dorigo 2006):

$$\sigma_l^i = \xi \sum_{e=1}^{k} \frac{\left|s_e^i - s_l^i\right|}{k-1} \tag{7}$$

in which, at each step i, the average distance from the chosen solution s_l to other solutions in the archive is determined and multiplied by the parameter ξ. The parameter $\xi > 0$, behaves similar to the pheromone evaporation rate in ACO. Appropriate selection of ξ will affect the search process to be less biased towards the points that have been already explored and stored in the archive. The higher values of ξ, will slow down the convergence of the algorithm. This whole process is repeated for any ith construction step, $i = 1, \ldots, n$, and the average distance is calculated upon the variable values at the ongoing step.

The number of memorized solutions in the archive, (k), determines the complexity of the resulted G^is. As inferred from Figs. 2 and 3, larger archive increases the complexity of the kernel pdfs owing to the fact that every solution in the archive represents a single pdf at each construction step.

In the first iteration, if no prior information is provided, the kernel pdfs may be initialized by selecting a more or less uniform distribution over the search domain (a, b). In this case, a set of k normal distributions $G^i(x)$ with uniformly distributed means are initialized and employed as:

$$G^i(x) = \sum_{l=1}^{k} \frac{1}{k} g_l^i \left(x, a + (2l-1)\frac{b-a}{2k}, \frac{b-a}{2k} \right) \tag{8}$$

where ω_l (Eq. 3) is equal to $\frac{1}{k}$, and the second and third terms define the mean and standard deviation of the ith pdf, respectively. At the end of the first iteration, the fitness function is evaluated for each solution and the first k solutions are transferred to the archive T for pheromone updating.

4 Proposed Improvements

This section presents the proposed improvements on the original ACO_R algorithm developed by Socha and Dorigo (2006). The improved version benefits from (1) adaptation operator and (2) explorer ants, which highly affect the performance of the algorithm.

4.1 Adaptation Operator

As mentioned, parameter q in Eq. 5 plays an important role in calculation of the weight of each Gaussian function in the Gaussian kernel pdfs.

For small values of q, a significant proportion of the total weight will be allotted to the most superior solutions in the archive. That is, the weights' allocation is excessively biased toward the high-ranked solutions. Since ants choose any single pdf probabilistically upon its weight, the superior solutions will be strongly preferred to inferior ones. This affects the search space to narrow down around the best available solutions, resulting in a rapid convergence. As an extreme case, when q approaches to zero, all ants will only choose the Gaussian function associated with the best-so-far solution in the archive. Hence, the algorithm converges before efficiently exploring the entire search space. Indeed, the final solution strongly depends on the random population in the first iteration.

For extremely large values of q, the weight distribution will be almost uniform among solutions in the archive. Therefore, single pdfs of different ranks will be available to sample from. As a result, ants sample from different parts of the search space; and a wide range of diverse solutions is generated. Although the decision space is broadly searched, a very slow convergence may result. The final solution may be more robust and reliable; however, the decision space is extensively searched with the cost of reduction in convergence rate. Therefore, an extensive sensitivity analysis might be needed to trade-off between the higher rate of convergence and the more robust solution.

On the other hand, the most efficient search is due to an initial wide skim of decision space; and accurate scan, afterward. While skimming the decision space, a rough estimation of promised areas is exerted; and the most probable regions encircling the optimum solution are defined. The latter scanning finely searches through superb areas, and enables the algorithm to precisely tune the final solution. Such process is executed only if either the value of q could change through advancement of the algorithm or an extensive sensitivity analysis may be needed to locate a good near optimal solution. For instance, the small value of q implies the large value for archive size to prevent the pre-mature convergence at initial generations. But, large archive extensively and unfavorably increases the computational effort and run-time. Hence, the archive size must be tuned finely in order to result a satisfactory performance of the algorithm without great computational effort. As clarified, the constant value for q imposes some unsuitable conditions on the algorithm.

To enhance the performance of the original algorithm, this paper presents an adaptation operator which automatically changes the value of q through successive iterations. The influence of the adaptation process on the algorithm is the same as choosing a small q and a large archive size in the original ACO_R but, by contrast, it does not require the great computational, run-time and sensitivity analysis effort.

In adaptation process, a primary rough amount of q is given to the algorithm; and it is gradually adjusted as the algorithm proceeds. The initial value is relatively large to lead a comprehensive search of the decision space. As the algorithm advances, this value is gently reduced. Such procedure causes the initial general search adaptively concentrates around the high-quality regions. Furthermore, the archive size is set to a reasonably moderate value. The proposed methodology benefits from both global and local search concepts. At initial iterations, when q is large, ants extensively explore different regions of the search space, locating diverse solutions. The adaptation routine reduces the value of q according to a pre-defined scheme. As iteration proceeds and q becomes smaller, few highly ranked solutions receive more chance to be chosen by the ants and search process is slowly localized. Therefore, the convergence speeds up and the best solution is tuned finely.

The following expression is defined by the authors to adaptively reduce the value of q through successive iterations:

$$q_{it} = q_{it-1} \times A_{it} \tag{9}$$

In which q_{it} and q_{it-1} are the values of q in iteration it and $it-1$, respectively; and A_{it} is the value of adaptation operator in iteration it. It changes non-increasingly through successive iterations. The following relation is proposed to calculate A_{it} at given iteration it:

$$A_{it} = \begin{cases} 1 & \text{if } \left(\frac{\text{Mean}(f_{1\ldots m})}{\text{Mean}(f_{1\ldots n})}\right)_{it} = \left(\frac{\text{Mean}(f_{1\ldots m})}{\text{Mean}(f_{1\ldots n})}\right)_{it-1} \\ \frac{\text{Mean}(f_{1\ldots m})}{\text{Mean}(f_{1\ldots n})} & \text{if } \left(\frac{\text{Mean}(f_{1\ldots m})}{\text{Mean}(f_{1\ldots n})}\right)_{it} \neq \left(\frac{\text{Mean}(f_{1\ldots m})}{\text{Mean}(f_{1\ldots n})}\right)_{it-1} \end{cases} \tag{10}$$

where $\text{Mean}(f_{1\ldots m})$ and $\text{Mean}(f_{1\ldots n})$ are the mean fitness values over first m and n($m < n$) ranked solutions in the archive at any iteration, respectively; and $(\cdot)_{it}$ and $(\cdot)_{it-1}$ return the associated values to iteration it and $it-1$, respectively. Note that the solutions in the archive are stored in ascending order where rank 1 is assigned to the solution with the least fitness value. Hence, for $m < n$, the values of A_{it} remain less than or equal to one.

Definition of the necessary conditions is the most important point in Eq. 10. These conditions show how the changes in the mean fitness values of the solutions in the archive affect the restriction on the search space. If this criterion in the archive remains unchanged through iterations it and $it-1$, the first condition is satisfied; and A_{it} becomes equal to 1. Thus, according to Eq. 9, the value of q does not reduce for current iteration which means more localization is not suitable. As the algorithm fails to find any superior solution, it is reasonable not to impose more restriction on the search space, keeping the search diversity at the same level as the last iteration.

On the other hand, if the mean fitness values of the solutions in the archive change; the second condition will be satisfied, and the value of adaptation operator is reduced upon the severity of the changes in the archive. As the algorithm finds some desirable solutions, the value of $\frac{\mathrm{Mean}(f_{1\ldots m})}{\mathrm{Mean}(f_{1\ldots n})}$ may noticeably drop. For $n >> m$, this drop will be more pronounced and the search space will be severely reduced accordingly.

In problems with large allowable range for decision variable, undesirable fast decreasing rate of A_{it} during initial iterations is possible. In order to reduce the chance of pre-mature convergence of the algorithm, a start criterion for adaptation process may be beneficial. The start criterion controls the stage where the adaptation operator must become active. It should be defined upon the problem under consideration. In a highly constrained optimization problem, one may activate the adaptation operator when at least one ant decides in feasible space.

4.2 Explorer Ants as Mutation Operators

This paper borrows the concept of explorer ants from Jalali et al. (2005). They employed the explorer ants as local search agents in the standard ACO algorithm to search the decision space in a completely random manner. This non-biased manner enables the algorithm to meet new areas where have not yet visited by the typical ants.

The behavior of explorer ants in this paper differs from those proposed by Jalali et al. (2005). Here, the explorer ants are biased towards the superior areas of search space but less severe than the typical ants. They first choose a single solution from the archive in order to use it during the construction steps. At any construction step, an explorer ant generates a trial value from the selected Gaussian function similar to the typical ants. Then, it is permitted to probabilistically mutate the value of the trial decision variable within a pre-defined range (Fig. 4). It provides a localized search

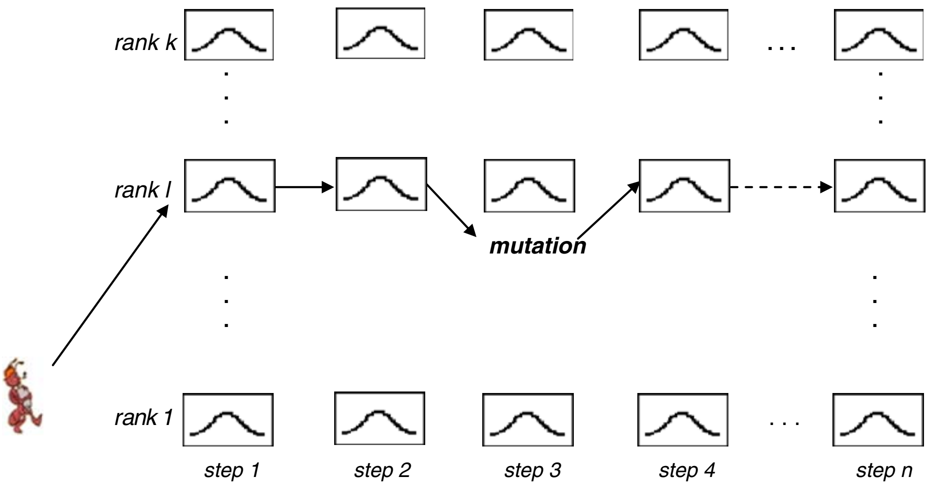

Fig. 4 Schematic presentation of solution construction by an explorer ant

possibility for the explorer ants. As the mutation range extends, the explorer ants are less impressed by the Gaussian functions; and a relatively large area around the initially generated values will be randomly searched. Therefore, the final value has more random essence. On the contrary, if the mutation range is small; the search process will be concentrated to a rather small area around the initial value.

The most important role of the explorer ants is to help original algorithm escape from the local optimums. When algorithm traps in a local optimum, almost all ants search about the same value for some decision variables. As the archive consists of superior solutions, some of these solutions may be selected to enter the archive. Since, at some steps, the variable values of all solutions are almost equal, the means of associated Gaussian functions become nearly the same. From Eq. 7, the standard deviation of these Gaussian functions approach zero. That is, all Gaussian functions in some steps become overlaid; and they represent a certain value rather than a range of values in search space. In this condition, all ants will generate the mean value of Gaussian pdf because the standard deviation, as the only factor to widen the pdf, is almost equal to zero. Since the mean values of Gaussian pdfs remain unchanged, extra search will not help the algorithm to escape from the local optimum. As stated, all Gaussian functions become overlaid, and jumping from one pdf to another will not change the result. In other words, if the original ACO_R traps in local optimums, there is almost no chance to survive. By contrast, discrete ACOs always have a little chance to survive from the local optimums. Any option at any construction step must have a minimum amount of pheromone all through the algorithm advancement. Certainly, pheromone existence, regardless of its value, provides a chance of generating diverse solutions even in local-trapped conditions.

Definition of explorer ants in continuous ant algorithm is quite helpful to escape from local optimums. Explorer ants can mutate the initial generated value, which leads to a different value from the mean of chosen pdf. If the resultant solution enters the archive, a pdf with a new mean value enters to local-trapped steps. Consequently, the standard deviations of all pdfs become greater than zero; and Gaussian functions become widened. This helps the population to generate more diverse solutions. In this case, jumping among Gaussian functions will certainly change the results.

From descriptions, the following expression is proposed by the authors to determine the mutation range:

$$\mathrm{MR}_{it}^{i} = f\left(\sigma_{it}^{i}\right) \tag{11}$$

in which, MR_{it}^{i} is the mutation range and $\boldsymbol{\sigma}_{it}^{i}$ is the vector of standard deviation at step i and iteration it. Equation 11 implies that, at any construction step, mutation range is calculated as a function of the standard deviations which steadily change through the algorithm. However, one may choose a rough mutation range upon own experience. The function of explorer ants may be more useful in multimodal problems which frequently face local-trapping issue.

5 Applications of the Model

To compare the performance of the proposed modified continuous ant algorithm with those of some other algorithms, a number of mathematical test functions are employed. Thereafter, both original and modified versions of ACO_R are applied to optimally operate a hydropower reservoir.

5.1 Mathematical Functions

In order to compare the performance of the proposed algorithm with the original ACO_R and few other methods for continuous optimization, a set of benchmark problems are employed as summarized in Table 1 (Socha and Dorigo 2006). To have a fair comparison, the same initialization intervals and required accuracy are used as reported in the literature.

The performances of the algorithms have been judged based on the mean number of function evaluations to reach the required accuracy. That is, certain accuracy is known as the stop criterion. The stop criterion is not the same for all test functions as employed by Socha and Dorigo (2006):

$$\begin{cases} |f - f^*| < \varepsilon_1 f + \varepsilon_2 & \text{function}: \ B_2, R_2, Z_2, \text{GP, MG and SM} \\ |f - f^*| < \varepsilon_{\min} & \text{functions}: \ \text{CG,TB and EL} \end{cases} \tag{12}$$

where f is the function value associated to the best solution found by the applied method; f^* is the prior known optimum; ε_1 and ε_2 are the relative and absolute

Table 1 Summary of test functions

Function	Formula
$B_2 \vec{x} \in [-100, 100]^n$, $n = 2$	$f_{B_2}(\vec{x}) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1) - \frac{2}{5}\cos(4\pi x_2) + \frac{7}{10}$, G. Min. = 0
Rosenbrock(R_n) $\vec{x} \in [-5, 10]^n$, $n = 2$	$f_{R_n}(\vec{x}) = \sum_{i=1}^{n-1}\left(100\left(x_i^2 - x_{i+1}\right)^2 + (x_i - 1)^2\right)$, G. Min. = 0
Zakharov(Z_n) $\vec{x} \in [-5, 10]^n$, $n = 2$	$f_{Z_n}(\vec{x}) = \left(\sum_{i=1}^{n} x_i^2\right) + \left(\sum_{i=1}^{n} \frac{ix_i}{2}\right)^2 + \left(\sum_{i=1}^{n} \frac{ix_i}{2}\right)^4$, G. Min. = 0
Goldstein and Price (GP) $\vec{x} \in [-2, 2]^n$, $n = 2$	$f_{GP}(\vec{x}) = \left[1 + (x_1 + x_2 + 1)^2 \times (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\right] * \left[30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)\right]$, G. Min. = 3
Martin and Gaddy (MG) $\vec{x} \in [-20, 20]^n$, $n = 2$	$f_{MG}(\vec{x}) = (x_1 - x_2)^2 + \left(\frac{x_1 + x_2 - 10}{2}\right)^2$, G. Min. = 0
Sphere model (SM) $\vec{x} \in [-5.12, 5.12]^n$, $n = 6$	$f_{SM}(\vec{x}) = \sum_{i=1}^{n} x_i^2$, G. Min. = 0
Cigar(CG) $\vec{x} \in [-3, 7]^n$, $n = 10$	$f_{CG}(\vec{x}) = x_1^2 + 10^4 \sum_{i=2}^{n} x_i^2$, G. Min. = 0
Tablet(TB) $\vec{x} \in [-3, 7]^n$, $n = 10$	$f_{TB}(\vec{x}) = 10^4 x_1^2 + \sum_{i=2}^{n} x_i^2$, G. Min. = 0
Ellipsoid(EL) $\vec{x} \in [-3, 7]^n$, $n = 10$	$f_{EL}(\vec{x}) = \sum_{i=1}^{n}\left(100^{\frac{i-1}{n-1}} x_i\right)^2$, G. Min. = 0

Table 2 Performances of proposed algorithm and other probability-learning algorithms

Test function	Proposed	ACO_R^a	$(1+1)$ES[a]	CSA-ES[a]	CMA-ES[a]	IDEA[a]	MBOA[a]
Ellipsoid	2,082	11,570	293,700	489,500	4,450	7,120	62,300
Cigar	2,195	5,376	2,342,400	3,072,000	3,840	17,664	46,080
Tablet	1,942	2,567	118,082	166,855	4,364	7,444	61,608

[a]Reported by Socha and Dorigo (2006)

errors, respectively; and $\varepsilon_{\min}$ is the required accuracy. Here, the same values reported in the literature are used, i.e. $\varepsilon_1 = \varepsilon_2 = 10^{-4}$ and $\varepsilon_{\min} = 10^{-10}$ (Socha and Dorigo 2006).

Since the allowable ranges of the decision variables were relatively small in the test functions, no special start-criterion for the adaptation process on q was required, and it started from the second iteration. Tables 2, 3, and 4 present the results obtained by proposed algorithm based on 20 independent runs and other ant and non-ant based methods reported by Socha and Dorigo (2006). The methods presented in Tables 2, 3, and 4 are discussed by Socha and Dorigo (2006) in appropriate details. Reported values are the mean number of function evaluations and the best performances are marked in bold.

As the results for all algorithms were not fully reported, it was not possible to conduct detail statistical analysis. Hence, the average number of function evaluations was selected as the only criterion for the purpose of performance comparison.

In the case of tuning parameters, the archive of varying size from 5 to 20 is used for different test functions for the proposed algorithm which is comparable with those employed by Socha and Dorigo (2006). They used the archive size $k = 50$ which redundantly increases the calculation process. Since the proposed algorithm tunes the parameter q adaptively, this parameter initially takes a relatively large value without requiring serious sensitivity analysis. In the original algorithm, however, the parameter q must be finely tuned through extensive trial tests.

The results demonstrates the satisfactory performance of the proposed modifications on the ACO_R in given benchmark problems. The performance is pronounced in the minimization problems with dimensions varying from 2 to 10. In comparison to other ant and non-ant related algorithms, the proposed algorithm performed better for all test problems. It required much less mean number of function evaluations in order to fulfill the stop criteria. The results of ACO_R in comparison with other methods are discussed by Socha and Dorigo (2006). Here, results show that the

Table 3 Performances of proposed algorithm and other ant-related algorithms

Test function	Proposed	ACO_R^a	CACO[a]	API[a]	CIAC[a]
B2	160	544	–[b]	–[b]	11,968
Rosenbrock (R2)	591	820	6,806	9,840	11,480
Goldstein and Price (GP)	155	384	5,376	–[b]	23,424
Martin and Gaddy (MG)	125	345	1,725	–[b]	11,730
Sphere model (SM)	361	781	21,868	10,153	49,984

[a]Reported by Socha and Dorigo (2006)

[b]Results were not available

Table 4 Performances of proposed algorithm and other meta-heuristics for continuous optimization

Test function	Proposed	ACO_R^a	CGA[a]	ECTS[a]	ESA[a]
B2	160	544	430	–[b]	–[b]
Rosenbrock (R2)	591	820	960	480	816
Goldstein and Price (GP)	155	384	416	231	786
Zakharov (Z2)	64	293	624	195	15,795

[a]Reported by Socha and Dorigo (2006)

[b]Results were not available

proposed improvements help the modified algorithm to outperform other methods in functions under consideration. Faster convergence of the proposed algorithm in comparison with original one may be related to the behavior of the adaptation operator. In original ACO_R, a large archive and a small value for q were reasonably considered by Socha and Dorigo (2006). Certainly, the constant small value for q leads the method to converge to the global solution in a relatively slow rate. On the contrary, the modified version benefits from the adaptation operator which adjusts the convergence speed through the advancement of the algorithm. It gradually hastens the convergence speed while visiting most of the promising areas of the search space. In addition, some suitable mutations may occur by means of explorer ants, which locate the algorithm properly near the global solution.

5.2 Operation of Hydropower Reservoir

In order to illustrate the performance of the proposed algorithm in highly nonlinear-non-convex water resources problems, operation of a hydropower reservoir has been considered.

The objective function is to minimize a measure of the sum of relative deficit of power generation from the installed capacity of the hydropower plant over operating period. The problem is considered for 240 operating periods for which the data is taken from Dez hydropower reservoir in south of Iran. The reservoir's minimum and maximum capacities are known to be 830 and 3340 MCM, respectively. Time distribution of the inflow to the reservoir for the assumed 240 operating periods is presented in Fig. 5. Each operating period is considered as a solution construction step with reservoir releases as decision variables. All ants assume a specified initial storage and calculate the reservoir storages according to the inflows to the reservoir using the continuity equation. Upper and lower limits for both reservoir releases and storages have been considered as constraints to the model. In mathematical statement, the objective function is defined as:

$$\text{O.F.} = \text{Min} \sum_{t=1}^{T} \left(1 - \frac{P_t}{\text{Power}} \right) \tag{13}$$

where P_t (kilowatt) is the generated power at time period t; Power (kW) = 650 kW is the installed capacity of the hydropower plant; and T is the total number of operating periods.

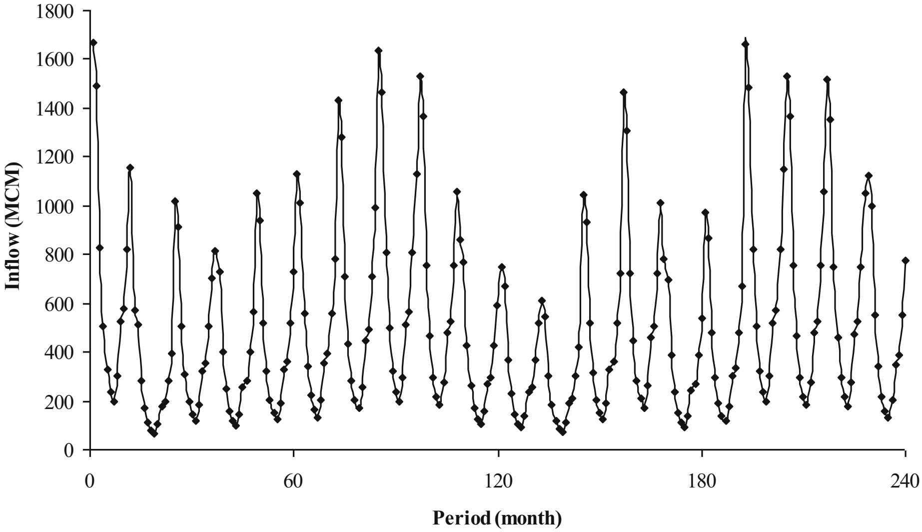

Fig. 5 The volumes of inflows to Dez reservoir during 240 operating periods

The problem is subjected to:

$$S_{t+1} = S_t + I_t - R_t \tag{14}$$

$$P_t\,(\mathrm{kW}) = \mathrm{Min}\left(\frac{e}{\mathrm{PF}}\frac{\gamma Q_t H_t}{1,000}, \mathrm{Power}\right) \tag{15}$$

$$H_t = \frac{h_t + h_{t+1}}{2} - \mathrm{TW} \tag{16}$$

$$h_t = a + b \times S_t + c \times S_t^2 + d \times S_t^3 \tag{17}$$

$$S_{\min} < S_t < S_{\max} \tag{18}$$

$$R_{\min} < R_t < R_{\max} \tag{19}$$

where γ (newton per cubic meter) is the specific weight of water; Q (cubic meter per second) is the discharge entering turbine at time period t; H_t (meter) is the effective head for energy production at time period t; e is efficiency coefficient; PF is the plant factor; h_t (meter) is water head available at time period t; and TW is the head of tail water; a, b, c and d are the coefficients determined by topography of the dam site. In the case, $e = 0.9$, $\mathrm{PF} = 0.417$, $\mathrm{TW} = 172$ masl, $S_{\min} = 830$ MCM, $S_{\max} = 3,340$ MCM, $R_{\min} = 0$, $R_{\max} = 1,000$ MCM, $a = 249.83$, $b = 5.9 \times 10^{-2}$, $c = -1 \times 10^{-5}$ and $d = 2 \times 10^{-9}$ are specified.

Table 5 Summary of parameters used in original and improved ACO_R for hydropower reservoir problem with 240 operating periods

	Case A	Case B
Total ants	40	40
No. of iterations	6,000	6,000
Archive size (k)	20	20
ξ	0.8	0.8
Q	0.5	0.1

Equation 15 imposes an upper bound on the actual energy production by taking the smaller value from the installed capacity and the potential power of the released water from the reservoir.

The nonlinear non-convex hydropower formulation is considered to emphasize the capabilities of the proposed algorithm. The decision variables are defined as releases during operating periods. An ant is penalized if the storage volume at any operating period violates the storage constraint represented by Eq. 18. The penalty function for reservoir storage deviation from its allowable range is expressed as:

$$\text{Penalty}_t = \begin{cases} C \times \left(1 - \frac{S_t}{S_{\min}}\right) \text{ if } S_t < S_{\min} \\ C \times \left(\frac{S_t}{S_{\max}} - 1\right) \text{ if } S_t > S_{\max} \end{cases} \tag{20}$$

where Penalty_t defines the value of the penalty imposed on the solution for violating the storage constraint at time period t; and $C(>1)$ is the penalty factor which accounts for the severity of the storage constraint violation. The penalty factor of $C = 10$ was used in the case considered. If any ant is penalized at different time periods during solution construction, a total penalty value equal to the sum of all penalties will be imposed on the agent.

In order to illustrate the contributions of the proposed adaptation operator and explorer ants to overall performance of the algorithm in the hydropower reservoir operation problem, two different combination of the operators (identified as cases A and B) are considered. Case A assumes an initial large value for q, whereas case B starts with a quite smaller value. As expected, initially large value of q relatively expands the search space causing slow rate of convergence. Whereas, small value of q results a fairly intensive search causing potential trapping in local optimums. Therefore, it is anticipated that adaptation in case A and explorer ants in case B might efficiently improve the quality of the final solution. Table 5 summarizes the

Table 6 Results obtained for hydropower reservoir problem with 240 operating periods using case-A conditions

	Original ACO_R	Improved ACO_R			
		Exp. = 0[a]	Exp. = 5[b]	Exp. = 8	Exp. = 10
Mean	43.315	30.848	26.202	26.123	25.552
SD	5.53	4.77	2.20	1.25	1.05

[a] Adaptation operator without explorer ants

[b] Number of explorer ants

Table 7 Results obtained for hydropower reservoir problem with 240 operating periods using case-B conditions

	Original ACO_R	Improved ACO_R			
		Exp. = 0[a]	Exp. = 5[b]	Exp. = 8	Exp. = 10
Mean	29.790	30.282	24.946	24.464	24.183
SD	1.61	1.70	0.66	0.89	0.62

[a]Adaptation operator without explorer ants

[b]Number of explorer ants

values of the parameters used for cases *A* and *B* corresponding to the problem with 240 operating periods.

Results of the model application to the defined hydropower reservoir operation for 240 operating periods under cases *A* and *B* are presented in Tables 6 and 7. According to Table 6, when adaptation operator is implemented in case *A*, the mean value and standard deviation of the objective function over 20 independent runs drop from 43.315 and 5.53 to 30.848 and 4.77 for original and improved algorithms, respectively. Inclusion of the explorer ants further drops the objective value to 26.202 as five explorer ants are employed. It is interesting to note that the standard deviation has been reduced by 50%. According to Table 7, in case *B*, the contribution of the adaptation operator to improvement of the solutions is not as significant. In this case, however, the explorer ants play an important role in solution improvement. As presented in Table 7, the first five explorer ants reduced the mean value and the standard deviation of the objective function by more than 17% and 60%,

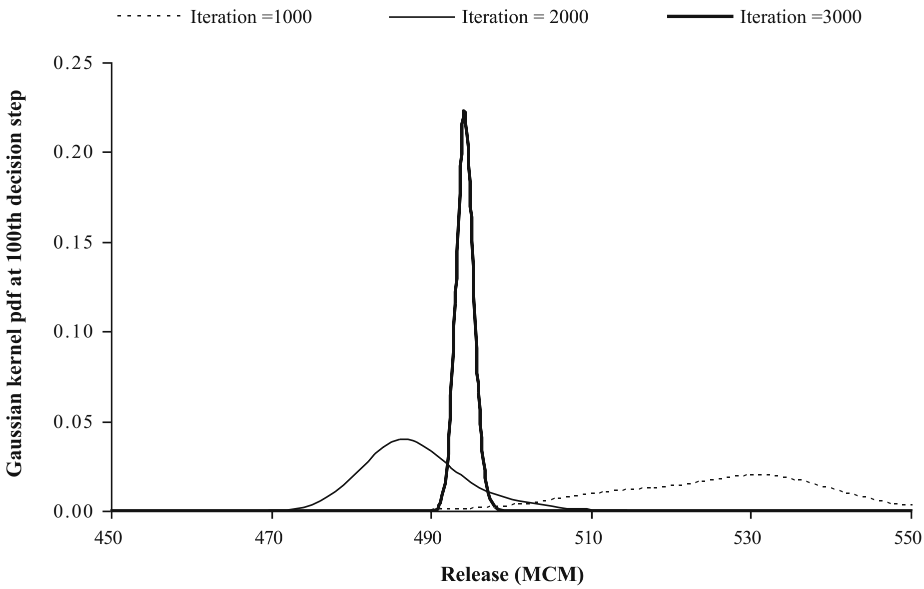

Fig. 6 The Gaussian kernel pdfs obtained by the best ant for 100th decision step at different iterations through the advancement of the algorithm

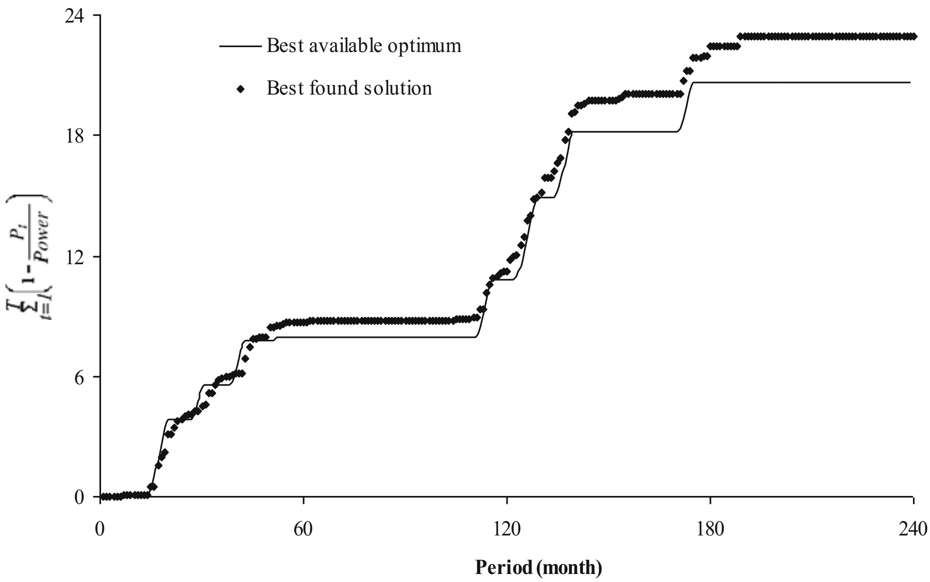

Fig. 7 Comparison between the results of the best found solution by the improved algorithm with those of the best available optimum

respectively. The best solutions found with the improved algorithm under cases *A* and *B* are determined as 24.04 and 22.95 units, respectively. Results clearly show how effectively the proposed improvements locate better near optimum solutions, both for the best solution and the average performance over large number of independent runs. Further consideration on results clarifies the remarkably less sensitivity of the improved algorithm, in comparison with the original one, to the initial value of parameter q.

Figure 6 illustrates how the Gaussian kernel pdf, obtained by the best ant at one of the construction steps, changes through the advancement of the algorithm. As shown, the Gaussian kernel pdf is gradually retracted from a relatively wide area in the decision space to a small area around the final value. At initial iterations, a rather uniform Gaussian kernel pdf provides a broad search for the ants while by the advancement of the algorithm, the Gaussian kernel pdf is contracted and the search process is finely tuned.

Figure 7 presents the cumulative values for relative deviation of the generated power from the installed capacity of the hydropower plant under the best solution. The results of the best solution obtained by the proposed improved ACO_R algorithm follows almost the same pattern as those of the best available optimum, the objective value of which is reported as 20.62 (Moeini 2007).

6 Conclusion

Ant colony optimization was initially proposed for discrete search space while in continuous domain, discretization of the search space has been practiced. Most of the proposed ant-based methods for continuous optimization are far from the original spirit of ant colony optimization algorithm. This paper presented an improved version of ACO_R algorithm which integrated adaptation operator and explorer ants into the original structure. Performance of the proposed model on some mathematical benchmark problems was quite satisfactory in comparison with those of some other ant and non-ant based methods. With the same stop criteria, the proposed model required remarkably less number of function evaluations. Inclusion of adaptation operator and explorer ants in the algorithm reduced the possibility of trapping into local optimums. Application of the algorithm to a single hydropower reservoir operation problem showed that the model is relatively robust in locating the good near optimal solutions. More desirable value of the objective function and lower standard deviation of the results may be realized as the strength of the proposed improved version of the algorithm compared to the original method.

The results of the model application showed that the adaptation operator was more effective in cases with large initial values for parameter q. For all cases considered in this study, few numbers of explorer ants significantly improved the overall performance of the algorithm. The profound assessment of the results demonstrated that the improved algorithm is not seriously sensitive to the number of explorer ants. That is, the number of explorer ants may not impose any serious restriction on the model application and results. It was highly recommended to include a small population of explorer ants along with the proposed adaptation operator to enhance the algorithm's performances.

References

Abbaspour KC, Schulin R, van Genuchten MT (2001) Estimating unsaturated soil hydraulic parameters using ant colony optimization. Adv Water Resour 24(8):827–841. doi:10.1016/S0309-1708(01)00018-5

Bilchev G, Parmee IC (1995) The ant colony metaphor for searching continuous design spaces. In: Fogarty TC (ed) Proceedings of the AISB workshop on evolutionary computation, vol 993 of LNCS. Springer, Berlin, pp 25–39

Bozorg Haddad O, Afshar A (2004) MBO (Marriage Bees Optimization), a new heuristic approach in hydrosystems design and operation. In: Proceedings of the 1st international conference on managing rivers in the 21st century: issues and challenges. Penang, Malaysia, 21–23 Sep 2004, pp 499–504

Dorigo M (1992) Optimization, learning and natural algorithms. PhD thesis, Politecnico di Milano, Milan, Italy

Dorigo M, Gambardella LM (1997) Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Trans Evol Comput 1(1):53–66. doi:10.1109/4235.585892

Dorigo M, Maniezzo V, Colorni A (1996) The ant system: optimization by a colony of cooperating ants. IEEE Trans Syst Man Cybern 26:29–42. doi:10.1109/3477.484436

Dreo J, Siarry P (2002) A new ant colony algorithm using the hierarchical concept aimed at optimization of multiminima continuous functions. In: Dorigo M, Caro GD, Sampels M (eds) Proceedings of the 3rd international workshop on ant algorithms (ANTS 2002), vol 2463 of LNCS. Springer, Berlin, pp 216–221

Esat V, Hall MJ (1994) Water resources system optimization using genetic algorithms. In: Hydroinformatics '94, Proceedings of the 1st international conference on hydroinformatics. Balkema, Rotterdam, The Netherlands, pp 225–231

Fahmy HS, King JP, Wentzel MW, Seton JA (1994) Economic optimization of river management using genetic algorithms (paper no 943034). International summer meeting, American Society of Agricultural Engineers, St Joseph, MI

Jalali MR, Afshar A, Mariño MA (2005) Improved ant colony optimization algorithm for reservoir operation (technical report). Hydroinformatics Center, Civil Engineering Department, Iran University of Science and Technology, Tehran, Iran

Jalali MR, Afshar A, Marino MA (2007) Multi-colony ant algorithm for continuous multi-reservoir operation optimization problem. Water Resour Manage 21:1429–1447. doi:10.1007/s11269-006-9092-5

Labadie JW (2004) Optimal operation of multireservoir systems: state of-the-art review. J Water Resour Plan Manage 130(2):93–111. doi:10.1061/(ASCE)0733-9496(2004)130:2(93)

Maier HR, Simpson AR, Zecchin AC, Foong WK, Phang KY, Seah HY, Tan CL (2003) Ant colony optimization for design of water distribution systems. J Water Resour Plan Manage 129(3): 200–209. doi:10.1061/(ASCE)0733-9496(2003)129:3(200)

Moeini R (2007) Fully and partially constrained ant algorithm for the optimization of sequential problems: application to optimal operation of reservoirs, M. Sc Thesis, Iran University of Science and Technology, Department of Civil Engineering

Monmarche N, Venturini G, Slimane M (2000) On how Pachycondyla apicalis ants suggest a new search algorithm. Future Gener Comput Syst 16:937–946. doi:10.1016/S0167-739X(00)00047-9

Nagesh Kumar D, Janga Reddy M (2006) Ant colony optimization for multi-purpose reservoir operation. Water Resour Manage 20:879–898. doi:10.1007/s11269-005-9012-0

Oliveira R, Loucks D (1997) Operating rules for multireservoir systems. Water Resour Res 33(4):839–852. doi:10.1029/96WR03745

Simpson AR, Maier HR, Foong WK, Phang KY, Seah HY, Tan CL (2001) Selection of parameters for ant colony optimization applied to the optimal design of water distribution systems. In: Ghassemi F et al (eds) Proceedings of the international congress on modeling and simulation. Canberra, Australia, pp 1931–1936

Socha K, Dorigo M (2006) Ant colony optimization for continuous domains. Eur J Oper Res 185(3):1155–1173. doi:10.1016/j.ejor.2006.06.046

Stützle T, Dorigo M (1999) ACO algorithms for the traveling salesman problem. In: Miettinen K, Mäkelä MM, Neittaanmäki P, Périaux J (eds) Evolutionary algorithms in engineering and computer science. Wiley, Chichester, UK, pp 163–183

Stützle T, Hoos HH (1997a) The MAX–MIN ant system and local search for the traveling salesman problem. In: Baeck T, Michalewicz Z, Yao X (eds) Proceedings of IEEE-ICEC-EPS '97, IEEE international conference on evolutionary computation and evolutionary programming conference. IEEE, Piscataway, NJ, pp 309–314

Stützle T, Hoos HH (1997b) Improvements on the ant system: introducing MAX–MIN ant system. In: Smith GD, Steele NC, Albrecht RF (eds) Proceedings of the international conference on artificial neural networks and genetic algorithms. Springer, Berlin, pp 245–249

Stützle T, Hoos HH (2000) MAX–MIN ant system. Future Gener Comput Syst 16(8):889–914. doi:10.1016/S0167-739X(00)00043-1

Wardlaw R, Sharif M (1999) Evaluation of genetic algorithms for optimal reservoir system operation. J Water Resour Plan Manage 125(1):25–33. doi:10.1061/(ASCE)0733-9496(1999)125:1(25)

Zecchin AC, Maier HR, Simpson AR, Roberts A, Berrisford MJ, Leonard M (2003) Max–min ant system applied to water distribution system optimization. In: Post DA (ed) Modsim 2003—international congress on modelling and simulation. Modelling and Simulation Society of Australia and New Zealand, Townsville, Australia, 2, pp 795–800