A hybrid cuckoo–harmony search algorithm for optimal design of water distribution systems
Razi Sheikholeslami, Aaron C. Zecchin, Feifei Zheng and Siamak Talatahari

ABSTRACT
Meta-heuristic algorithms have been broadly used to deal with a range of water resources optimization problems over the past decades. One issue that exists in the use of these algorithms is the requirement of large computational resources, especially when handling real-world problems. To overcome this challenge, this paper develops a hybrid optimization method, the so-called CSHS, in which a cuckoo search (CS) algorithm is combined with a harmony search (HS) scheme. Within this hybrid framework, the CS is employed to find the promising regions of the search space within the initial explorative stages of the search, followed by a thorough exploitation phase using the combined CS and HS algorithms. The utility of the proposed CSHS is demonstrated using four water distribution system design problems with increased scales and complexity. The obtained results reveal that the CSHS method outperforms the standard CS, as well as the majority of other meta-heuristics that have previously been applied to the case studies investigated, in terms of efficiently seeking optimal solutions. Furthermore, the CSHS has two control parameters that need to be fine-tuned compared to many other algorithms, which is appealing for its practical application as an extensive parameter-calibration process is typically computationally very demanding.

Key words | cuckoo search algorithm, harmony search, optimization, water distribution systems

INTRODUCTION
Meta-heuristics have been widely used to handle water resources optimization problems over the past four decades (Maier et al. 2014). The relevant research is especially active in the water distribution system (WDS) design field as this problem type is typically difficult due to the large search space and the highly nonlinear constraints. Nowadays, a vast number of scientific works dealing with WDSs optimization are available in the literature (as, for example, Farmani et al. (2006), McClamong et al. (2013), etc.). The meta-heuristic optimization algorithms that have previously been applied to WDS design problems include genetic algorithms (GAs) (Simpson et al. 1994; Savic & Walters 1997), particle swarm optimization (PSO) (Suribabu & Neelakantan 2006), tabu search (TS) (Cunha & Ribeiro 2004), ant colony optimization (ACO) (Maier et al. 2005; Zecchin et al. 2006), simulated annealing (SA) (Cunha & Sousa 1999), differential evolution (DE) (Vasan & Simonovic 2010), harmony search (HS) (Geem et al. 2002), cross-entropy (Perelman & Ostfeld 2007), and charged system search (Sheikholeslami et al. 2014).

Despite the successes in finding optimal or near-optimal solutions, the practical applications of meta-heuristic algorithms are not without difficulties. One of the main issues is their slow convergence rate towards the optimum, resulting in high computational overheads that are typically beyond the availability for practical application (Maier et al. 2014). This is especially the case for the water resources problems as their simulation models used in the optimization process (function evaluation) are often computationally expensive.

doi: 10.2166/hydro.2015.174
A number of studies have been undertaken to improve the efficiency of the meta-heuristic algorithms applied to water resources problems. These studies can be divided into two main categories according to the means adopted to enhance the convergence speed.

The first category aims to reduce the computational time of the simulation model, which is often the most time-consuming part within the optimization process. By way of example, Broad et al. (2005) introduced a metamodeling approach to optimize a WDS design problem, in which the artificial neural networks (ANNs) were used to replace the hydraulic simulation model. More recently, Bi & Dandy (2014) also used ANNs to substitute the full hydraulic and water quality simulation model in their optimization framework, in order to expedite the convergence rate.

The second category attempts to reduce the computational overheads associated with the meta-heuristic algorithms by means of improving their searching efficiency. The improved efficiency is often achieved through the hybridization of the meta-heuristic algorithms and deterministic optimization techniques. For example, Cisty (2010) proposed a hybrid GA-linear programming (LP) method to seek the minimum-cost design of WDSs. In the GA-LP framework, a GA was used in the outer loop to decompose a complex looped network into a group of equivalent branched networks, followed by the implementation of a LP in the inner loop to optimize each branched network. Zheng et al. (2011) combined a nonlinear programming (NLP) and a DE algorithm for optimizing WDS design, where an NLP was used to approximate the optimal solutions, and a DE algorithm was subsequently employed to polish the identified approximate solutions. In a more recent study, Zheng et al. (2014) coupled a binary linear programming with a DE algorithm to improve the optimization efficiency of the WDS design problems.

In parallel with the development of the meta-heuristic-deterministic optimization models, research has also been carried out to hybridize multiple meta-heuristic algorithms for improving their searching ability. For instance, Keedwell & Khu (2005) proposed a hybrid method, in which a heuristic-based, local representative cellular automata approach was developed to provide a good initial population for GA runs. Geem (2009b) presented an improved HS method for the optimal design of WDSs by incorporating the searching mechanisms of the PSO. More recently, Sedki & Ouazar (2012) developed a hybrid PSO-DE model, taking advantage of the PSO’s strong local searching strength and DE’s great exploration ability.

The studies mentioned above have made great contributions in building knowledge for improving optimization efficiency of meta-heuristic algorithms. To further progress research in this field, this study aims to develop a new hybrid optimization framework in which a cuckoo search (CS) algorithm is combined with a HS method (denoted as CSHS). The utility of the proposed CSHS method in terms of tackling WDS design problems will be investigated in the present study. It is a well-established fact that the WDS optimization problem belongs to the class of non-deterministic polynomial-time hard problems known as NP-hard which means that for an N-pipe network, the computational time required for a rigorous algorithm is, at best, an exponential function of N and is thus enormous even for relatively small WDS. In particular, when large-scale problems are considered, meta-heuristics such as the proposed CSHS are one of the best alternatives that experts can often rely on, since exact algorithms take exponential time to find an optimal solution to NP-hard problems. Therefore, using a new meta-heuristic optimization technique is one of the interesting, if not the best, way of treating WDS optimization problems. Furthermore, the coupled CSHS–EPANET framework is a multi-purpose model and can be extended to tackle other water network management problems, which is appealing for its practical applications.

The CS algorithm is a relatively new optimization technique, which is designed according to the Lévy flight and brood parasitic behavior of some cuckoo species (Yang & Deb 2009). It has been proven to deliver good performance in dealing with a number of optimization problems, ranging from mathematical problem optimization, engineering design, neural network training, to the traveling salesman problem (Civicioglu & Besdok 2013; Gandomi et al. 2013; Yang & Deb 2014). Despite the great potential in efficiently finding optimal solutions for a range of problem types, the CS algorithm has received limited application in the WDS optimization field. To the authors’ knowledge, the only work is Wang et al. (2012), who combined CS with the non-dominated sorting GA (NSGA-II) to optimize WDS design problems with multiple objectives.
This study focuses on the application of the CS algorithm to the WDS design problems in the single-objective domain, and more importantly, attempts to improve its performance through the incorporation of the HS mechanism. The specific contributions of this study include: (i) the investigation of the CS’s ability in finding single-objective optimal solutions for WDS design problems; (ii) the proposal of a hybrid CSHS algorithm; and (iii) the demonstration of the CSHS’s utility in optimizing WDS design.

**CUCKOO SEARCH AND HARMONY SEARCH ALGORITHMS**

Since the proposed CSHS method involves the standard CS and HS algorithms, it is necessary to briefly outline the main mechanisms of both methods, as shown in the following sub-sections.

The cuckoo search algorithm

The CS algorithm is inspired by the brood parasitism of some cuckoo species by laying their eggs in the nests of other host birds (Yang & Deb 2009). In this algorithm, each egg in a nest represents a candidate solution, and a cuckoo egg indicates a new solution. The strategy is to use the new, and potentially better, solutions (cuckoos) to replace a not-so-good solution in the nests, following the three rules below (Yang & Deb 2009):

- **Rule 1** Each cuckoo lays one egg at a time, and dumps its egg in a randomly chosen nest, which corresponds to the initialization process of other algorithms such as GAs.
- **Rule 2** The best eggs (i.e., solutions) are contained in a fraction of the nests and will carry over to the next generation. This is equivalent to the elitism scheme used in the GAs so that the best solutions are passed on to the next generation.
- **Rule 3** The number of nests is fixed and a host bird can find an alien egg with a specified probability, \(p_a \in (0,1)\). In this case, the host bird can either throw the egg away or abandon the nest so as to build a completely new nest in a new location.

The pseudo-code of the standard CS algorithm is given in Figure 1.
An essential part of the CS algorithm is the use of random walk in both local and global search steps, as shown in Figure 1. In the global search step, the new solution at $t+1$ iteration is generated randomly by Lévy flights as follows:

$$X_i^{(t+1)} = X_i^{(t)} + \alpha \times \mathbf{L}_i^{(t)} \otimes \mathbf{rand}^{(t)} \otimes (X_i^{(t)} - X_{\text{global}}^{(t)})$$

(1)

where $\mathbf{L}_i^{(t)}$ is a random walk based on the Lévy flights, $\mathbf{rand}^{(t)}$ is a random vector whose elements are normally distributed with zero mean and a unit standard deviation; $X_{\text{global}}^{(t)}$ is the global best solution. The product $\otimes$ denotes element-wise multiplications (Hadamard product).

The Lévy flight (Pavlyukevich 2007) is a random walk that is characterized by a series of instantaneous jumps chosen from a probability density function with a power law tail (Walton et al. 2011). One of the most efficient and simple ways in applying Lévy flights is to use the so-called Mantegna algorithm (Mantegna 1994). According to the Mantegna algorithm, the $i,d$ element of $\mathbf{L}$ in Equation (1) is performed using the following equation:

$$L_{i,d} = \frac{u_{i,d}}{|v_{i,d}|^\beta} \quad (i = 1, 2, ..., N; d = 1, 2, ..., D)$$

(2)

where $D$ is the dimensionality of the search space, $\beta$ is a parameter in the range of $[1, 2]$ (here $\beta$ is considered to be 1.5), $u_{i,d}$ and $v_{i,d}$ are random numbers drawn from normal distribution $N(\mu, \sigma^2)$ with zero mean ($\mu = 0$) and the standard deviations ($\sigma$) are calculated by:

$$\sigma_u = \left[ \frac{\Gamma(\beta + 1) \sin \left( \frac{\pi \beta}{2} \right)}{\Gamma \left( \frac{\beta + 1}{2} \right) \beta^2} \right]^{1/2}, \quad \sigma_v = 1$$

(3)

where $\Gamma$ is the standard Gamma function.

On the other hand, in the local search step, the $p_a$ fraction of the worst solutions (nests) are discovered and replaced by new ones; therefore, the positions of the new solutions are generated by random walk as follows:

$$X_i^{(t+1)} = X_i^{(t)} + \mathbf{r}_1^{(t)} \otimes \mathbf{H}(p_a - r_2^{(t)}) \otimes (X_i^{(t)} - X_k^{(t)})$$

(4)

where $\mathbf{r}_1^{(t)}$ and $\mathbf{r}_2^{(t)}$ are random vectors uniformly distributed in the range of $(0,1)$, $\mathbf{H}$ is the Heaviside step function whose values are zero for negative argument and one for positive argument, $p_a$ is the discovering probability, and $X_i^{(t)}$ and $X_k^{(t)}$ are two different solutions selected randomly by random permutation.

After producing the new solution, $X_i^{(t+1)}$, it will be evaluated and compared to the $X_i^{(t)}$. If the fitness of the new solution is better than the objective fitness of the previous solution, $X_i^{(t+1)}$ is accepted as a new basic solution; otherwise, $X_i^{(t)}$ would be kept (Yang & Deb 2014).

The harmony search algorithm

The HS method, originated by Geem et al. (2001), is an optimization algorithm inspired by the working principles of musical harmony improvisation. Similar to other meta-heuristic approaches, HS is a random search technique. It does not require any prior domain knowledge such as gradient information of the objective function. However, unlike population-based approaches, it utilizes only a single search memory to evolve. Therefore, the HS method has the distinctive feature of algorithm simplicity (Geem 2010). The mainframe of the basic HS algorithm can be described as shown in Figure 2.

The usage of the harmony memory (HM) as shown in Figure 2 is important because it ensures that good harmonies are considered as elements of new solution vectors. The initial HM consists of a certain number of randomly generated solutions for the optimization problem. For a $D$-dimension problem, the elements of HM can be represented as $[x_{ij}]$ (i = 1,..., HMS; j = 1,..., D). The harmony memory size (HMS) is typically set to be between 10 and 100 (Geem 2009a). After generating an initial HM, each component of a new solution ($HM_{ij}^{(t)}$) is obtained based on the harmony memory considering rate (HMCR). The HMCR is defined as the probability of selecting a component from HM members, and 1–HMCR is, therefore, the probability of random generation. If a new solution comes from HM, it can be further mutated according to the pitch adjusting rate (PAR). The PAR determines the probability of a candidate from HM for mutation. In Figure 2, bw is a bandwidth, controlling the local range of pitch adjustment, which should be related to the scales of the problem of
interest. In most cases, we can use \( bw = O(L/10) \), where \( L \) is the characteristic scale of the problem of interest, while in some cases \( bw = O(L/100) \) can be more effective and avoid flying too far.

Finally, the new solution is evaluated and if it yields a better fitness than that of the worst member in the HM, it will be replaced instead of that one. Otherwise, it is eliminated. The generation of new solutions and memory consideration steps are alternatively performed until a termination criterion is satisfied.

## The Proposed Hybrid CSHS Algorithm

The standard CS algorithm is one of the most successful optimization techniques (Yang & Deb 2014). However, it has two main drawbacks, especially in the case of complex multi-modal problems. The first weakness is the slow convergence rate of the algorithm (Walton et al. 2011). In other words, given enough computation, the standard CS will always find the optimum solution, but as the search relies entirely on random walks a fast convergence cannot be guaranteed (Yang & Deb 2010). The second weakness of the algorithm is that, in the standard CS, there is no information exchange between individuals and, essentially, the searches are performed independently (Walton et al. 2011; Long & Jiao 2014). In order to overcome these limitations, the proposed hybrid CSHS algorithm combines the optimization capabilities of HS and CS. There are various hybrid models of HS with other meta-heuristics in the literature (Alia & Mandava 2014). The proposed hybrid CSHS integrates some HS components within the standard CS. In order to increase the exploitation capability of the algorithm so as to avoid being trapped into local optima and speeding up the convergence, the CSHS algorithm uses the memory (HM) and pitch adjusting operation from HS algorithm which serves as a mutation operator in the new hybrid method.

The proposed hybrid algorithm is a double stage optimization technique which employs the CS algorithm in the first stage and the HS algorithm in the second one. The first stage utilizes two search capabilities of the standard CS: local search and global search, controlled by a switching/discovery probability of CS \((p_d)\). Also, its global search

### Figure 2 | The pseudo-code of HS.

```plaintext
The HS algorithm

- Initialize the algorithm parameters:
  - Harmony memory size (HMS), pitch adjusting rate (PAR), harmony memory considering rate (HMCR), and bandwidth distance (bw)

- Initialize the Harmony Memory (HM) randomly, \( \text{HM}_i^{(0)} = [x_{i,0}, \ldots, x_{i,0}]^T \) (\( t = 0, i = 1, 2, \ldots, \text{HMS} \))

for \( i = 1 : \text{HMS} \)

  - Evaluate the fitness of each individual in HM

end for

while \( i < \text{MaxEva} \)

  for \( j = 1 : D \)

    - \( r_1 = \text{uniform random number between 0 and 1} \)
      - if \( r_1 < \text{HMCR} \) (memory consideration)
        - Choose a solution randomly (HM, \( i^{th} \)) among HMS existing harmony vectors
        - \( r_2 = \text{uniform random number between 0 and 1} \)
        - if \( r_2 < \text{PAR} \) (pitch adjustment)
          - \( r = \text{uniform random number between 0 and 1} \)
          - \( x_{ij}^{(t)} = x_{ij}^{(t)} + bw \times (2 \times r - 1) \)
        - end if
      - else (random selection)
        - Select a new value \( (x_{ij}^{(t)}) \) randomly from the possible value set (\( \Phi \))
      - end if

  end for

  Evaluate fitness of new solution, \( F(\text{HM}_i^{(t)}) \)

  if \( \text{HM}_i^{(t)} \) is better than existing harmony vectors then update HM

  \( i = i+1 \)

end while
```
uses Lévy flights, rather than standard random walks. In the second stage, pitch adjustment is carried out by tuning the solution within a given bandwidth. A small random amount is added to or subtracted from an existing solution stored in HM. Essentially, pitch adjustment is a refinement process of local solutions. The use of a HM in CSHS allows the selection of the best solutions that may belong to different regions in the search space. Since the standard CS algorithm is memory-less, there is no information stored dynamically during the search, while the hybrid CSHS uses a memory that contains some information extracted online during the search. In other words, the history of the search stored in a memory can be used for generating new solution candidates. Therefore, incorporating the HM and PAR into the CSHS algorithm ensures that good local solutions are retained while the randomization makes the algorithm able to globally explore the search space more effectively. Such interactions between global search and local search components could be an important factor for the success of the CSHS algorithm over other algorithms because it guarantees a balanced combination of the exploration and exploitation. In the proposed hybrid method, exploration is represented in the form of randomization with a random component attached to a deterministic component in order to explore the search space effectively and efficiently (e.g., Equation (1)), while exploitation is the utilization of past solutions so as to select the potentially good solutions (Equation (2)) via elitism (Rule 2 in the CS) or use of memory (HM) or both.

After explaining the main elements of the proposed hybrid algorithm, the framework of CSHS is illustrated as follows. Without losing the attractive features of the original CS and HS, the CSHS begins with random generation of the initial population (or N host nests). Then, the vectors of the HM are generated. After finding the best solution/nest ($X_{global}$) in the first stage, it will be included in HM if its fitness is better than the fitness of the existing harmony vectors. In the second stage, a new harmony vector (solution) is created based on the HS strategy. After memory updating, the best harmony vector ($HM_{best}$) is determined. Then, $HM_{best}$ vector substitutes the $X_{global}$ only if it has better fitness. This procedure is repeated to update the current best solution of the population in every iteration. Figure 3 shows the outline of the proposed CSHS method.

THE CSHS METHOD FOR THE OPTIMAL DESIGN OF WDS

Problem formulation

In this article, the WDS design is formulated as a least-cost single objective optimization problem with a selection of pipe sizes as the decision variables, while the network layout and its connectivity, nodal demand, and minimum head requirements are imposed as design constraints.

Here, a design $\Phi$ is defined as a set of $n_p$ decisions where $n_p$ is the number of pipes to be sized, that is $\Phi = \{D_1, D_2, \ldots, D_{n_p}\}$ where $D_i$ is the selected diameter for pipe $i$. The optimization problem can be stated mathematically as follows:

Minimize:

$$C(\Phi) = \sum_{i=1}^{n_p} l_i c_i$$  \hspace{1cm} (5)

Subject to:

$$H(\Phi) \geq h_{\text{min}}$$  
$$D_i \in \{D\}$$  \hspace{1cm} (6)

where $C(\Phi)$ is the network cost; $l_i$ is the pipe length; $c_i$ is the unit cost of pipe $i$; $H(\Phi)$ represents the performance constraints of the design solution $\Phi$, with $H(\Phi) \leq h_{\text{min}}$ showing that the design pressure head at each demand node is above (or equal to) its corresponding minimum allowable pressure head $h_{\text{min}}$; and $D$ is a set of commercially available pipe diameters. The determination of $H(\Phi)$ is normally performed by a hydraulic simulation model.

Constraint handling

In order to handle the minimum nodal head constraints, a penalty approach is utilized. If the nodal head exceeds the allowable limit, the penalty is zero; otherwise the amount of penalty is obtained by dividing the violation of the allowable limit to the limit itself. By solving the nonlinear energy and mass balance equations for flows and heads in the network (determination of $H(\Phi)$), the pressure of each node is obtained then these values are compared to the allowable...
limits to calculate the penalty functions as:

\[
\begin{cases}
H_j^{\text{min}} \leq H_j \Rightarrow \Delta_j = 0 \\
H_j^{\text{min}} > H_j \Rightarrow \Delta_j = \frac{H_j^{\text{min}} - H_j}{H_j^{\text{min}}} 
\end{cases}
\]  

(7)

where \(H_j\), \(H_j^{\text{min}}\), and \(\Delta_j\) are the pressure head, minimum required pressure head, and the amount of constraint violation at node \(j\), respectively. Using this approach, the optimization process is extended to include the constraints by introducing the cost function as:

\[
F_{\text{cost}} = \left(1 + \sum_{j=1}^{n_n} \Delta_j\right)^{\delta} \times C(\Phi)
\]  

(8)

where \(F_{\text{cost}}\) is the penalized cost of the network, \(\delta\) is the penalty exponent, and \(n_n\) is the total number of nodes in the network.

The static penalty function method with constant \(\delta\) has certain drawbacks, for example, trial-and-error process has to be performed to determine appropriate values for the penalty exponent. This repeated process is time-consuming and is even not allowed in some real-world applications. To cope with the above problems, in Equation (8), the dynamic penalty method is applied. Here, the exponent \(\delta\) is selected so that it increases the penalties as the search progresses. The value of the penalty function exponent is important because it governs the rate of increase in the cost of infeasible designs, which directly affects the exploration of the CSHS. In the first iterations, if \(\delta\) has a large value, the solutions tends to narrow the search space to designs that, while feasible, are more expensive than the optimal design and reduce the exploration of the solution space (Joines & Houck 1994). However, within the last iterations, if \(\delta\) has a small value, the solutions have an undesirable tendency to converge to least-cost, but infeasible, designs that have a very small penalty. Setting a
large value for $\delta$ in last iterations may help to prevent convergence to infeasible designs by increasing the applied penalty. Therefore, as the penalty increases it puts more and more selective pressure on the CSHS to find a feasible solution. Thus, in the first step of the search process, $\delta$ is set to 1.5 and ultimately increased to 2.5. (These two values were selected after a number of fine-tuning trials.)

**A combined simulation–optimization model**

Here the CSHS algorithm is coupled with the widely used water distribution network software, EPANET 2 (Rossman 2000). After analyzing a model by EPANET 2, the pressure of each node is obtained, then these values are compared to the allowable limits. EPANET programmer’s toolkit was provided by the United States Environmental Protection Agency that is a dynamic link library of functions which allows developers to customize EPANET’s computational engine for the user’s specific needs. Here, the algorithms were implemented in MATLAB and optimization runs were carried out on a computer with an Intel Core i5 CPU, 2.53 GHz processor, and 3.00 GB RAM. A brief description of the steps in the proposed algorithms for pipe network optimization is given below:

1. Generate $N$ population of nests randomly in the solution space. Each of the $N$ populations represents a possible combination of pipe indices.
2. Compute the network cost $C(\Phi)$ for each of the $N$ solutions after converting the randomly generated pipe indices to the pipe sizes available in the market.
3. Update the input file of the simulator (only the diameters are changed).
4. Perform hydraulic analysis of each network. EPANET 2 is used to analyze the network and check the pressure at some nodes which are required to meet certain nodal pressures.
5. Compute violations ($\Sigma \Delta$), if the nodal head at any node is less than the required minimum.
6. Calculate the total cost of the network ($F_{tot}$) using the network cost and the penalty found in steps 2 and 5, respectively.
7. The total cost found in step 6 is used as the fitness value for each of the trial networks.

**Parameter setting for CSHS**

The important parameters of the proposed algorithm are $\alpha$ (scaling factor), $p_a$ (discovering probability of alien eggs/solutions), $PAR$ (pitch adjusting rate), and $HMCR$ which control the local search and global search behavior of the algorithm. Due to the importance of the first two parameters, an extensive sensitive analysis for $\alpha$ and $p_a$ is performed for the first case study (Hanoi problem) in the next section. The other parameters belong to the HS stage of the algorithm and will be determined in this sub-section.

In order to use the HS mechanism effectively, the CSHS algorithm adopts the parameters $PAR$ and $HMCR \in (0,1)$. If $HMCR$ is too low, only a few elite solutions are selected, and the algorithm may converge too slowly. If this rate is extremely high, the solutions in the HM are mostly used, and other ones are not explored well, so that good regions may be missed. Furthermore, small $PAR$ values can cause poor performance of the algorithm and a considerable increase in required iterations. However, large $PAR$ values usually cause the improvement of best solutions in final generations when the algorithm converges to optimal solution vector. To solve this problem, inspired by the SGHS (a self-adaptive global best HS) algorithm (Pan et al. 2010), a self-adaptive strategy is presented in this section. Unlike the standard HS algorithm, this technique employs an adaptive parameter tuning method. By using this strategy $HMCR$ and $PAR$ become the configurable components of the CSHS and can be adjusted to alter the performance of the algorithm. Table 1 lists the parameters (that must be set before algorithm execution) required for major types of meta-heuristics including GA, PSO, ACO, HS, SA, and CSHS.

In this method, the main assumption is that the $HMCR$ (or $PAR$) value is normally distributed in the range of $[0.8, 0.99]$ ([0.01, 0.5]) with mean $HMCR_m (PAR_m)$ and standard deviation of 0.01 (0.05) (Pan et al. 2010). Initially, $HMCR_m (PAR_m)$ is set at 0.85 (0.25) (Poursalehi et al. 2013); and then the algorithm starts with a $HMCR$ ($PAR$) value generated according to the normal distribution. During the evolution process, the $HMCR$ ($PAR$) value of a generated solution that has successfully replaced the worst member in the HM is recorded. After a specified number of generations (known as the learning period), $HMCR_m (PAR_m)$ are re-computed by averaging all the recorded good
HMCR (PAR) values during this period. With the new mean and the given standard deviation of 0.01 (0.05), a new HMCR (PAR) value is produced and used in the subsequent iterations. The above procedure is repeated, and so an appropriate HMCR (PAR) value can be gradually learned to suit the specific problem within the specific phases of the search process. In this way, the difficulty of finding suitable values for these parameters is solved and a self-adaptation operator adjusts the HMCR and PAR values automatically. According to previous studies, the choice of the start points for HMCR and PAR does not affect the performance of the algorithm (Pan et al. 2010; Kulluk et al. 2011; Poursalehi et al. 2013).

**CASE STUDY RESULTS AND DISCUSSION**

In this section, four well-known benchmark networks are optimized using the present method. The final results are compared to the solutions of other advanced meta-heuristic methods to demonstrate the performance of the CSHS method. The first network (N1) is the Hanoi network, first proposed by Fujiwara & Khang (1990), which consists of 32 nodes, 34 pipes, and 3 loops. This network has no pumping station as it is fed by gravity from a reservoir with a 100 m fixed head. The minimum head requirement of the other nodes is 30 m and the Hazen–Williams coefficient for each new pipe is 130. The second design example (N2), first introduced by Cisty (2010), is the double Hanoi network. Since this network is derived from the basic Hanoi network, its optimal cost is known. All the parameters for the reservoir, nodes, and lines in the double Hanoi water distribution network are the same as in the original Hanoi network. The third test problem is the Balerma irrigation network (N3) which was presented for the first time by Reca & Martínez (2006). This network has a total of 443 demand nodes supplied by four source nodes. There are 454 pipes, arranged in eight loops. The minimum required pressure head is 20 m for each node. The last problem (N4) was taken from a town in the southeast of China which has 237 pipes, one reservoir, and 192 demand nodes. The head provided by the reservoir is 65 m. The minimum pressure requirement for each demand node is 18 m. This network was originally presented by Zheng et al. (2013b). Here, the optimization process is terminated after a fixed number of function evaluations, which is 60,000 for N1, 90,000 for N2, and 5,000,000 for N3 and N4 case studies.

**Population size study**

Figure 4 shows the results of the proposed CSHS method applied to three case studies with different population sizes (N) and harmony memory size (HMS). Several CSHS runs with different initial random numbers were performed for each case study to allow a reliable comparison. As demonstrated in Figure 4, in terms of the best and average cost solutions based on 10 runs with different starting random number seeds, the CSHS with a larger N and HMS performed better for each case study. However, the number of evaluations required to find optimal solutions with a larger N and HMS are increased remarkably. Considering both the solution quality and computational efficiency, the CSHS obtained good results when \( N = 30, \) \( HMS = 15 \), \( N = 300, \) \( HMS = 150 \), and \( N = 200, \) \( HMS = 100 \) were selected for the N1, N3, and N4 case studies, respectively. The best parameters obtained for the Hanoi network was used for the double Hanoi problem. By comparing the numerical investigations (Figure 4) and the selected \( \{ N, HMS \} \) for each case study, one approximate heuristic
A hybrid cuckoo–harmony search algorithm for optimal design of WDSs

Guideline for practical implementations can be $HMS = \frac{N}{2}$ for the CSHS algorithm applied to a WDS optimization problem.

Case study 1

The configuration of the water distribution network in Hanoi ($N_1$) is shown in Figure 5. The cost of commercially available pipe sizes {12, 16, 20, 24, 30, 40, in inches} is {45.73, 70.40, 98.38, 129.30, 180.80, 278.30, in dollar/meter}. The combinations of such a 34 pipe network system results in $6^{34} = 2.86 \times 10^{26}$ possible designs. Details of this network and the system data are given in Fujiwara & Khang (1990).

Table 2 reports the statistical results and the required number of evaluations to find the best cost by the present algorithm and some selected meta-heuristic algorithms from the literature. The proposed CSHS found the best feasible solution of $6.0811 \times 10^6$ after 31,800 function evaluations. As shown in Table 2, the best convergence speed belongs to MBA and IMBA algorithms with 22,450 and 16,400 function evaluations, respectively; however, the average and maximum costs obtained by these methods were higher than that produced by the CSHS. Comparison with other advanced and hybrid algorithms such as HD-DDS, PSO-DE, and NLP-DE demonstrates the effectiveness and efficiency of the proposed method in solving this problem.

Using the CSHS model, the average and maximum cost obtained from 200 runs are $6.1075 \times 10^6$ and $6.1598 \times 10^6$, respectively; whereas in the standard CS model, the obtained maximum cost is $6.2237 \times 10^6$ with an average of $6.1953 \times 10^6$. A convergence comparison between the proposed CSHS and standard CS algorithms is illustrated in Figure 6. As clearly shown in this figure, the CSHS algorithm was able to converge faster than the standard CS algorithm in terms of finding the best solution as well as the best average cost. The better performance of the CSHS for this case study could be attributed to its greater ability to explore efficiently the search space with the aid of HS operators and thus enhancing the chances of finding the global optimum with fewer function evaluations.
A sensitivity analysis is performed for the two important parameters ($\alpha$ and $p_a$) of the CSHS algorithm. In order to avoid the possible randomness of the search process due to the use of different initial solutions, the Hanoi problem is solved 20 times for different parameter configurations. It is worth mentioning that each major type of meta-heuristic algorithm has a number of parameters that must be fine-tuned before algorithm execution for different problems. The best parameters can be different depending on the problem size and complexity; however, due to the large computational requirements, it is impractical to test all possible parameter combinations. Therefore, a comprehensive set of experiments is conducted to identify the best parameters for the CSHS algorithm.
time the sensitivity analysis of the other networks has not been carried out and the best parameters' configuration obtained for the Hanoi network was used for the others. For the range of values of $\alpha = \{0.01, 0.02, 0.04, 0.06, 0.08, 0.10\}$ and $p_a = \{0.15, 0.25, 0.35, 0.45, 0.50, 0.75\}$, the N1 case study is solved and the statistical results are shown in Figures 7 and 8.

Figure 6 | Convergence properties of the CSHS and the standard CS for the N1 case study obtained from 200 runs.

Figure 7 | Results from various scaling factor ($\alpha$) and discovering probability ($p_a$) values (statistics are based on 20 runs).
sensitivity analysis was performed while fixing other parameters \((N = 30, \text{HMS} = 15)\) and the maximum number of evaluations was set to 60,000.

In Equation (1), the scale of the random search is controlled by multiplying the generated Lévy flight by a scaling factor \((\alpha)\). Thus, a small \(\alpha\) may slow down the convergence of CSHS because of the limitation in the exploration of only a small subspace of the whole search space. On the other hand, a large \(\alpha\) may cause the solution to scatter around the optima as in a random search. From Figures 7 and 8, it can be concluded that while (in most cases) \(\alpha = 0.06\), it makes the algorithm more effective; otherwise, when \(\alpha > 0.06\), new solutions are created outside of the design domain and thus waste evaluations, and while \(\alpha < 0.06\), considerable increase in iterations are needed to find the optimum solution (Figure 8).

According to Equation (4), the local search is very intensive with about \(p_a\) of the search time, while global search takes about \(1-p_a\) of the total search time. When \(p_a = 0.25\), about 75\% of the total search time is spent on the global searching which allows the search space to be explored more efficiently on the global scale, and consequently the global optimality can be found with a higher probability (Figure 7). As a result, \(p_a = 0.25\) and \(\alpha = 0.06\) are suitable values for the CSHS algorithm.

### Case study 2

For the double Hanoi network, all the properties for the reservoir, nodes, and pipes are the same as the original Hanoi network on both mirrored sub-networks with the addition of the first pipe (from the reservoir to node 2), which is shortened from 100 to 28.9 m. This change was made for the sake of obtaining the same head in node 2 (with a diameter of 40 in) as in the original Hanoi network. The total solution space is then equal to \(6^{67} = 1.37 \times 10^{52}\). Network layout for this problem (\(\text{N}_2\)) is shown in Figure 9. For double Hanoi network the reference optimal solution for different algorithms can be calculated as follows:

\[
C_{DH} = 2C_H - 2l_1c_1 + 28.9c_1
\]

where \(C_{DH}\) is the reference global optimum cost of the double Hanoi network; \(C_H\) is the optimal cost of the Hanoi network; \(l_1\) is the length of the first pipe on the original network (100 m); and \(c_1\) is the unit price of diameter 40 in ($278.28) (Cisty 2010).

For our solution described in the previous example ($6.081 \times 10^6$), the reference global optimum solution \((C_{DH})\) of the double Hanoi network should be $12.114 \times 10^6$. 
The best results obtained by the CSHS, standard CS, BB–BC (Tahershamsi et al. 2012), GA (Cisty 2002), OptiDesigner (Cisty 2002), and HS (Geem et al. 2002) are summarized in Table 3. Using the optimum cost of the Hanoi network ($C_H$), the reference global optimum solutions of the double Hanoi network can be calculated as $12.114 \times 10^6$ (for CS, GA, and HS), $12.182 \times 10^6$ (for OptiDesigner), and $12.400 \times 10^6$ (for BB–BC), respectively.

The CSHS found the best feasible solution of $12.347 \times 10^6$ after 81,900 evaluations. Therefore, deviation from reference optimal solution for the CSHS algorithm is 1.92%. As can be seen from Table 3, comparison with other algorithms such as CS, GA, and HS reveals that the CSHS algorithm is better in term of closeness to the reference global minimum. In other words, the CSHS algorithm is less likely to be trapped by local optimal solutions.

### Table 3 | Performance comparison for the N$_2$ case study

<table>
<thead>
<tr>
<th>Method</th>
<th>Hanoi network cost ($)</th>
<th>Double Hanoi network cost ($)</th>
<th>Deviation from reference global optimum (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard CS</td>
<td>$6.081 \times 10^6$</td>
<td>$12,871,876$</td>
<td>6.23</td>
</tr>
<tr>
<td>OptiDesigner (Cisty 2002)</td>
<td>$6.115 \times 10^6$</td>
<td>$12,795,541$</td>
<td>5.04</td>
</tr>
<tr>
<td>GA (Cisty 2002)</td>
<td>$6.081 \times 10^6$</td>
<td>$12,600,624$</td>
<td>4.01</td>
</tr>
<tr>
<td>HS (Geem et al. 2002)</td>
<td>$6.081 \times 10^6$</td>
<td>$12,404,680$</td>
<td>2.39</td>
</tr>
<tr>
<td>BB–BC (Tahershamsi et al. 2012)</td>
<td>$6.224 \times 10^6$</td>
<td>$12,647,789$</td>
<td>2.00</td>
</tr>
<tr>
<td>CSHS</td>
<td>$6.081 \times 10^6$</td>
<td>$12,346,537$</td>
<td>1.92</td>
</tr>
</tbody>
</table>

The reference global optimum solutions of the double Hanoi network can be calculated as $12.114 \times 10^6$ (for CS, GA, and HS), $12.182 \times 10^6$ (for OptiDesigner), and $12.400 \times 10^6$ (for BB–BC), respectively.

The CSHS found the best feasible solution of $12.347 \times 10^6$ after 81,900 evaluations. Therefore, deviation from reference optimal solution for the CSHS algorithm is 1.92%. As can be seen from Table 3, comparison with other algorithms such as CS, GA, and HS reveals that the CSHS algorithm is better in term of closeness to the reference global minimum. In other words, the CSHS algorithm is less likely to be trapped by local optimal solutions.

### Case study 3

Balerma is a water distribution network in the Sol-Poniente County in Almeria Province, Spain (Figure 10). There are
454 pipes to be designed using a set of 10 PVC pipes with diameters between 125 and 600 mm, and an absolute roughness coefficient of $k = 0.0025$ mm (Reca & Martinez 2006). In this case study the total enumeration number reaches the impressive amount of $10^{54}$. Also the Darcy–Weisbach equation has been adapted to calculate the head losses, using EPANET 2.

Table 4 summarizes and compares statistical results and required number of evaluations for convergence of the proposed algorithm and other selected meta-heuristics. As shown in Table 4, the best solution found by the CSHS algorithm for the N3 case study was €$1.988 \times 10^6$, which is 3.38% higher than the currently best known solution (€$1.923 \times 10^6$) reported by Zheng et al. (2011) using an NLP-DE method, but lower than solutions obtained by GA, HS, and GHEST.

The number of evaluations required for the CSHS algorithm to first reach the optimal solutions was 3 million,
which is less than those required by GA and HS in Table 4. While the best convergence speed belongs to GHEST with 290,500 function evaluations, the fitness value obtained by this method was higher than that produced by the CSHS. Also the HD–DDS yielded the best solution requiring 30 million evaluations. In addition, the minimum feasible solution obtained by the standard CS model is €2.036 × 10^6 (spending 4.5 million evaluations) and an average is €2.079 × 10^6 which is 2.41 and 2.36% higher than the best solution and average cost of CSHS, respectively. A convergence comparison between the proposed CSHS and standard CS algorithms is illustrated in Figure 11.

In order to guarantee equal conditions for comparison of the performance of different algorithms, an approach is to carry out the same number of evaluations of the fitness function. The maximum number of evaluations, \( \text{Eval}_{\text{max}} \), should depend on the network complexity which is a function of the number of links \( n_p \) and the number of commercially available pipe diameters \( m \). Baños et al. (2010) proposed the following equation to set the number of evaluations as:

\[
\text{Eval}_{\text{max}} = K \times n_p \times \log_{10}(m) \tag{9}
\]

If we set \( K = 1,000 \), \( n_p = 454 \), and \( m = 10 \), the resulting number of function evaluations is 454,000 for the N3 network (Baños et al. 2010). The efficiency of the CSHS in solving this large-scale and real-world design example is confirmed by Table 5.

As shown in Table 5, the CSHS algorithm is superior to the original GA, SA, HS, and CS algorithms in terms of finding the best solution in the same computational overhead. Considering the solution quality, comparison with other hybrid algorithms such as PSHS and MSATS (mixed SA TS) demonstrates the effectiveness of the proposed method in solving this problem.

Table 4 | Performance comparison for the N3 case study

<table>
<thead>
<tr>
<th>Method</th>
<th>Best cost (€10^6)</th>
<th>Average cost (€10^6)</th>
<th>Worst cost (€10^6)</th>
<th>No. of evaluations</th>
<th>No. of different runs</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA (Reca &amp; Martinez 2006)</td>
<td>2.302</td>
<td>2.334</td>
<td>2.350</td>
<td>10 × 10^6</td>
<td>10</td>
</tr>
<tr>
<td>HS (Geem 2009b)</td>
<td>2.018</td>
<td>NA</td>
<td>NA</td>
<td>10 × 10^6</td>
<td>NA</td>
</tr>
<tr>
<td>MBA (Sadollah et al. 2014)</td>
<td>2.211</td>
<td>2.384</td>
<td>2.433</td>
<td>45,400</td>
<td>10</td>
</tr>
<tr>
<td>IMBA (Sadollah et al. 2014)</td>
<td>2.064</td>
<td>2.202</td>
<td>2.338</td>
<td>45,400</td>
<td>10</td>
</tr>
<tr>
<td>GHEST (Bolognesi et al. 2010)</td>
<td>2.002</td>
<td>2.055</td>
<td>NA</td>
<td>290,500</td>
<td>10</td>
</tr>
<tr>
<td>HD–DDS (Tolson et al. 2009)</td>
<td>1.941</td>
<td>NA</td>
<td>NA</td>
<td>30 × 10^6</td>
<td>10</td>
</tr>
<tr>
<td>SADE (Zheng et al. 2013a)</td>
<td>1.983</td>
<td>1.995</td>
<td>NA</td>
<td>1.3 × 10^6</td>
<td>10</td>
</tr>
<tr>
<td>NLP–DEa (Zheng et al. 2011)</td>
<td>1.923</td>
<td>1.927</td>
<td>1.934</td>
<td>2 × 10^6</td>
<td>10</td>
</tr>
<tr>
<td>Standard CS</td>
<td>2.036</td>
<td>2.079</td>
<td>2.205</td>
<td>4.5 × 10^6</td>
<td>10</td>
</tr>
<tr>
<td>CSHS</td>
<td>1.988</td>
<td>2.031</td>
<td>2.158</td>
<td>3 × 10^6</td>
<td>10</td>
</tr>
</tbody>
</table>

*The best known solution to the Balerma network.

Table 5 | Performance comparison for the N3 case study with the same number of function evaluations

<table>
<thead>
<tr>
<th>Method</th>
<th>Cost (€10^6)</th>
<th>Fixed no. of evaluations</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA (Baños et al. 2010)</td>
<td>3.555</td>
<td>454,000</td>
</tr>
<tr>
<td>SA (Baños et al. 2010)</td>
<td>3.476</td>
<td>454,000</td>
</tr>
<tr>
<td>MSATS (Reca et al. 2008)</td>
<td>3.298</td>
<td>454,000</td>
</tr>
<tr>
<td>PSHS (Geem 2009b)</td>
<td>2.633</td>
<td>454,000</td>
</tr>
<tr>
<td>HS (Geem 2009b)</td>
<td>2.601</td>
<td>454,000</td>
</tr>
<tr>
<td>GHEST (Bolognesi et al. 2010)</td>
<td>2.178</td>
<td>454,000</td>
</tr>
<tr>
<td>Standard CS</td>
<td>2.614</td>
<td>454,000</td>
</tr>
<tr>
<td>CSHS</td>
<td>2.599</td>
<td>454,000</td>
</tr>
</tbody>
</table>

Figure 11 | Convergence properties of the CSHS and the standard CS for the N3 case study obtained from 10 runs.
Case study 4

The configuration of the last case study (N4), taken from a town in the southeast of China, is shown in Figure 12. A total of 14 pipes ranging from 150 to 1,000 mm are used for this network and the Hazen-Williams coefficient for each pipe is 150 and the total solution space is equal to $14^{237} = 4.29 \times 10^{271}$. The commercially available pipe sizes and their unit cost were provided by Kadu et al. (2008). This case study was first investigated by Zheng et al. (2013b). They developed a novel decomposition-based approach (DBA) using graph theory in which the DE algorithm is employed for optimization process as a meta-heuristic algorithm. Also, they applied the standard DE and GA algorithms with tuned parameters to this case study without network decomposition.

The statistics of the results for the last case study are given in Table 6. These include the best and average cost of solutions, and the number of evaluations required to find the best cost solution. For comparison, Table 6 also lists the results of other optimization techniques that have previously been used to optimize this case study including GA, DE, and DBA.

It is seen from Table 6 that the proposed CSHS gives a best design cost of $11.766 \times 10^6$ which is 5.86% lower than the best design cost obtained by the standard CS. Furthermore, the number of evaluations performed using the CSHS algorithm is only 3 million which is less than those of the GA (i.e., 4,546,000), SDE (i.e., 4,730,200), DBA (i.e., 3,215,685), and standard CS (i.e., 4,000,000) algorithms. In addition, the average cost solution generated by the proposed method is $11.80$ million, which is 2.37 and 3.56% higher than the average cost solution of the DE and DBA while 1.02 and 6.27% lower than the average cost solutions of the GA and standard CS, respectively. Thus, for this problem the best cost optimized by CSHS is consistent with the literature but the average cost is slightly high.
The performance of the CSHS is demonstrated using four well-known WDS case studies and the results are compared to that of standard CS and previously applied optimization methods. For almost all case studies, CSHS is shown to outperform standard CS both in terms of the ability to find the minimum solution and computational efficiency.

The current best-known solution for the N1 case study is \(6.081 \times 10^6\) which was found using the GENOME by Reca & Martinez (2006). This solution has also been found by the proposed CSHS method. As can be observed from the results of the N1 case study (Table 2), CS and CSHS exhibit similar performance in terms of finding the best-known solution, while CSHS was found to show better performance in terms of computational efficiency in comparison with other standard and hybrid algorithms (e.g., DE, GHEST, NLP-DE, and PSO-DE). It should be noted that the best convergence speed belongs to MBA and IMBA algorithms (Sadollah et al. 2014); however, the average and worst costs obtained by these algorithms were slightly higher than that obtained by the proposed CSHS. Moreover, in order to determine the ability of finding solutions close to the global minimum, the proposed hybrid method has been tested on the N2 case study. As can be seen from Table 3, deviation of the CSHS from global optimal solution is 1.92% while it is 6.23% for the original CS which indicates the greater ability of the CSHS to effectively explore the search space by incorporating the HS operators. Finally, the CSHS performance can be valuable in the large-scale optimization problems as evidenced by results on the optimization of the last two real-world and complex WDSs (N3 and N4 case studies), where the new algorithm is clearly competitive with the other advanced optimization methods. Our results (Table 5) confirm that in solving the N3 benchmark problem, with a similar computational budget, CSHS performs better than almost all algorithms and for the N4 benchmark problem the best cost obtained by CSHS is consistent with the literature but the average cost is slightly high (Table 6). One of the main challenges that exists in the use of meta-heuristics is that these algorithms are not adequately tested and no solid conclusions can be drawn on their searching behavior. Therefore, the proposed hybrid method would need to be further examined in order to determine its behavioral characteristics, including (1) the effectiveness of the search in finding optimal or near-optimal solutions,

<table>
<thead>
<tr>
<th>Method</th>
<th>Best cost ($10^5$)</th>
<th>Average cost ($10^5$)</th>
<th>No. of evaluations</th>
<th>No. of different runs</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA\textsuperscript{a}</td>
<td>11.85</td>
<td>11.92</td>
<td>4,654,000</td>
<td>10</td>
</tr>
<tr>
<td>DE\textsuperscript{a}</td>
<td>11.45</td>
<td>11.52</td>
<td>4,730,200</td>
<td>10</td>
</tr>
<tr>
<td>DBA\textsuperscript{a}</td>
<td>11.37</td>
<td>11.38</td>
<td>3,215,685</td>
<td>10</td>
</tr>
<tr>
<td>Standard CS</td>
<td>12.46</td>
<td>12.54</td>
<td>4,000,000</td>
<td>10</td>
</tr>
<tr>
<td>CSHS</td>
<td>11.77</td>
<td>11.80</td>
<td>3,000,000</td>
<td>10</td>
</tr>
</tbody>
</table>

\textsuperscript{a}Zheng et al. (2013b).

**CONCLUSION**

In this paper, the use and efficiency of the CS algorithm are investigated in the context of WDSs optimization. It is found that a standard CS algorithm is sometimes unable to produce acceptable solutions to WDS optimization problems. To improve the performance of the standard CS, a new hybrid algorithm, namely CSHS, based on the combined concepts of the CS and HS, is proposed to solve design problems of WDSs. The detailed implementation procedure of this hybrid meta-heuristic is also presented. By incorporating HS concept into standard CS, it is intended to enhance the efficiency and global convergence behavior of the standard CS.

For the CSHS algorithm, the improvements include utilizing the memory (HM) that contains information extracted online during a search and the addition of the pitch adjusting operation of HS during evolution process. Here, the HS strategy is used for fine tuning of the best solution obtained by a standard CS algorithm. In fact, the HM vectors become the CS population and then the evolving process is performed as the usual CS procedure. Obviously, the HM is a pool of the elite solutions and plays a key role in the algorithm. Another improvement is applying a self-adaptive strategy for the dynamically adaption of the control parameters (i.e., HMCR and PAR) using a learning mechanism. These unique features work in a combined framework and can ensure the efficiency of the proposed algorithm. Also, a sensitivity analysis is performed for the CSHS algorithm parameters where the population size, scaling factor (\(\alpha\)), and discovering probability of alien eggs (\(p_a\)) are concerned.
(2) convergence properties, and (3) the amount of the search effort spent in the feasible and infeasible regions of the solution space.

ACKNOWLEDGEMENTS

The first author gratefully acknowledges the facilities provided by the Global Institute for Water Security (GIWS), University of Saskatchewan, to carry out this research project.

REFERENCES


McClymont, K., Keedwell, E., Savic, D. & Randall-Smith, M. 2013 A general multi-objective hyper-heuristic for water
distribution network design with discolouration risk. 


First received 15 August 2015; accepted in revised form 19 October 2015. Available online 27 November 2015