24

Whale Optimization Algorithm – Modifications and Applications

Ali R. Kashani
Department of Civil Engineering
University of Memphis, Memphis, Tennessee, United States

Charles V. Camp
Department of Civil Engineering
University of Memphis, Memphis, Tennessee, United States

Moein Armanfar
Department of Civil Engineering
Arak University, Arak, Iran

Adam Slowik
Department of Electronics and Computer Science
Koszalin University of Technology, Koszalin, Poland

CONTENTS

24.1 Introduction

The whale optimization algorithm (WOA) algorithm as a nature-inspired metaheuristic optimization algorithm imitates the social behavior of humpback whales [1]. WOA is based on a mathematical model of their hunting behaviors. To be more exact, humpback whales utilize an exploration procedure called the bubble-net feeding method. Humpback whales desire to hunt a school of krill or small fishes near the surface. This foraging is accomplished by creating distinctive bubbles along a circle or '9'-shaped path. To accomplish this task, whales utilize two different tactics: upward-spirals and double-loops. The main procedure of a WOA can be summarized in three main steps: encircling prey; bubble-net attacking method; and search for prey. The first feature, encircling the prey, pushes all the search agents to move toward the best-found solution (leader). Next, the bubble-net attacking method simulates whales' movement path to approach the prey. To be more precise, whales move on a shrinking circular route along a spiral-shape path. This behavior provides the exploitation ability in the WOA. Another important characteristic of a metaheuristic algorithm is exploration; WOA uses search for prey to explore the search space. In this step, the position of the i-th search agent will be updated using one of the randomly selected search agents. The remainder of this chapter is organized accordingly. In Section 24.2 a summary of the fundamental of WOA algorithm is presented. In Section 24.3 some modifications of WOA are described. Finally, in Section 24.4 the application of a WOA to the optimization of shallow foundation is discussed.

24.2 Original WOA algorithm in brief

In this section, the fundamentals of the original WOA are discussed using a step-by-step pseudo-code. As presented in Algorithm 31, the WOA conducts an optimization procedure for a D-dimensional objective function $OF(.)$. In step 2, the boundary limitations for the j-th variable where $j = \{1, ..., number\ of\ decision\ variables\}$ is defined for the i-th agent as $[X_{i,j}^{min}, X_{i,j}^{max}]$. Next, parameters for the algorithm are defined, such as number of search agents ($SearchAgents\_no$) and maximum number of iterations (MI). In the next step, a population of $SearchAgents\_no$ search agents is generated randomly. Each agent is defined by a D-dimensional vector accordingly:

$$X_i = \{x_{i,1}, x_{i,2}, ..., x_{i,D}\} \tag{24.1}$$

where $x_{i,1}$ to $x_{i,D}$ are decision variables varying between $X_{i,j}^{min}$ and $X_{i,j}^{max}$.

As a fundamental in a WOA, search agents try to search the space around a leader (agent representing the current best solution). There are two strategies for selecting the leader: the overall best solution (for exploitation); and a randomly selected agent (for exploration). Therefore, in step 5, the global best solution and its associated X_i vector are attributed to the leader. After that, the current iteration counter is initialized as zero. Now, the main loop of the WOA is started. In step 9, each i-th search agent position is updated through an iterative loop where the value of α is updated following a decreasing pattern. Next, two coefficient vectors, A and C, are updated in steps 11 and 12, respectively. An important feature of the hunting behavior of humpback whales is the way they get close to the prey. Humpback whales move along a shrinking circle (steps 16 and 19) and spiral-shaped path (step 22), simultaneously. In the WOA, one of these movements is chosen based on a probability of 50%. In step 13, a random number p is generated between 0 and 1 and if $p < 0.5$ the search for global optimum solution uses a shrinking circle; otherwise, the spiral-shaped path strategy is applied. If the shrinking circle is selected, and $|A| < 1$ then the best-found solution is chosen as a leader and used to update position; otherwise, an agent is selected randomly and used as the leader. In step 22, if $p > 0.5$ the position of the i-th agent is updated following a spiral-shaped path. In step 25, estimate the $OF(.)$ for all the search agents. Next, the overall global best solution is updated. Similar to other optimization algorithms, the best-found solution and its associated X_i vector are proposed as the final result when the termination criteria are satisfied.

Algorithm 31 Pseudo-code of the original WOA.

1: determine the $D-th$ dimensional objective function $OF(.)$
2: determine the range of variability for each $j-th$ dimension $\left[X_{i,j}^{min}, X_{i,j}^{max}\right]$
3: determine the WOA algorithm parameter values such as $SearchAgents\_no$ – number of search agents, MI – maximum iteration
4: randomly create positions X_i for $SearchAgentsA\_no$ (each agent is a D-dimensional vector)
5: find the best search agent and call it leader
6: $Iter = 0$
7: **while** termination condition not met (here is reaching MI) **do**
8: **for** each i-th search agent **do**
9: update α value to decrease from 2 to 0 using $\alpha = 2 - Iter \times \left(\frac{2}{MI}\right)$
10: $A = 2\alpha \times rand - \alpha$
11: $C = 2 \times rand$
12: determine p as a random number between 0 and 1
13: **if** p<0.5 **then**
14: **if** | A |<1 **then**
15: update the position of i-th agent using: $X\left(Iter\right) = X^*\left(Iter-1\right) - A \cdot \mid C \cdot X^*\left(Iter-1\right) - X\left(Iter-1\right) \mid$
16: **else**
17: randomly select one of the search agents as a leader

update the position of the i-th agent using: $X(Iter) = X_{rand} - A \cdot \mid C \cdot X_{rand} - X(Iter-1) \mid$
end if
else
update the position of i-th agent using $X(Iter) = D^{'} \cdot exp(bl) \cdot \cos(2\pi l) + X^{*}(Iter-1)$
end if
end for
calculate $OF(.)$
update best-found solution
Iter=Iter+1
end while
post-processing the results

24.3 Modifications of WOA algorithm

24.3.1 Chaotic WOA

In a study by Kaur and Arora [2] a chaotic version of the WOA (CWOA) was developed to improve its performance and convergence speed. Like many other metaheuristic algorithms, there are some stochastic parameters in a WOA which affect the algorithm's performance. It has been proved using chaotic maps to control the randomness of those components can be considerably advantageous. Kaur and Arora [2] proposed to use 10 different chaotic maps to control parameter p in line 13 of the Algorithm 31. Based on this approach, an initial random number is selected for the chaotic map. Afterward, the value of p will be updated in each generation via a given chaotic map. Table 24.1 lists the chaotic maps utilized by Kaur and Arora [2].

In Table 24.1, a is a control parameter and CP_{i+1} is a pseudo random produced value using chaotic maps. CP_i varies between 0 and 1 where in the CWOA an initial point of 0.7 has been chosen.

In this effort, 20 well-known benchmark functions were studied to assess the effectiveness of the proposed modification. Results illustrated that the convergence rate of WOA has been improved by applying chaos maps. Statistical testing and convergence histories of modified WOA by chaotic maps determined superiority of chaotic WOA over the original WOA. Kaur and Arora [2] demonstrated that the Tent map was the best chaos mapping strategy.

24.3.2 Levy-flight WOA

Metaheuristic optimization algorithms explore the solution space based on exploration and exploitation. Incorporating a Levy-flight distribution to algorithms has been proved to be effective in this regard. Abdel-Basset et al. [3]

TABLE 24.1
Different chaos maps.

ID	Name	Formulation
1.	Logistic	$CP_{i+1} = aCP_i\left(1 - CP_i\right)$
2.	Cubic	$CP_{i+1} = aCP_i\left(1 - CP_i^2\right)$
3.	Sine	$CP_{i+1} = \frac{a}{4} sin\left(\pi CP_i\right)$
4.	Sinusoidal	$CP_{i+1} = aCP_i^2 sin\left(\pi CP_i\right)$
5.	Singer	$CP_{i+1} = \mu(7.86CP_i - 23.31CP_i^2 + 28.75CP_i^3 - 13.302875CP_i^4), \mu = 1.07$
6.	Circle	$CP_{i+1} = mod\left(CP_i + b - \left(\frac{a}{2\pi}\right) sin\left(2\pi CP_i\right), 1\right)$ $a = 0.5$ and $b = 0.2$
7.	Iterative	$CP_{i+1} = sin\left(\frac{a\pi}{CP_i}\right)$
8.	Tent	$CP_{i+1} = \begin{cases} \frac{CP_i}{0.7} & CP_i < 0.7 \\ \frac{10}{3}(1 - CP_i) & CP_i \geq 0.7 \end{cases}$
9.	Piecewise	$CP_{i+1} = \begin{cases} \frac{CP_i}{p} & 0 \leq CP_i < p \\ \frac{CP_i - p}{0.5 - p} & p \leq CP_i < 0.5 \\ \frac{1 - p - CP_i}{0.5 - p} & 0.5 \leq CP_i < 1 - p \\ \frac{1 - CP_i}{p} & 1 - p \leq CP_i < 1 \end{cases}$
10.	Gauss/mouse	$CP_{i+1} = \begin{cases} 1 & CP_i = 0 \\ \frac{1}{mod(CP_i, 1)} & \text{otherwise} \end{cases}$

developed an improved version of the WOA based on incorporating Levy-flight to WOA as well as using a chaos theory for the random parameter. In this paper, the authors tried to provide broader movement domain for the search agents by adjusting the parameter C in line 12 of Algorithm 31. To this end, this parameter is replaced with an isotropic Levy step as Equation (24.2):

$$Levy = \frac{\lambda\Gamma\left(\lambda\right) sin\left(\frac{\pi\lambda}{2}\right)}{\pi} \times \frac{1}{s^{1+\lambda}}, s >> s_0 > 0 \tag{24.2}$$

$$s = \frac{U}{|V|^{\lambda-1}}, U \sim N\left(0, \sigma_u^2\right), V \sim N\left(0, \sigma_v^2\right) \tag{24.3}$$

$$\sigma_u^2 = \left[\frac{\Gamma\left(1+\lambda\right)}{\lambda\Gamma\left(\frac{1+\lambda}{2}\right)} \times \frac{sin\left(\frac{\pi\lambda}{2}\right)}{2^{\frac{(\lambda-1)}{2}}}\right]^{\frac{1}{\lambda}}, \sigma_v^2 = 1 \tag{24.4}$$

where *Levy* is a step size, $\Gamma\left(\lambda\right)$ is the standard gamma function with large steps $s > 0$ which is drawn according to Magneta algorithm [4], U and V are samples produced by a Gaussian normal distribution with zero mean value and σ_u^2 and σ_v^2 variance values.

In addition, a logistic chaotic map (presented as ID 1 in Table 24.1) is selected to adjust parameter p.

Another modification in this algorithm is enlisting a mutation operator in case of observing no improvement in the solution. This mutation operator works based on three functions: swap, displacement, and reversion. The first function swaps two individuals randomly. The second function replaces a random subset of the individuals and inserts it in another random section of the population. The third function reserves a random subset of the individuals.

The results demonstrated that this algorithm worked more efficiently as complexity of the optimization problems increased.

24.3.3 Binary WOA

There are many optimization problems with discrete solution spaces. In these problems, it is necessary to modify an algorithm's suitability for a binary search space. Kumar and Kumar [5] proposed a binary version of a WOA based on changing three main phases of the standard WOA, shrinking and encircling, bubble-net attacking method, and prey search. In this modified WOA, the position of the search agents during the shrinking and encircling prey phases are updated accordingly:

$$X\left(Iter\right) = \begin{cases} complement\left(X\left(Iter-1\right)\right) & \text{if } rand < C_{step} \\ X\left(Iter-1\right) & \text{otherwise} \end{cases} \tag{24.5}$$

where C_{step} is the step size given as

$$C_{step} = \frac{1}{1+e^{-10(A\times D-0.5)}} \tag{24.6}$$

In Equation (24.6), A is the coefficient vector in line 11 of Algorithm 31 and D is as follows (line 16 Algorithm 31):

$$D = \left|C \times X^{*}\left(Iter-1\right) - X\left(Iter-1\right)\right| \tag{24.7}$$

The second modification was applied to the bubble-net behavior of whales as follows:

$$X\left(Iter\right) = \begin{cases} complement\left(X\left(Iter-1\right)\right) & \text{if } rand < C_{step'} \\ X\left(Iter-1\right) & \text{otherwise} \end{cases} \tag{24.8}$$

where $C_{step'}$ is as follows:

$$C_{step'} = \frac{1}{1+e^{-10(A\times D'-0.5)}} \tag{24.9}$$

And $D^{'}$ is proposed as:

$$D' = X^{*}\left(Iter-1\right) - X\left(Iter-1\right) \tag{24.10}$$

The third modification considered in the searching for prey phase is based on the following

$$X\left(Iter\right) = \begin{cases} complement\left(X\left(Iter-1\right)\right) & \text{if } rand < C_{step''} \\ X\left(Iter-1\right) & \text{otherwise} \end{cases} \tag{24.11}$$

where $C_{step''}$ is proposed to be as follows:

$$C_{step''} = \frac{1}{1 + e^{-10(A \times D'' - 0.5)}} \tag{24.12}$$

In the Equation (24.13), D'' is defined by the following formulation:

$$D'' = |C \times X_{rand} - X\,(Iter - 1)\,| \tag{24.13}$$

where X_{rand} is a randomly selected whale.

Kumar and Kumar [5] compared their proposed binary WOA algorithm with six other optimization algorithms for several benchmark test functions. In their paper, Kumar and Kumar [5] claimed that their improved algorithm demonstrated better performance based on the obtained results.

24.3.4 Improved WOA

To achieve a better performance with a WOA, several studies have been conducted with different strategies. Sun et al. [6] attempted to improve WOA by adopting three modifications: considering a quadratic interpolation, employing a Levy flight, and using cosine function for updating α in each iteration. Quadratic interpolation (QI) finds the minimum point of the quadratic curve passing through three selected points in the solution space [6]. Following this strategy, quadratic crossover produces a new solution using the best search (X^*) agent and two other solution vectors (Y and Z) using the equation below:

$$x_i = 0.5 \times \frac{\left(y_i^2 - z_i^2\right) \times f\,(X^*) + \left(z_i^2 - x_i^{*2}\right) \times f\,(Y) + \left(x_i^{*2} - y_i^2\right) \times f\,(Z)}{(y_i - z_i) \times f\,(X^*) + (z_i - x_i^*) \times f\,(Y) + (x_i^* - y_i) \times f\,(Z)} \tag{24.14}$$

where i represents i-th dimension of vectors X^*, Y and Z.

This crossover operator used in the exploration phase with a probability threshold of 0.6. To be more exact, a probability decision value will be produced randomly. Then, the previously mentioned quadratic crossover will be utilized to produce new solutions for the probability values of less than 0.6. Otherwise, the spiral-shaped path as the original WOA algorithm methodology will be utilized.

Another improvement is to use a Levy-flight approach. In this modified WOA, a Levy-flight approach replaces the shrinking encircling mechanism (line 15 to 20 in Algorithm 31) to help the algorithm avoid a local minimum. To this end, a simple power-law vision of the Levy distribution is utilized as follows:

$$L\,(s) \sim |s|^{-1-\beta}, 0 < \beta \leq 2, s = \frac{\mu}{|v|^{\frac{1}{\beta}}} \tag{24.15}$$

where β is an index, s is the step length of Levy flight and μ and v are given by the following equations:

$$\mu \sim N\left(0, \sigma_u^2\right), v \sim N\left(0, \sigma_v^2\right), \sigma_\mu = \left[\frac{\Gamma\left(1+\beta\right) \cdot sin\left(\pi \cdot \frac{\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right) \cdot \beta \cdot 2^{\frac{(\beta-1)}{2}}}\right], \sigma_v = 1 \tag{24.16}$$

Based on this method, the position of whales would be updated using the following equation:

$$X\left(Iter\right) = X\left(Iter-1\right) + \frac{1}{\sqrt{Iter-1}} \times sign\left(rand - 0.5\right) \oplus Levy \tag{24.17}$$

where $\oplus$ denotes entry-wise multiplications, *sign* is a sign function with three possible outputs of -1, 0, and 1. In the above equation *Levy* can be calculated as follows:

$$Levy = random\left(size\left(D\right)\right) \oplus L\left(\beta\right) \sim 0.01\left(\frac{\mu}{|v|^{\frac{1}{\beta}}}\right)\left(X_i - X^*\right) \tag{24.18}$$

where size (D) is the scale of the problem, and X_i is the i-th solution vector. As the third modification, a nonlinear controlling pattern was proposed for parameter a as follows:

$$a = 2cos\left(\frac{Iter}{MI}\right) \tag{24.19}$$

In this modified version of WOA, the parameter a is updated using Equation (24.18) and then used to compute the A value. Sun et al. [6] demonstrated improved solutions with their proposed modified WOA for several benchmark functions. It noted in this paper that these modifications prevented WOA from premature convergence and helped the algorithm to escape from the local minima.

24.4 Application of WOA algorithm for optimum design of shallow foundation

24.4.1 Problem description

Shallow footings are critical geotechnical structures that play an important role in behavior of the whole foundation system. To help maintain the serviceability of footings, building codes provide many limitations and rules for their design and analysis. The main challenge for an engineer is to meet all the code's regulations while attempting to find an optimum design. Metaheuristic optimization algorithms are very useful in obtaining a footing design that

meets code requirements and minimizes cost. To this end, the design procedure should be defined in the form of an objective function and input design variables. As demonstrated in Figure 24.1, the design variables are length of footing (X_1), width (X_2), thickness (X_3), depth (X_4), bar number along the longer direction (R_1), the number of bars in long direction (R_2), bar number along the shorter direction (R_3), number of bars in short direction (R_4), bar number of dowels (R_5). Regularly, a shallow foundation may be designed using these mentioned design variables. In this study, this method is simulated for solving the examples 1 and 2 in Section 24.4. Gandomi and Kashani [7] proposed two additional input parameters (E_x and E_y) to determine the location of the column at the top of shallow foundation. It can be helpful to reach more uniform tension distribution under the shallow footing by which a more cost-effective design results. The effect of this modification is examined through the third numerical example in Section 24.4.

The objective function is based on the cost of the shallow footing given as [7, 8]:

$$f_{cost} = C_e V_e + C_f A_f + \xi C_r M_r + \frac{f'_c}{f'_{cmin}} C_c V_c + C_b V_b \tag{24.20}$$

where C_e is the unit cost of excavation, C_f is the unit cost of the framework, C_r is the unit cost of reinforcement, C_c is the unit cost of concrete, C_b is the unit cost of backfill, respectively. Table 24.2 lists the cost values for this problem. ξ is a scale factor that gives the reinforcing steel term a magnitude comparable to that of the other terms, f'_{cmin} is the minimum allowable strength of concrete, V_e is volume of excavation, A_f is area of framework, M_r is the mass of reinforcement, V_c is volume of concrete, and V_b is volume of compacted backfill.

As mentioned earlier, some structural and geotechnical conditions are required to guarantee the strength and stability of the foundation. These limitations have been applied to the design procedure as constraints tabulated in Table 24.3 [8].

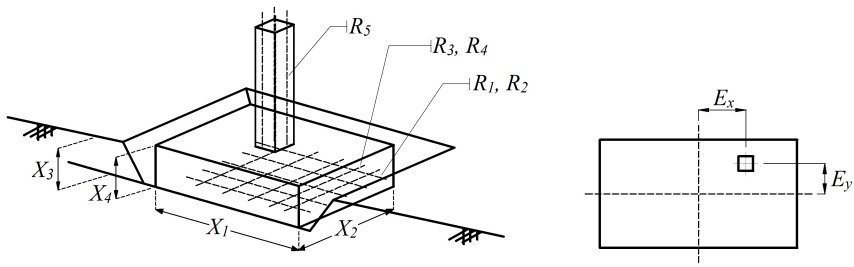

FIGURE 24.1
Schematic view of a shallow foundation with design variables.

TABLE 24.2
Unit cost values.

Input parameter	Unit	Symbol	Value
Excavation	$\$/m^3$	C_e	25.16
Concrete framework	$\$/m^2$	C_f	51.97
Reinforcement	$\$/kg$	C_r	2.16
Concrete	$\$/m^3$	C_c	173.96
Compacted backfill	$\$/m^3$	C_b	3.97

The parameters utilized in Table 24.3 are: FS_B, factor of safety against bearing capacity; FS_{min}, the minimum factor of safety for bearing capacity; δ_{max}, the maximum allowable settlement; V_n, the shear capacity of reinforced concrete foundation; V_u, the ultimate shear force; M_n, the moment capacity of foundation; M_u, the ultimate moment; A_s, steel reinforcement area; $A_{s,min}$, minimum reinforcement area; ϵ_s, tension steel strain; b_{col}, width of the column; cover, the concrete cover; n_{bar}, number of bars in either long or short direction; ω, either X_1 in long direction or X_2 in short direction; d, diameter of rebars and d_{dowel} the bend diameter of the hook dowels. To handle the mentioned constraints a static penalty function approach proposed by Homaifar et al. [9] is utilized. In this way, the objective function would be imposed by a factored violations term following the below equation:

$$fitness_i(X) = f_i(X) + \sum_{j=1}^{m} R_{k,j}\phi_j^2(X) \tag{24.21}$$

where $R_{i,j}$ are the penalty coefficients used, ϕ_j is amount of violation, m is the number of constraints, $f(X)$ is the unpenalized objective function, and $k = 1, 2, ..., l$, where l is the number of levels of violation defined by the user.

24.4.2 How can WOA algorithm be used for this problem?

In this sub-section, the automation of the design of shallow footings using the WOA algorithm is presented. First, $SearchAgents\_no$ search agents are randomly generated within the permitted bound constraints. There are 11 decision variables in the problem; each agent is an 11-dimensional vector. A penalty function proposed by Homaifar et al. [9] is used to reflect any violation of the constraints defined by standard codes to the objective function. In the next step, the best solution will be found by evaluating all the agents' objective function values. Now, the values of α, A, C, and p are initialized. For every i-th search agent, when $p < 0.5$ and $|A| < 1$, the best search agent is chosen as the leader; otherwise, a randomly selected search agent is the leader. When $p > 0.5$, the spiral updating path method is applied. Finally, the penalized objective function is evaluated through a minimization procedure by WOA.

TABLE 24.3
Geotechnical and structural constraints.

Constraint	Function
$g_1(x)$	$\frac{FS_B}{FS_{min}} - 1 > 0$
$g_2(x)$	$\frac{\delta_{max}}{\delta} - 1 > 0$
$g_{[3-5]}(x)$	$\frac{V_n}{V_u} - 1 > 0$
$g_{[6-7]}(x)$	$\frac{M_n}{M_u} - 1 > 0$
$g_{[8-10]}(x)$	$\frac{A_s}{A_{s,min}} - 1 > 0$
$g_{[11-12]}(x)$	$\frac{\epsilon_s}{0.005} - 1 > 0$
$g_{13}(x)$	$\frac{\left(\frac{x_1}{2} - \frac{b_{col}}{2} - cover\right)}{l_{d,short}} - 1 > 0$
$g_{14}(x)$	$\frac{\left(\frac{x_2}{2} - \frac{b_{col}}{2} - cover\right)}{l_{d,long}} - 1 > 0$
$g_{15}(x)$	$\frac{H}{2cover + d_{b,long} + d_{b,short} + \frac{d_{bend}}{2} + d_{dowel} + l_{d,dowel}} - 1 > 0$
$g_{[16-17]}(x)$	$\frac{H}{2cover + n_{bars} d_b + (n_{bars} - 1) s_{min}} - 1 > 0$
$g_{[18-19]}(x)$	$\frac{2cover + d_b + (n_{bars} - 1) s_{max}}{\omega} - 1 > 0$
$g_{20}(x)$	$\frac{P_{bearing}}{P_u} - 1 > 0$
$g_{21}(x)$	$\frac{D_{max}}{D} - 1 > 0$
$g_{22}(x)$	$\frac{D}{D_{min}} - 1 > 0$

This procedure will be repeated iteratively until satisfying the termination criteria.

24.4.3 Description of experiments

In this section, a numerical example is presented to design a low-cost shallow footing using the WOA. Table 24.4 lists combinations of dead and live vertical loads and moments applied to a foundation situated on a cohesionless soil. A column with a cross-sectional area of 400 mm × 400 mm and reinforced with 8 numbers of reinforcement with diameters of 16 mm is selected to direct the effective loads and moments onto the footing. Table 24.5 lists the soil parameters for this design. Over excavation of the length and the width of the footing are equal to 0.3 m. The yield strength of steel reinforcements is 400 MPa and the compressive strength of concrete is 21 MPa. The unit weight of concrete is 23.5 kN/m^3 and the concrete cover is 7 cm. The depth to the bottom of the footing from the ground surface (D) is 0.5 m. Table 24.6 lists the domains for the decision variables defined in Sub-section 24.4.1. There are several inequality constraints to control design of the footing. The geotechnical related constraints guarantee bearing capacity of the soil foundation and prescribe allowable settlement. However, structural constraints estimate the footing strength for bearing capacity of concrete, flexural moment, one-way and two-way shear forces. Finally, the geometrical criteria are related to the

TABLE 24.4
Loads and moments combinations for the case study.

Loading type	Dead	Live
Uniaxial load	650 kN	350 kN
Moment	400 $kN \cdot m$	150 $kN \cdot m$

TABLE 24.5
Utilized soil parameters for the case study.

Input parameters	Unit	Symbol	Values
Internal friction angle of base soil	$^{\circ}$	ϕ	30
Unit weight of base soil	kN/m^3	γ_s	18
Poisson's ratio	-	μ_s	0.3
Elasticity modulus of soil	kPa	E_s	10.500
Maximum allowable settlement	mm	δ	25
Factor of safety for bearing capacity	-	$SF_{Bdesign}$	3

TABLE 24.6
Design variables permitted domain.

Design variables	X_1	X_2	X_3	X_4	R_1	R_2
Unit	cm	cm	cm	cm	–	–
Lower bound	400	400	0	300	2	3
Upper bound	4000	4000	3000	3000	20	18
Design variables	R_3	R_4	R_5	E_x	E_y	
Unit	–	–	–	cm	cm	
Lower bound	2	3	4	-2000	-2000	
Upper bound	20	18	20	2000	2000	

location of the column at the top of foundation and the depth of the shallow foundation.

24.4.4 Results obtained

Table 24.7 lists the final design results for three different shallow footing designs obtained with the WOA. For the current simulations, the maximum number of generations and search agents are 1000 and 50, respectively. Over 101 runs, Table 24.7 lists values for the best and worst designs, and the mean and standard deviation (SD) for the entire set. For all cases, the total number of function evaluations is 50,000. In Case I, a shallow footing with a uniaxial load is designed. A footing design which satisfies all the geotechnical and structural limitations was found after 65 generations. For Case II, an additional external moment is added to the loadings in Case I. In this case, the WOA found valid solutions with no constraint violations after 45 generations

TABLE 24.7
Design cost values for numerical case studies.

Model	Best	Worst	Mean	SD	Median
Case I	43,449.51	50,867.28	44,099.64	1,896.90	43,548.68
Case II	82,913.69	86,744.27	84,424.89	905.71	84,722.97
Case III	49,218.44	113,860.89	62,590.00	21,403.45	50,576.35

and the best solution was obtained after 399 generations. Due to the additional applied moment in CASE II, the best design has a 95.63% increase in cost compared to Case I. In Case III, the position of the column on the top of the footing is changed dynamically using two additional input variables (E_x and E_y). This attempt was helpful to decrease the final cost about 27.17%. In this case, the WOA did not find a stable design until the 23^{rd} generation and the best design was obtained after 348 iterations.

24.5 Conclusions

In this study, several whale optimization algorithms were presented in a detailed step-by-step description. Many important modifications to the general WOA were described and reviewed. To demonstrate the effectiveness of the algorithm, three shallow footings are designed using the WOA. To this end, we explained how WOA deals with a shallow footing optimization problem using a step-by-step algorithm. As can be seen in Section 24.4 the algorithm reached a valid and stable design after some initial iterations. The WOA provides an automated approach to the often tedious and monotonous trial-and-error procedure of designing a foundation that meets all the geotechnical stability and structural strength criteria. More importantly, the WOA significantly reduces the overall cost of the structure while meeting all design criteria.

References

1. S. Mirjalili, A. Lewis, A. "The whale optimization algorithm". *Advances in Engineering Software*, vol. 95, pp. 51-67, 2016.
2. G. Kaur, S. Arora. "Chaotic Whale Optimization Algorithm". *Journal of Computational Design and Engineering*, vol. 5(3), pp. 275-284, 2018.

3. M. Abdel-Basset, L. Abdle-Fatah, A.K. Sangaiah. "An improved Lévy based whale optimization algorithm for bandwidth-efficient virtual machine placement in cloud computing environment". *Cluster Computing*, pp. 1-16, 2018.
4. R. N. Mantegna. "Fast, accurate algorithm for numerical simulation of Levy stable stochastic processes". *Physical Review E*, vol. 49(5), pp. 4677-4683, 1994.
5. V. Kumar, D. Kumar. "Binary whale optimization algorithm and its application to unit commitment problem". *Neural Computing and Applications*, pp. 1-29, 2018.
6. Y. Sun, X. Wang, Y. Chen, Z. Liu. "A modified whale optimization algorithm for large-scale global optimization problems". *Expert Systems with Applications*, vol. 114, pp. 563-577, 2018.
7. A.H. Gandomi, A.R. Kashani. "Construction cost minimization of shallow foundation using recent swarm intelligence techniques". *IEEE Transactions on Industrial Informatics*, vol. 14(3), pp. 1099-1106, 2018.
8. C.V. Camp, A. Assadollahi. "CO2 and cost optimization of reinforced concrete footings using a hybrid big bang-big crunch algorithm". *Structural and Multidisciplinary Optimization*, vol. 48(2), pp. 411-426, 2013.
9. A. Homaifar, S.H.Y Lai, X. Qi. "Constrained optimization via Genetic Algorithms". *Simulation*, vol. 62(4), pp. 242- 254, 1994.