ISSN 1392-1207. MECHANIKA. Vol. 31, No. 5, 2025: 499–504

Application of Worm Optimization Algorithm to Design Challenges: A Focus on Compression Springs

Harun GÖKÇE*, Serhat KARAP**
**Gazi University, Faculty of Technology, Industrial Design Engineering Department, 06560, Ankara Türkiye,*
E-mail: harungokce@yahoo.ca (Corresponding Author)
***TÜBİTAK Defense Industries Research and Development Institute, 06484, Ankara, Türkiye,*
E-mail: serhat.karap@tubitak.gov.tr
https://doi.org/10.5755/j02.mech.39982

1. Introduction

Metaheuristic algorithms offer significant advantages, including not relying on a strictly defined mathematical model, not requiring gradient information, possessing robust search capabilities, and achieving a good balance between solution quality and computational cost. For these reasons, they are widely used to optimize the durability, weight, wear, and corrosion requirements of mechanical systems. Processes characterized as goal-oriented and constrained decision-making activities that aim to produce products meeting well-defined requirements are referred to as engineering design. During the design process, the designer uses mathematical analyses, experiences, and intuitions to develop the design.

Several analyses are conducted to determine whether the developed design is acceptable [1]. Design optimization consists of a search space (feasible solutions), specific objectives (objective functions), and a search process (optimization methods). Feasible solutions are the set of designs derived from all possible values of design variables. Optimization methods aim to find the best (optimal) design among these feasible solutions [2].

Mechanical design encompasses processes where the designer continually optimizes objectives such as durability, bending, wear, weight, and corrosion based on needs [2]. Many mechanical design optimization problems are challenging to solve using conventional optimization methods due to specific constraints [3]. In real-world design optimization problems, the large number of design parameters and the nonlinear effects of these parameters on the objective function increase problem complexity and introduce the possibility of multiple local optima. This pushes designers seeking global optima in mechanical optimization problems to employ more effective and efficient evolutionary optimization methods rather than classical techniques [4].

Evolutionary algorithms are population-based, nature-inspired metaheuristic algorithms. Depending on their source of inspiration, they may be based on swarm intelligence, biological systems, or principles of physics or chemistry [5]. Due to these characteristics, they have been widely preferred in optimization studies over the past two decades. Popular evolutionary algorithms include Simulated Annealing (SA), Genetic Algorithms (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Grey Wolf Optimizer (GWO) and Artificial Bee Colony (ABC) Optimization, which have proven effective [6]. Metaheuristic approaches are well-known and widely applied not only in computer science but also in other fields. Their simplicity, flexibility, and ability to avoid local optima are significant reasons for their preference in various domains [7].

A compression spring is a system that deforms under force or torque and returns the stored energy when it regains its original shape. Springs are elements in mechanical designs that provide functions such as flexibility, vibration isolation, energy absorption, and shock attenuation and are among the most heavily loaded components in machines. General spring design typically involves using trace and cut methods to determine factors such as load, deflection, the number of active coils, and the mean diameter of the spring wire [8].

The design optimization of springs is conducted based on the principles of minimum weight or minimum volume, depending on their application. This study aims to determine the optimal design parameters of a compression spring based on minimum volume. First defined by Sandgren, this problem represents a nonlinear optimization problem. Sandgren proposed using nonlinear integer programming methods to solve this problem [9]. Deb and Goyal highlighted the challenges of solving such design optimization problems using traditional methods, given the binary, discrete, and continuous nature of variable values in engineering design problems, and proposed a combined GA technique employing binary and real-coded variables for different variable types [10]. Lampinen et al. modified the Differential Evolution Algorithm (DEA) to solve nonlinear problems with fully discrete and continuous values [11]. Yokota et al. optimized helical spring design for optimum volume using the GA method [12]. Şahin et al. adapted the GWO algorithm to solve the problem and demonstrated its success [8].

Laith et al. presented a comprehensive review of meta-heuristic optimization methods used to solve engineering design problems. Many problems solved with current optimization techniques [13]. Moreover, specialized review articles on the algorithm have also been published, where the subject is explored in greater depth. Faris at al. discussed the development of GWO and its different variant [14]. Chakraborty at al. provided an extensive review of swarm intelligence algorithms, which are inspired by the collective behavior of decentralized systems [15]. These algorithms have also been used for process optimization in the industry. For example, Calp et al. explored the optimization of project scheduling in dynamic CPM and PERT networks using GA to enhance efficiency and minimize project duration [16]. In a different study, Dener and Calp addressed the optimization of exam scheduling problems in central exams using GA to improve scheduling efficiency and fairness [17]. Trabelsi et al. employed interval computation and constraint propagation to optimize the design of a compression spring for a linear vehicle suspension system, enhancing its performance and reliability [18].

In this study, the optimal design of a compression spring based on minimum volume was carried out using a new metaheuristic optimization algorithm, the Worm Optimization Algorithm (WOA), and the results obtained were compared with previous studies. The first section introduces the optimization problem of minimizing the volume of a compression spring. The second section provides a detailed explanation of the WOA. The third section presents experimental studies and comparative results. The final section concludes with recommendations and conclusions.

2. Optimization of Pressure Spring to Minimum Volume

The optimization of a compression spring for minimum volume, first defined by Sandgren [9], aims to minimize the volume of the spring under static load conditions. The chosen compression spring is manufactured from ASTM A228 music wire spring steel [19], which limits the wire diameter to the values presented in Table 1. To define the optimization problem, three design variables are used: wire diameter (*d*), mean outer diameter (*D*) and the number of active coils (*l*) as illustrated in Fig. 1.

Among these design variables, *D* is continuous, *N* is an integer, and *d* is a discrete variable that can take one of the 42 values listed in Table 1.

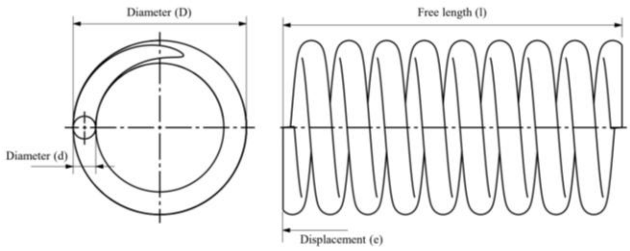


Fig. 1 General 3D model view of the compression spring

The objective function of the problem is given in Eq. (1):

$$f(x) = \frac{\pi^2 x_2 x_1^2 (x_3 + 2)}{4}. \tag{1}$$

The constraints and equations defined for the problem are presented in Table 2 and Table 3.

Other problem parameters are defined as follows: maximum free length of the spring $l_{max} = 14.0$ in., maximum operating load $F_{max} = 1000.0$ lb., pre-load applied for compression $F_p = 300.0$ lb., allowable maximum shear stress $S = 189000.0$ psi, minimum wire diameter $d_{min} = 0.2$ in., maximum outer diameter $D_{max} = 3.0$ in., shear modulus of material $G = 11.5 \times 10^6$ psi, allowable maximum deflection under pre-load $\sigma_{pm} = 6.0$ in., deflection from pre-load to maximum load position $\sigma_w = 1.25$ in. The value ranges for design variables (x_1, x_2, x_3) are as defined above.

Table 1

ASTM A228 standard compression spring wire diameters (x_1, in.) [19]

0.0090	0.0095	0.0104	0.0118	0.0128	0.0132	0.0140
0.0150	0.0162	0.0173	0.0180	0.0200	0.0230	0.0250
0.0280	0.0320	0.0350	0.0470	0.0470	0.0540	0.0630
0.0720	0.0800	0.0920	0.1050	0.1200	0.1350	0.1480
0.1620	0.1770	0.1920	0.2070	0.2250	0.2440	0.2630
0.2830	0.3070	0.3310	0.3620	0.3940	0.4375	0.5000

Table 2

Constraints defined for the problem [9]

$g_1(x) = \frac{8C_f F_{max} x_2}{\pi x_1^3} - S \le 0$	$g_2(x) = l_f - l_{max} \le 0$	$g_3(x) = d_{min} - x_1 \le 0$	$g_4(x) = x_2 - D_{max} \le 0$
$g_5(x) = 3.0 - \frac{x_2}{x_1} \le 0$	$g_6(x) = \sigma_p - \sigma_{pm} \le 0$	$g_7(x) = \sigma_p + \frac{F_{max} - F_p}{K} + 1.05(x_3 + 2)x_1 - l_f \le 0$	
$g_8(x) = \sigma_w - \frac{F_{max} - F_p}{K} \le 0$			

Table 3

Parameters used in the constraint equations [9]

$C_f = \frac{4\left(\frac{x_2}{x_1}\right) - 1}{4\left(\frac{x_2}{x_1}\right) - 4} + \frac{0.615 x_1}{x_2}$	$K = \frac{G x_1^4}{8 x_3 x_2^3}$	$\sigma_p = \frac{F_p}{K}$	$l_f = \frac{F_{max}}{K} + 1.05(x_3 + 2)x_1$
$0.2 \le x_1 \le 1$, $0.6 \le x_2 \le 3$, $1 \le x_3 \le 70$			

3. Worm Optimization Algorithm

WOA is a new type of bio-inspired metaheuristic algorithm that takes inspiration from the contributions of worms to nature, such as aerating the soil through their burrowing actions and enriching it with waste organic matter [20]. The method is inspired by two types of worm reproduction (Reproduction 1 and Reproduction 2). Reproduction 1 involves producing a single offspring on its own, while Reproduction 2 involves producing one or more offspring simultaneously, successfully executed through nine improved crossover operators. Additionally, WOA incorporates Cauchy mutation (CM). Based on the nine improved crossover operators, nine different WOA variations, producing one, two, or three offspring, are proposed.

```
Objective Function f(x), x = (x1, x2, ..., xd)
Define parameters (population size N, maximum number of iterations MaxIter, lower and uppler limits lb,
ub, and discrete values d1_values)
Initialize the population with random solutions
Evaluate the fitness of each individual and identify the best solution (BestSolution)

Set iteration count t = 0
While (t < MaxIter):
  - Update parameter a ((linearly decreasing from 2 to 0)
  For i = 1 to N:
    - Generate random values r1, r2 and p
    - Calculate coefficients A and C:
      A = 2 * a * r1 - a
      C = 2 * r2
    - Update position:
      If (p < 0.5):
        - calculate distance:   D = |C * BestSolution - X|
        - new position:   X_new = BestSolution - A * D
      Else:
       Randomly select an individual (X_rand)
        - calculate distance:   D = |C * X_rand - X|
        - new position:   X_new = X_rand - A * D
    - Check boundaries and adjust d1_values to the nearest discrete value
  - Evaluate the fitness of new individuals and update BestSolution

Increment iteration count t = t + 1

Return BestSolution
```

Fig. 2 Pseudocode of WOA for the problem

WOA is an optimization method inspired by the behaviours of algae, which adapt naturally to environmental conditions to access the resources they need for producing nutrients. In this study, the design of compression springs for minimum volume was optimized using WOA, and the performance of WOA was evaluated for this problem. The algorithm's performance was compared with optimization methods applied to the problem in previous studies. Experimental results demonstrated that WOA is capable of solving design optimization problems consistently and with a low convergence rate. The pseudocode for the algorithm defined for the problem is presented in Fig. 2.

4. Results and Discussion

In this section, the performance of the Worm Optimization Algorithm (WOA) in solving the problem was measured and evaluated comparatively. The best solution values obtained by WOA were compared with the best solutions from previous studies (Table 4). However, sufficient statistical evaluations could not be found in earlier works. For a more detailed and fair comparison, the problem was also solved using WOA, and the results were compared. WOA was chosen because it has been widely applied to optimization problems in recent years [21].

The necessary codes for applying WOA to the problem were prepared and executed using MATLAB 2024b software. Measurements carried out on a 64-bit Windows 10 Pro operation system using an Intel(R) Core(TM) i7-10510U 1.80 GHz CPU with 16 GB RAM. The algorithm's population size was set to 30, and the number of iterations was set to 500. Thus, the maximum number of fitness evaluations was limited to 9000. The cutting force, adaptation parameter, and energy loss parameter, which are WOA-specific parameters, were applied as 2, 0.5, and 0.3, respectively. Both algorithms were run 100 times, and the results were obtained.

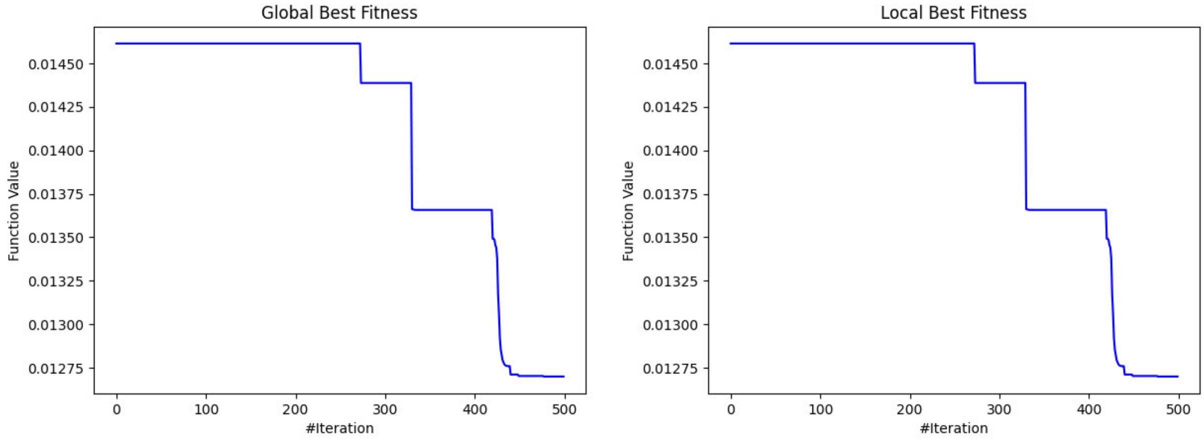
The best values obtained in Table 4 are presented in comparison with the results of previous studies. According to these values, WOA can achieve the known best value for the problem. Additionally, the maximum fitness evaluation count of the algorithm was found to be 7625. It is clearly seen that WOA converges to the solution earlier compared to other algorithms.

Fig. 3 and Fig. 4 analyze the performance of the WOA during the optimization process. Firstly, the Global Best Fitness and Local Best Fitness graph demonstrate how the algorithm optimizes the best solutions over iterations. The Global Best Fitness graph reveals a decrease in the fitness value of the best individual within the entire population as the number of iterations increases. This indicates that the algorithm successfully converges towards better solutions, achieving the desired optimization of the objective function.

Table 4

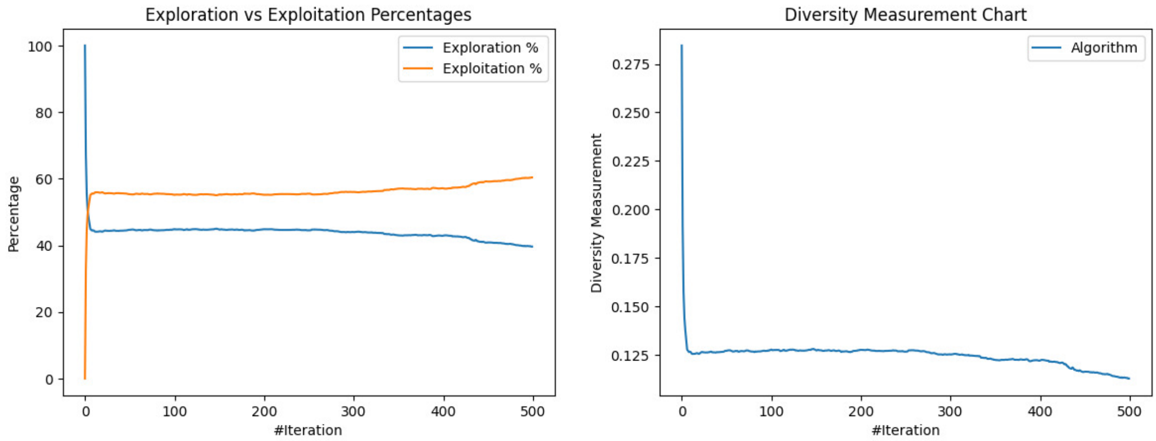
Optimal outcomes achieved for the compression spring

Design variables	Sandgren 1990, [9]	GeneAS 1997, [10]	DE 1999, [11]	PSO 2004, [22]	GWO 2017, [8]	AAA 2018, [23]	WOA
$x_1(d)$, mm	0.2830	0.2830	0.2830	0.2830	0.2830	0.2830	0.2830
$x_2(D)$, mm	1.1807	1.2260	1.22304	1.22304	1.22304	1.22304	1.22304
$x_3(N)$, mm	10	9	9	9	9	9	9
$g_1(x)$, psi	-54309	-713.510	-1008.8114	-1008.8114	-1008.6445	-1008.8114	-1008.8114
$g_2(x)$, in.	-8.8187	-8.933	-8.9456	-8.9456	-8.945628	-8.945635	-8.9456
$g_3(x)$, in.	-0.08298	-0.083	-0.083	-0.083	-0.083	-0.083	-0.083
$g_4(x)$, in.	-1.8193	-1.491	-1.777	-1.777	-1.7769574	-1.776958	-1.776941
$g_5(x)$	-1.1723	-1.337	-1.3217	-1.3217	-1.3217053	-1.32169	-1.32199
$g_6(x)$, in.	-5.4643	-5.461	-5.4643	-5.4643	-5.464283	-5.46428	-5.464279
$g_7(x)$, in.	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
$g_8(x)$	0.0000	-0.0090	0.0000	0.0000	-4.886e-06	-1.1102e-15	-4.886e-06
$f(x)$, in^3.	2.7995	2.665	2.65856	2.65856	2.658562	2.658559	2.658559
Number of Suitability Value Calculations	N/A	N/A	26000	15000	8979	7265	8971



a b

Fig. 3 Performance graphs of WOA methods: a – global best fitness, b – local best fitness



a b

Fig. 4 Performance graphs of WOA methods: a – exploration vs. exploitation percentages, b – diversity measurement chart

Similarly, the Local Best Fitness graph reflects the improvement of local solutions. The consistent decline in both graphs confirms that the algorithm effectively reduces the spring weight at both global and local levels.

Secondly, the Exploration vs. Exploitation Percentages graph evaluates the balance between the algorithm's search and development processes within the solution space. At the beginning of the iterations, the exploration rate is observed to be high, but as the iterations progress, the exploitation rate increases. This behavior shows that the algorithm initially explores a broader solution space to identify potential solutions and then focuses on refining the existing ones to achieve more optimized results. This balance plays a crucial role in optimization processes, indicating that the algorithm operates effectively by preserving diversity and ensuring convergence.

Lastly, the Diversity Measurement Chart assesses the diversity among individuals in the population. A decrease in diversity is observed as the number of iterations increases. This reduction indicates that the algorithm minimizes differences among solutions, converging toward an optimal solution. However, maintaining a certain level of diversity is crucial to prevent the algorithm from being trapped in local minima. In this context, the graphs academically validate that the optimization process was successfully completed and that effective solutions were provided for the spring weight reduction problem.

4. Conclusions

In this study, the WOA was applied for the first time to the optimization of a compression spring design with the objective of minimizing its volume. The results obtained demonstrate that WOA successfully reached the known best solutions for the problem. Furthermore, the algorithm exhibited a high convergence rate and consistency across multiple iterations, validating its reliability and robustness in solving such optimization problems.

The comparative analysis revealed that WOA performed competitively with other established optimiza-tion methods like GA, PSO and GWO. Despite showing a slight disadvantage in runtime, WOA compensated for this with its ability to achieve the optimal solution with fewer fitness evaluations, highlighting its efficiency in exploration and exploitation phases. These characteristics make WOA particularly suitable for complex engineering design problems, where the balance between computational cost and solution quality is critical.

Additionally, the integration of innovative features such as improved crossover operators and Cauchy mutation contributed to WOA's enhanced performance. These features allowed the algorithm to maintain diversity in the search space and effectively avoid local optima, ensuring a higher likelihood of finding the global optimum. This advantage is especially significant for mechanical design problems that often involve nonlinear constraints and multiple variables. From a broader perspective, this study showcases the potential of bio-inspired algorithms like WOA in addressing real-world engineering challenges. While traditional methods face limitations in handling the complexity and nonlinearity of such problems, WOA offers a flexible and adaptive alternative that can cater to diverse optimization needs. Its application in minimizing the volume of compression springs is just one example of its versatility.

Future work could focus on further refining the algorithm to address its relative weakness in runtime by incorporating parallel processing techniques or hybrid approaches that combine WOA with other optimization methods. Additionally, testing WOA on other complex engineering design problems, such as multi-objective optimization or dynamic systems, would further validate its applicability and broaden its scope. In conclusion, the WOA emerges as a promising tool for optimization in mechanical and structural design, bridging the gap between computational efficiency and solution quality. Its successful application in this study paves the way for more extensive use in various domains, such as aerospace, automotive, and energy systems, where high precision and robust optimization solutions are indispensable.

References

1. **Zelinka, I.; Snášel, V.; Abraham, A.** 2013. Handbook of optimization: From Classical to Modern Approach. Springer. 1100p. https://doi.org/10.1007/978-3-642-30504-7.
2. **Rao, R. V.; Savsani, V. J.; Vakhaira, D.P.** 2011. Teaching–learning based optimization: A novel method for constrained mechanical design optimization problems, Computer Aided Design 43(3): 303- 315. https://doi.org/10.1016/j.cad.2010.12.015.
3. **Gokce, H.** 2024. Using of grasshopper optimization algorithm approach for optimal weight design problem of the spur gear, Mechanika 30(5): 472-477.
4. **Avcu, M. E.; Gokce, H.** 2023. Weight Optimization of the Gearbox Using Interior Point Method, International Journal of Precision Engineering and Manufacturing 24: 113-118. https://doi.org/10.1007/s12541-022-00723-1.
5. **Bonebeau, E.; Dorigo, M.; Theraulaz, G.** 1999. Swarm Intelligence: From Natural to Artificial Systems. New York: Oxford University press. 320p. https://doi.org/10.1093/oso/9780195131581.001.0001.
6. **Fister, I. Jr.; Yang, X. S.; Fister, I.; Brest, J.; Fister, D.** 2013. A brief review of nature inspired algorithms for optimization, Elektrotehniški Vestnik 80(3), 116-122.
7. **Mirjalili, S.; Mirjalili, S. M.; Lewis, A.** 2014. Grey wolf optimizer, Advances in Engineering Software 69: 46-61. https://doi.org/10.1016/j.advengsoft.2013.12.007.
8. **Şahin, İ.; Dörterler, M.; Gökçe, H.** 2017. Optimum Design of Compression Spring According to Minimum Volume Using Grey Wolf Optimization Method, Gazi Journal of Engineering Sciences 3(2): 21-27.
9. **Sandgren, E.** 1990. Nonlinear Integer and Discrete Programming in Mechanical Design Optimization, ASME Journal of Mechanical Design 112(2): 223-229. https://doi.org/10.1115/1.2912596.
10. **Deb, K.; Goyal, M.** 1997. Optimizing Engineering Designs Using a Combined Genetic Search, 7th International Conference on Genetic Algorithms.
11. **Lampinen, J.; Zelinka, I.** 1999. Mixed Integer-Discrete-Continuous Optimization By Differential Evolution, Part 1: the optimization method. In: Ošmera, P. (ed.) Proceedings of MENDEL'99, 5th International Mendel Conference on Soft Computing 71-76.
12. **Yokota, T.; Taguchi, T.; Gen, M.** 1997. A solution method for optimal weight design problem of helical spring using genetic algorithms, Computers & Industrial Engineering 33(1-2): 71–76. https://doi.org/10.1016/S0360-8352(97)00044-2.
13. **Abualigah, L.; Elaziz, M. A.; Khasawneh, A. M.; Alshinwan, M.; Ibrahim, R. A.; Al-qaness, M. A. A.; Mirjalili, S.; Sumari, P.; Gandomi, A. H.** 2022. Metaheuristic optimization algorithms for solving real-world mechanical engineering design problems: a comprehensive survey, applications, comparative analysis, and results, Neural Computing and Applications 34: 4081-4110. https://doi.org/10.1007/s00521-021-06747-4.
14. **Faris, H.; Aljarah, I.; Al-Betar, M. A.; Mirjalili, S.** 2018. Grey wolf optimizer: a review of recent variants and applications, Neural Computing and Applications 30: 413-435. https://doi.org/10.1007/s00521-017-3272-5.
15. **Chakraborty, A.; Kar, A.** 2017. Swarm Intelligence: A Review of Algorithms. In: Patnaik, S.; Yang, X. S.; Nakamatsu, K. (eds) Nature-Inspired Computing and Optimization. Modeling and Optimization in Science and Technologies 10: 475-494. https://doi.org/10.1007/978-3-319-50920-4_19.
16. **Calp, M. H.; Akcayol, M. A.** 2018. Optimization of Project Scheduling Activities in Dynamic CPM and PERT Networks Using Genetic Algorithms, Süleyman Demirel University Journal of Natural and Applied Sciences 22(2): 615-627.
17. **Dener, M.; Calp, M. H.** 2018. Solving the exam scheduling problems in central exams with genetic algorithms, Mugla Journal of Science and Technology 4(1): 102-115. https://doi.org/10.22531/muglajsci.423185.
18. **Trabelsi, H.; Yvars, P. A.; Louati, J.; Haddar, M.** 2015. Interval computation and constraint propagation for the optimal design of a compression spring for a linear vehicle suspension system, Mechanism and Machine Theory 84: 67-89. https://doi.org/10.1016/j.mechmachtheory.2014.09.013.
19. Standard Specification for Steel Wire, Music Spring Quality, ASTM International Designation: A228/A228M – 14, US Department of Defense, 2007.
20. **Avcu, M. E.; Gokce, H.; Şahin, İ.** 2024. Chapter 32 - A WOA-based path planning approach for UAVs to avoid collisions in cluttered areas, in Handbook of Whale Optimization Algorithm: 449-461. https://doi.org/10.1016/B978-0-32-395365-8.00038-5.
21. **Arnaout, J. P.** 2020. A worm optimization algorithm to minimize the make span on unrelated parallel machines with sequence-dependent setup times, Annals of Operations Research 285: 273-293. https://doi.org/10.1007/s10479-019-03138-w.

22. **He, S.; Prempain, E.; Wu, Q. H.** 2004. An improved particle swarm optimizer for mechanical design optimization problems, Engineering Optimization 36(5): 585-605.
https://doi.org/10.1080/03052150410001704854.
23. **Atilla, Ü.; Dörterler, M.; Şahin, İ.** 2018. Yapay ağ algoritmasının tasarım optimizasyon problemlerindeki performansı üzerine bir çalışma: Basınç yayı örneği (A study on the performance of artificial neural network algorithms in design optimization problems: The case of pressure springs), Bilişim Teknolojileri Dergisi 11(4): 349-355 (in Turkish).
https://doi.org/10.17671/gazibtd.452992.

H. Gökçe, S. Karap

APPLICATION OF WORM OPTIMIZATION ALGORITHM TO DESIGN CHALLENGES: A FOCUS ON PRESSURE SPRINGS

S u m m a r y

Meta-heuristic algorithms are approximate algorithms that are used in situations where traditional optimization techniques cannot achieve acceptable results. Specifically, they are widely used to achieve optimum design of machine elements. Designing springs according to minimum weight or volume is one of the most basic problems in this field. In this study, a bio-inspired meta-heuristic Worm Optimization Algorithm (WOA) inspired by the contribution of worms to nature was used to solve the problem. The results obtained were compared with those obtained by other algorithms and it was seen that similar data was obtained recently. As a result, it was determined that this algorithm can be used for optimum design of mechanical structures.

Keywords: worm optimization algorithm, pressure spring, design optimization.

Received December 29, 2024
Accepted October 22, 2025