Proceeding Series of the Brazilian Society of Computational and Applied Mathematics, v. 7, n. 1, 2020.

Trabalho apresentado no XXXIX CNMAC, Uberlândia - MG, 2019.

Proceeding Series of the Brazilian Society of Computational and Applied Mathematics

---

Particle Swarm Optimization Method in Optimization of Grid Shell Structures

Ali Mohades Khorasani [1]

Mathematics and Computer Science Department, Amirkabir University of Technology, Tehran, Iran

Marjan Goodarzi [2]

Mathematics and Computer Science Department, Amirkabir University of Technology, Tehran, Iran

Centro de Matemática, Computação e Cognição–CMCC, Universidade Federal do ABC – UFABC, Santo André, SP, Brazil

Majid Forghani-elahabad [3]

Centro de Matemática, Computação e Cognição–CMCC, Universidade Federal do ABC – UFABC, Santo André, SP, Brazil

Abstract. In general, there are two stages in designing a grid shell; creating a form and optimizing the form. Several different approaches such as evolutionary methods or gradient-based techniques have been used for optimization stage. Among evolutionary methods, researchers mostly have used the genetic algorithms (GA) whereas the particle swarm optimization (PSO) has been shown to be more efficient than the GA in discrete problems. Hence, here, we use PSO method for improving the regularity of grid shell structures. It is illustrated how PSO method can be used in optimizing the grid shell structures. The technique is explained step by step, and therefore there is no need for any previous knowledge of PSO.

Keywords. Grid shells, Shape optimization, Particle swarm optimization

1 Introduction

For optimization of a grid shell structure, various aspects of the structures from the economic and structural aspects to the aesthetic, functional, and constructional ones have been considered and analyzed in the literature by gradient-based techniques such as compass method [2], evolutionary approaches such as genetic algorithms [6, 7], or some other techniques such as sphere packing approach [1]. For example, Richardson et al. [6] presented a two-phase design technique that uses dynamic relaxation for finding the funicular form of the grid shell in the first phase, and obtains the optimal nodal positions employing

[1]mohades@aut.ac.ir.

[2]marjangoodarzi@aut.ac.ir, m.goodarzi@ufabc.edu.br

[3]m.forghani@ufabc.edu.br

a generic algorithm taking the material minimization and improved structural performance into account. Using a multi-objective genetic algorithm, Winslow et al. [7] established a design tool for synthesis of optimal grid shell structures taking into account two or more load cases such as wind load. Regularity is one of the aspects of a grid shell which effects on the economy of the structure. In fact, decreasing the number of different types of the elements, regularity of a grid shell lessens the cost of manufacture and the assembly of the components [4]. Therefore, we use the particle swarm optimization (PSO) method to improve the regularity of a grid shell structure in this work. In fact, among all the evolutionary methods such as memetic algorithms, genetic algorithms, PSO, shuffled frog leaping, and ant-colony systems [3], the most considered technique in the literature of optimization of grid shells is the genetic technique [1, 6, 7]. However, the PSO technique performs better than the genetic method in terms of success rate and solution quality [3], and this is why PSO is considered in this work. It is noted that the main contribution of this work is to illustrate how the PSO technique can be used for improving the regularity of grid shells.

2 Main block

Here, we first briefly explain the PSO method to make the work readable without any previous knowledge of this method, and then it is described how the method can be used for improving the grid shells' regularity.

PSO method, which is a nature-inspired evolutionary optimization technique, has been initially proposed by Kennedy and Eberhart [5] and has been applied in numerous different optimization problems [3,5]. It is based on exploration and exploitation. The former is the ability to explore diverse regions for locating a good optimum and the latter is the ability of concentrating the search on every side of an encouraging space for purifying a potential solution [5]. This way, the particle moves throughout the space guided by the memory of its own best position (*Pbest*) and knowledge of overall best position (*Gbest*), and considering the inertia to keep the previous direction of movement. The experiences of *Pbest* and *Gbest* are accelerated by some factors multiplied by random numbers. Moreover, an inertia factor is always considered in this technique. In fact, PSO is an iterative approach which starts with an initial population and tends to the best solution iteratively. Each member of the population has two attributes position and velocity which change in every iteration. Let ${x_i}^k$ and ${v_i}^k$ respectively denote the position and velocity of particle i in the kth iteration in the search space. The velocity of particle i for the next iteration is determined by

$$v_i^{k+1} = wv_i^k + c_1 r_1 \left(Pbest_i^k - \ x_i^k \right) + c_2 r_2 \left(Gbest^k - \ x_i^k \right), \tag{1}$$

where w is the inertia factor, c_1 and c_2 are the acceleration factors for the experiences of *Pbest* and *Gbest*, respectively, r_1 and r_2 are random numbers (vectors) in the interval [0, 1]. It is noted that ${Pbest_i}^k$ is the best experienced positon for particle i up till the kth iteration, and $Gbest^k$ is the best experienced position among all the particles so far. After calculating the velocity of the particle i, its position is obtained from

$$x_i^{k+1} = v_i^{k+1} + x_i^k. \tag{2}$$

Usually, the inertia factor, i.e., w, is selected between 0.4 and 0.9. However, some researchers demonstrated that considering a greater amount for w, even w=1, at the beginning, and then decrease it dynamically in each iteration using a damping factor, the algorithm converges to the best solution faster. Therefore, we also consider w=1 with a damping factor of $wdamp = 0.99$ in this work which increases the velocity of convergence. The acceleration coefficients are considered here as $c_1 = 2$ and $c_2 = 2$. To illustrate the method, a simple grid containing two regular pentagons is considered which contains 11 nodes and 20 elements (see Fig. 1). There are 11 nodes and 25 edges in this simple grid. It is assumed that the nodes on the bigger polygon are fixed and cannot be displaced, and hence the five edges of the bigger pentagon are also fixed. Hence, the main aim here is to displace the nodes numbered 6, 7, ..., 11 so that the regularity of the given grid is improved. Since the nodes and edges of the bigger pentagon are fixed, there are 6 nodes and 20 edges which can be changed. The length of the edges from node i to node 5+i, for $i = 1, 2, \ldots, 5$, is $l_1 = \ldots = l_5 = 2$. The length of the edge between node i and node 6+i, for $i = 1, 2, \ldots, 5$, is $l_6 = \ldots = l_{10} = 2.8541$, and the lengths of the edges form nodes 6, 7, 8, 9, and 10 to node 11 are respectively equal to $l_{11} = 0.97473$, $l_{12} = 1.0787$, $l_{13} = 1.0787$, $l_{14} = 0.97473$, and $l_{15} = 0.90451$. Moreover, the length of the smaller pentagon's side is $l_{16} = \ldots = l_{20} = 1.1756$.

To improve the regularity of the given grid in Fig. 1 by using the PSO algorithm, it is required to define the particles. Since the nodal positions should be changed to improve the regularity, we consider the coordinates of nodes numbered 6, 7, ..., 11 as the particles. At the beginning, the associated particle with the given simple grid is $X^* = \begin{bmatrix} x \\ y \end{bmatrix}^* = \begin{bmatrix} 0.0955 & 0.3090 & -0.8090 & -0.8090 & 0.3090 & 1.0000 \\ -0.0000 & 0.9511 & 0.5878 & -0.5878 & -0.9511 & -0.0000 \end{bmatrix}$, where the first column is the coordinates associated with the inner point in the smaller pentagon, and the other columns are the coordinates related to the nodes of smaller pentagon.

To generate an initial population, the lower bound $LX = X^* - k \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$ and upper bound $UX = X^* + k \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$ are considered, where k can be any real number, and coordinates for n points, i.e., n particles, are randomly generated between LX and UX. It is noted that we consider $k = 1.4$ and set the velocity of all the particles to zero for the initial population, i.e., ${v_i}^0 = 0$, for $i = 1, 2, \ldots, n$. Moreover, in our experimentatl results, n =15 is considered as the number of population.

Since it is aimed to improve the regularity of the grid, the cost function is set as the standard deviation of all the 20 variable lengths in the grid, i.e.,

$$C(X) = \left(\frac{1}{m-1} \sum_{i=1}^{m} \left(l_i - \bar{l} \right)^2 \right)^{\frac{1}{2}}, \tag{3}$$

where l_i is the length of ith element in the grid, for $i = 1, 2, \ldots, m$, and $\bar{l} = \frac{1}{m} \sum_{i=1}^{m} l_i$.

For example, for the given grid in Fig. 1, the 20 lengths are $l_1 = \ldots = l_5 = 2$, $l_6 = \ldots = l_{10} = 2.8541$, $l_{11} = 0.97473$, $l_{12} = 1.0787$, $l_{13} = 1.0787$, $l_{14} = 0.97473$, $l_{15} = 0.90451$, and $l_{16} = \ldots = l_{20} = 1.1756$, and consequently the grid cost is 0.75653.

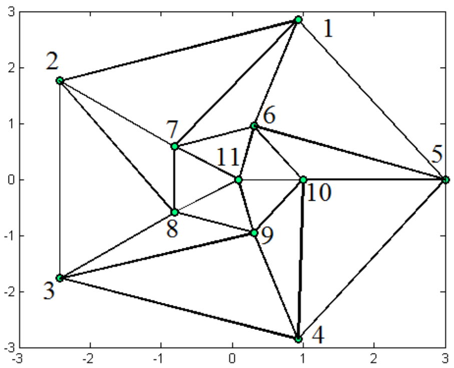

Figure 1: A simple grid example.

For velocity clamping, we consider $V_{max} = 0.1(UX - LX)$ and $V_{min} = -V_{max}$, and then after calculation of velocities by using Eq. (1), they are adjusted by

$$V = \begin{cases} V_{min} & \text{if } V < V_{min} \\ V & \text{if } V_{min} \leq V \leq V_{max} \\ V_{max} & \text{if } V_{max} < V. \end{cases} \tag{4}$$

For velocity mirror effect, the following equation is considered for the given grid in Fig. 1.

$$v_i^k = \begin{cases} -v_i^k & \text{if} \quad x_i^{k+1} < Lx_i \quad \text{or} \quad x_i^{k+1} > Ux_i \\ v_i^k & \text{otherwise.} \end{cases} \tag{5}$$

Apart from all the modifications and limits on the velocities, the new positions of some particles may be out of search area. Therefore, in each iteration, if the position of a particle exceeds the lower or upper bounds, the position of the particle is replaced with the associated bound to keep being in the search area.

$$x_i^{k+1} = \begin{cases} Lx_i & \text{if } x_i^{k+1} < Lx_i \\ x_i^{k+1} & \text{if } Lx_i \leq x_i^{k+1} \leq Ux_i \\ Ux_i & \text{if } Ux_i < x_i^{k+1} \end{cases} \tag{6}$$

As the stopping criteria in PSO method, one may (1) stop by exceeding the given maximum number of iterations, (2) stop when the improvement of solution in a given number of iterations is less than a given limit, (3) stop when a satisfactory solution is determined, or (4) stop when the cost function slope is almost zero.

In our numerical experiments for this example, it was seen that with the selected factors and the first group of PSO parameters, the algorithm finds the solution in less than 140 iterations. The final grid obtained by the algorithm is given in Fig. 2. The standard deviation of 20 lengths in the final grid is 0.1104. It is noted that theoretically is impossible to have the standard variation of zero in this example. The PSO's diagram is provided in Fig. 3. This diagram shows how the algorithm gets close to the final solution.

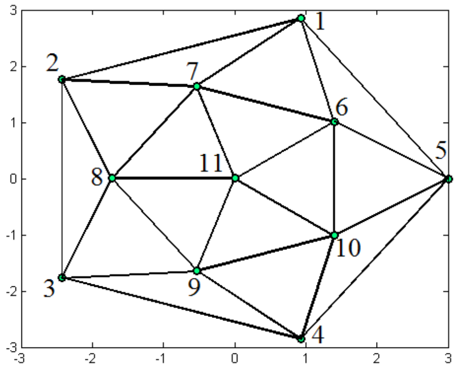

Figure 2: The final regular grid obtained by the algorithm.

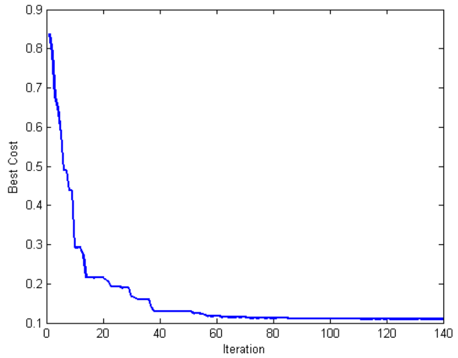

Figure 3: The PSO diagram.

3 Concluding remarks

Grid, lattice, or reticulated shells are generally defined as structures with the shape and rigidity of a double curvature shell consisting of a grid not a continuous surface. Several gradient-based and evolutionary approaches have been proposed in the literature for optimization of grid shell structures. Among all the evolutionary techniques, genetic technique is the one which has been employed the most whereas the particle swarm optimization (PSO) technique has been shown to be more efficient than the genetic algorithm in discrete optimization. Hence, in this work, the PSO algorithm was employed to improve the regularity of grid shell structures. A simple grid containing two regular pentagons was considered to illustrate the approach in details.

Acknowledgement

The first two authors thanks Amirkabir University of Technology and the third author thanks Universidade Federal do ABC for supporting this work.

References

[1] P. Basso, A.E.D. Grosso, A. Pugnale, and M. Sassone. Computational morphogenesis in architecture: cost optimization of free-form grid shells, *Journal of the International Association for Shell and Spatial Structures*, 50: 143-150, 2009.

[2] L. Bouhaya, O. Baverel, and J.F. Caron. Optimization of gridshell bar orientation using a simplified genetic approach, *Struct Multidisc Optim*, 50: 839–848, 2014.

[3] E. Elbeltagi, T. Hegazy, and D. Grierson. Comparison among five evolutionary-based optimization algorithms, *Adv. Eng. Inform.*, 19: 43–53, 2005.

[4] R. Haghnazar, H. Nooshin, and M. Golabchi. Improving the regularity of geodesic domes using the concept of stepping projection, *Int. J. Space Struct.*, 29: 81–95, 2014.

[5] J. Kennedy, and R.C. Eberhart. Particle swarm optimization. *Proceedings of the IEEE international conference on neural networks IV*, 1942–1948, 1995.

[6] J.N. Richardson, S. Adriaenssens, R.F. Coelho, and P. Bouillard, Coupled form-finding and grid optimization approach for single layer grid shells. *Eng. Struct.*, 52: 230–239, 2013.

[7] P. Winslow, S. Pellegrino, and S. Sharma. Multi-objective optimization of free-form grid structures. *Struct. Multidiscip. Optim.*, 4(1-6): 257–269, 2010.

