# Optimum Design of Plane Trusses Using Mathematical and Metaheuristic Algorithms on a Spreadsheet

**Vagelis Plevris**
Department of Civil and Architectural Engineering, Qatar University, Doha, Qatar
vplevris@qu.edu.qa

**Ismail Hafez Mohammed Elias**
Department of Civil and Architectural Engineering, Qatar University, Doha, Qatar
ie1707600@student.qu.edu.qa

**Aristotelis E. Charalampakis**
Department of Civil Engineering, University of West Attica, Athens, Greece
achar@uniwa.gr

## Abstract

Mathematical optimization refers to the process of finding the values of variables that maximize or minimize a function. Structural optimization is the process of designing a structure in such a way as to minimize its weight or cost, while meeting a set of performance requirements, ensuring that it is robust, lightweight, and efficient. Two large categories of optimization algorithms are mathematical and metaheuristic algorithms. The ones of the first rely on mathematical principles, are deterministic and exact but may fail if the problem is too large or complex. The latter category, metaheuristics, represents algorithms that are used to find approximate solutions. They are high-level strategies that guide the search toward a good solution, rather than being a specific, deterministic algorithm. They are often used for problems where it is difficult or impractical to find the optimal solution using exact methods. Metaheuristics typically involve iteratively improving a solution through some type of search or exploration process. They make use of techniques from probability and statistics, such as randomization and stochastic optimization, to explore the search space and guide the search toward good solutions. Some examples include genetic algorithms, simulated annealing, differential evolution (DE), particle swarm optimization (PSO), and ant colony optimization. In this study, a mathematical optimizer and two metaheuristics (DE, PSO), are employed for the optimum structural design of plane truss structures aiming to minimize the weight of the structure under constraints on allowable displacements and stresses. A 10-bar plane truss is considered as the numerical example of the study. The constraints are checked by performing an analysis with matrix methods. All calculations are done on a spreadsheet. The results of the algorithms are compared to each other as well as to results from the literature in terms of convergence speed, number of function evaluations, and accuracy of the solution.

**Keywords:** Optimization; Metaheuristic; Differential Evolution; Particle Swarm Optimization; Spreadsheet

## 1 Introduction

A spreadsheet application is a computer program that allows users to create and edit spreadsheets. Spreadsheets are used to organize data and perform calculations with numerical information. They consist of rows and columns, with cells that can contain text, numbers, or formulas. Some popular spreadsheet applications include Microsoft® Excel® (hereafter, simply Excel), Google Sheets, Open

Office Calc., and Apple Numbers. Spreadsheets are commonly used for budgeting, financial analysis, and data management, but they also see significant applications in various engineering fields, such as structural engineering, (Christy, 2006), water engineering, (Pandit, 2015), geotechnical engineering, (Ozcep, 2010), among others. They can do basic calculations such as cost estimates, schedule and cost control, and markup estimation, as well as structural calculations of reactions, stresses, strains, deflections, and others. Many engineering firms rely on spreadsheets for complex engineering calculations of all kinds. Spreadsheets may not exhibit the full power and advanced capabilities that advanced coding with a computer language can offer, but they have the advantage that they are simple to use and easy to understand, without the need for the user to use coding to make simple or more complex calculations.

In this work, we employ Excel for the optimum design of plane truss structures. In particular, we use the Excel Solver, which is a built-in optimizer of Excel, and the xl Optimizer software[1] which is an external plug-in for Excel. The Solver has been successfully used for solving various kinds of structural analysis problems (Bhatti, 2005; Rady & Mahfouz, 2022; Shahnam, 2003).

## 2 Structural Optimization of Trusses

Structural optimization is the process of designing a structure (such as a bridge, building, or aircraft) in such a way as to minimize its weight or cost, while still meeting a set of performance requirements. It is a branch of engineering that involves the use of optimization techniques and principles of structural mechanics to design structures that are robust, lightweight, and efficient. Structural optimization typically involves three steps: (i) defining the design variables, constraints, and objective function; (ii) formulating the optimization problem; and (iii) solving the problem. The design variables are the parameters that can be varied in the design, such as the shape, size, and material of the structure. The constraints are the requirements that the structure must meet, such as strength, stability, and durability. The objective function is a measure of the performance of the structure, such as its weight or cost.

A truss is a structure that is composed of a set of slender, pin-jointed members that are used to span a distance and support loads. Trusses are commonly used in engineering and construction. They are usually made of steel or other metals and are used in a wide variety of applications, including bridges, buildings, towers, and other structures. In truss sizing optimization, the aim is usually to find the right sections of the structural members of a truss that would lead to minimum weight while satisfying the design constraints. Such a problem can be formulated mathematically as:

$$\begin{aligned} &\min_{x_i} \quad W(\boldsymbol{x}) = \sum_{i=1}^{N_e} L_i \cdot x_i \cdot \rho_i \\ &\text{Subject to} \\ &g_k(\boldsymbol{x}) \leq 0, \quad k = \{1, \ldots, K\} \end{aligned} \tag{1}$$

Where $W(\boldsymbol{x})$ is the total structural weight (or mass), $N_e$ is the number of elements, $\boldsymbol{x}$ is a vector with the cross-section areas $x_i$ of each member, $L_i$ is the length of each member, $\rho_i$ is the material density of element $i$, and $g(\boldsymbol{x})$ are the behavioral constraints, $K$ in total. Side constraints are also usually imposed on the lower and upper limits of the design variables, i.e. $lb_i \leq x_i \leq ub_i$, where $lb_i$ and $ub_i$ denote the lower and upper bounds of the $i$-th design variable, respectively.

[1] Xl Optimizer, Add-in version overview, accessed 5 January 2023, https://xloptimizer.com/overview

## 3 Optimization Algorithms

In this work, we have used the Excel Solver and the xl Optimizer plug-ins for our optimization needs. The Solver has been used for optimizing the truss structure using mathematical algorithms and in particular the **Generalized Reduced Gradient (GRG)** method. Various metaheuristic algorithms are available in the literature for dealing with constrained structural optimization problems (Lagaros et al., 2022). In xl Optimizer, we have used two of the offered metaheuristic algorithms, namely, **Differential Evolution (DE)** and **Particle Swarm Optimization (PSO)**.

### 3.1 Generalized Reduced Gradient: Single Point and Multi start

GRG is an algorithm for solving optimization problems with continuous variables and linear or nonlinear constraints. It is an iterative method that starts with an initial feasible point and then uses the gradient of the objective function to move toward the optimal solution. GRG works by constructing a quadratic approximation of the objective function at each iteration and then solving a series of quadratic programming sub problems to find the next iterate. The algorithm stops when the quadratic approximation is good enough, or when a specified termination criterion is met. One of the main advantages of the GRG method is that it can handle a wide range of optimization problems, including problems with linear constraints, nonlinear constraints, and mixed integer variables. It is also relatively efficient, especially for problems with a large number of variables. However, it can be sensitive to the initial point and may not always converge to the global optimum. It can also be computationally expensive for large problems with many constraints. GRG is the default algorithm used by Solver and it is offered in two variations: (i) "Single point" (default setting), where the algorithm is run once, and (ii) "Multi start" where GRG will run repeatedly, starting from different (automatically chosen) starting values for the decision variables. This process will usually find a better solution, at the expense of requiring more computing time.

### 3.2 Differential Evolution

Differential Evolution (DE) is a metaheuristic optimization algorithm that was introduced by (Storn & Price, 1995). It is a population-based algorithm used to find approximate solutions to optimization problems, particularly problems with a large number of variables. In DE, a population of candidate solutions is maintained, and each member of the population is represented by a set of variables or parameters. At each iteration, DE generates a new candidate solution by combining the variables of three existing solutions in a specific way. This new candidate solution is then evaluated, and if it is better than the worst of the three solutions that were used to generate it, it is added to the population. The key idea behind DE is that the combination of variables from different solutions can create new, potentially better solutions that would not have been found by simply exploring the search space randomly. DE uses several strategies to control the search process, such as scaling and crossover, which help to balance exploration and exploitation and avoid getting stuck in local minima. It is simple to implement and has been shown to be effective in many types of problems, particularly those with a large number of variables or complex, multimodal search spaces. DE comes in many variations Georgioudakis & Plevris (2020a); Georgioudakis & Plevris (2020b) and has been applied successfully to, many structural engineering optimization problems in the past (Georgioudakis & Plevris, 2018; Kao et al., 2020). In this work, we use the standard DE/Rand/1/bin scheme.

### 3.3 Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a metaheuristic optimization algorithm that is inspired by the social behavior of birds. It was first introduced by Kennedy & Eberhart (1995) as a computational method for optimizing continuous, nonlinear problems. In PSO, a set of “particles” are used to search for a good solution to a problem. Each particle represents a potential solution to the problem and is initialized with a randomly generated position and velocity. The position of each particle is updated at each iteration based on its previous position and velocity, as well as, the positions and velocities of the other particles in the swarm. The movement of the particles is guided by two types of information: the “personal best” position of each particle, which is the best position that the particle has achieved so far; and the “global best” position, which is the best position that has been achieved by any particle in the swarm. At each iteration, each particle adjusts its velocity and position based on these two sources of information, with the goal of moving toward better positions in the search space. PSO has several attractive properties, including simplicity, flexibility, and robustness. It is particularly well suited to problems with many variables, or problems that are multimodal or noisy. It has been applied to a wide range of optimization problems, including function optimization, clustering, and image processing. It has also very interesting applications in structural engineering (Charalampakis & Dimou, 2010; Plevris et al., 2011).

## 4 Numerical Example

The 2D truss that is investigated and optimized is presented in Fig. **1**. The Young’s modulus $E$is equal to 10,000 ksi (68.95 GPa), the length $L$ is 360 in (9.144 m), the load $P$ is 100 kip (444.82 kN) and the material density $\rho$ is 0.1 lb/in$^3$ (2767.99 kg/m$^3$). There are 10 truss members and each one represents a design variable; this results in 10 design variables for the optimization problem, corresponding to the section area for each member, in the interval [0.1, 35] (in$^2$) (0.64 to 225.81 cm$^2$). The constraints of the optimization problem are imposed on stresses and displacements. In particular, the maximum allowable stress is 25 ksi (172.37 MPa) for any member, in compression or tension, while the maximum allowable displacement is 2 in (5.08 cm), in any of the ±x and ±y directions. Buckling is not taken into account as a constraint in this problem. The optimization objective is the minimization of the total weight (or total mass) of the structure, under the imposed constraints. This example is a standard benchmark problem that has been studied thoroughly by several researchers in the literature (Plevris, 2009). All simulations and analyses are performed in Excel on a personal computer running Windows 11, equipped with an Intel Core i9-8950HK (2.90 GHz) and 64 GB Ram.

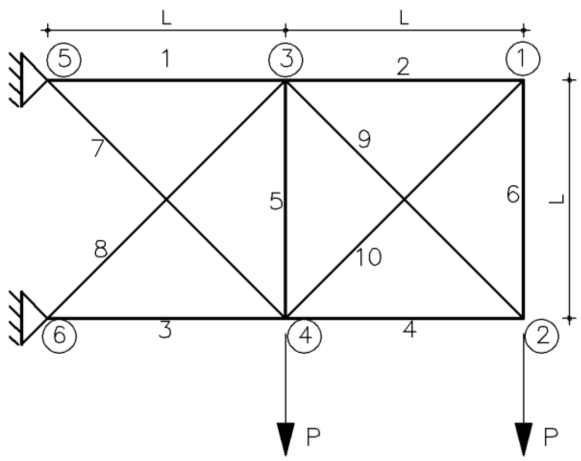


**Fig. 1:** The 10-Bar Plane Truss Model

### 4.1 Results from the Literature

Table 1 shows various results from the literature for the same problem. In this table, constraint violations are reported using a bold font. The best results that strictly do not violate any of the constraints are the ones reported by (Plevris & Papadrakakis, 2011) and (Charalampakis & Tsiatas, 2019). Some authors report lower objective function values, but their designs violate some of the constraints, as shown in the table.

**Table 1:** Optimization results from the literature

| | **(Galante, 1992)** | **(Haftka & Gürdal, 1992)** | **(El-Sayed & Jang, 1994)** | **(Memari & Fuladgar, 1994)** | **(Ghasemi et al., 1997)** | **(Perez & Behdinan, 2007)** | **(Plevris & Papadrakakis, 2011)** | **(Charalampakis & Tsiatas, 2019)** |
|---|---|---|---|---|---|---|---|---|
| A1 ($in^2$) | 30.4400 | 30.5200 | 32.9700 | 30.5610 | 25.7300 | 33.5000 | 30.5218 | 30.5310 |
| A2 ($in^2$) | 0.1000 | 0.1000 | 0.1000 | 0.1000 | 0.1090 | 0.1000 | 0.1000 | 0.1000 |
| A3 ($in^2$) | 21.7900 | 23.2000 | 22.7990 | 27.9460 | 24.8500 | 22.7660 | 23.1999 | 23.1970 |
| A4 ($in^2$) | 14.2600 | 15.2200 | 14.1460 | 13.6190 | 16.3500 | 14.4170 | 15.2229 | 15.2280 |
| A5 ($in^2$) | 0.1000 | 0.1000 | 0.1000 | 0.1000 | 0.1060 | 0.1000 | 0.1000 | 0.1000 |
| A6 ($in^2$) | 0.4510 | 0.5510 | 0.7390 | 0.1000 | 0.1090 | 0.1000 | 0.5514 | 0.5500 |
| A7 ($in^2$) | 7.6280 | 7.4570 | 6.3810 | 7.9070 | 8.7000 | 7.5340 | 7.4572 | 7.4590 |
| A8 ($in^2$) | 21.6300 | 21.0400 | 20.9120 | 19.3450 | 21.4100 | 20.4670 | 21.0364 | 21.0450 |
| A9 ($in^2$) | 21.3600 | 21.5300 | 20.9780 | 19.2730 | 22.3000 | 20.3920 | 21.5285 | 21.5110 |
| A10 ($in^2$) | 0.1000 | 0.1000 | 0.1000 | 0.1000 | 0.1220 | 0.1000 | 0.1000 | 0.1000 |
| Weight (lb) | 4999.22 | 5060.93 | 5013.39 | 4981.09 | 5095.64 | 5024.25 | 5060.85 | 5060.86 |
| Max Stress (ksi) | **25.08667** | **25.00271** | **31.28849** | 20.60011 | 18.52549 | **25.01711** | 24.99998 | 24.99991 |
| Max Displ. (in) | **2.02798** | 1.99996 | **2.01307** | **2.06047** | **2.01368** | **2.03890** | $2.00000^-$ | $2.00000^-$ |

## 4.2 Computational Tools: Excel Solver and Xl optimizer

In this work, we have used the built-in Solver of Excel and xl Optimizer software. In particular, we have used two variations of the GRG mathematical algorithm of the Solver, namely the Single point GRG and the Multi start GRG, and two metaheuristic algorithms of xl Optimizer, namely DE and PSO. It has to be noted that Solver also offers a third alternative, an Evolutionary Algorithm (a Genetic Algorithm, GA) (Goldberg, 1989), while xl Optimizer offers a variety of additional optimization algorithms, such as different variants of DE, Enhanced PSO (EPSO), GA, Simulated Annealing, Artificial Bee colony, and others. In addition, xl Optimizer can perform multi-objective optimization in which two or more objectives, usually conflicting with each other, are to be optimized simultaneously, an advanced topic with very interesting and useful engineering applications (Lagaros et al., 2005; Papadrakakis et al., 2002).

To use both tools, one first needs to set up the problem in an Excel spreadsheet. This typically involves defining the variables to be optimized and the constraints that must be satisfied. In our example, all these data are entered in Excel via their corresponding cells in a spreadsheet, i.e. the cell that contains the objective function, the cells that contain the variables, and the cells that contain the constraints. The truss analysis is done using matrix methods (the direct stiffness method) for plane trusses. For simplicity, no Macros or any other coding has been used for analysis or any other purposes. All the analysis is done using matrix operations in Excel, using standard Excel functions for matrix manipulation, such as “Mmult”, “Minverse” and “Transpose”, without the use of any coding. In most cases, we used the default settings of each optimization tool, unless otherwise stated.

Excel Solver is a tool for Microsoft Excel that allows one to find an optimal solution to a problem that has multiple variables and constraints. The Solver comes together with Excel, as part of its basic installation, but it is deactivated by default and the user has to enable it through Excel options (Add-ins). Many different types of problems can be solved using Solver, including linear programming, nonlinear programming, and integer programming problems. It can also be used to solve problems involving complex formulas or large datasets.

Xl Optimizer is an advanced optimization toolbox that can be installed as an add-in to Excel. It implements a large variety of customizable, state-of-the-art metaheuristic algorithms in a single environment, featuring an intuitive user interface. The xl Optimizer add-in is installed in Excel's ribbon and does not require any external programs. It is ideal for very difficult (multi-parametric, non-differentiable, discontinuous, combinatorial, deceptive, etc.) and/or expensive optimization problems.

### 4.3 Optimization Results with Solver

**Single Point Solution**

First, we use the GRG Nonlinear optimizer, which is the default optimizer of Solver. The solver options used are the following default options: "*Max Time: Unlimited, Iterations: Unlimited, Precision: 0.000001, Use Automatic Scaling, Convergence: 0.0001, Population Size: 100, Random Seed: 0, Derivatives Forward, Require Bounds, Max Sub problems: Unlimited, Max Integer Sols: Unlimited, Integer Tolerance: 1%, Assume Non Negative*". In the single point solution, it is important to define a proper starting point, as the solution reached usually depends on the point initially set. In our case, we started the run with a "heavy" design, where design variables have been set to their upper limit, i.e., 35 $in^2$. We also tried other options, starting all design variables from the 25, 15, and 5 $in^2$points, but we did not notice any significant difference in the obtained results. The results of the Single point solution with the GRG Nonlinear optimizer are shown in Table 2. The minimum obtained is 5076.67 lb. The max displacement is reported as 2 in, with a very slight violation of this constraint. The solution time was very fast, 0.797 seconds, for 30 iterations of the algorithm.

**Multi Start Solution**

For the Multi start case, the default population size given by Solver is 100 but instead, we used the value of 10 as we noticed that there is no significant improvement in the objective function after the first runs, so a value of 100 would only cause delay. The starting point is again set with all design variables set at their highest value (35 $in^2$). Using Multi start, we obtain a better result (5060.84 lb), at the expense of spending more computing time (14.4 times more than the time needed for the Single Point Solution). In addition, there are slight violations of the constraints, as shown in Table 2.

### 4.4 Optimization Results with Xl optimizer

In xl Optimizer the termination criterion is set as "*OR (FE>=20000, TIME_MIN>10)*", which means that the maximum number of iterations is 20,000 for all methods used, and also the maximum time of calculations is 10 minutes. The latter criterion was never triggered as the criterion related to the maximum number of iterations was triggered before, for all cases. The default constraint function is "*1000*V+100*". In our case, we used a stricter constraint, "*1000*V+10000*", to make sure that the optimum design will be feasible (with no constraint violations) in the end.

**Differential Evolution**

The DE variant used is the default one offered by the program. It is the DE/Rand/1/bin scheme using the following default settings: "*Population code = '50', Population = 50, F = 0.5, Cr = 0.9, Asynchronous updating of solutions = No*". The results of DE are presented in Table 2. The minimum obtained is 5060.87 lb with no violations of the constraints. The solution time was 69.098 seconds for 20,000 iterations of the algorithm.

**Particle Swarm Optimization**

The PSO variant used is the standard PSO with the following default program settings: "*Population code = '20', Population = 20, C1 = 2, C2 = 2, W0 = 0.8, Gamma = 0.4, Elitism = Yes.*" The results are presented in Table 2.The results of PSO are also presented in Table 2. The minimum obtained is 5080.57 lb with a very small violation of the displacement constraint. The solution time was 73.073 s for 20,000 iterations of the algorithm, which is the largest time required by any algorithm tested.

Fig. 2 shows the convergence histories for both DE and PSO metaheuristic algorithms. We see that PSO converges faster during the first runs, but then DE takes the lead after around 6000 iterations. The total number of iterations for each algorithm is 20,000 but this graph has been limited to the first 10,000 iterations for illustration purposes. After 10,000 iterations, DE had reached an objective value of 5062.67 (vs 5060.87 in the end), and PSO had already reached the same optimum as the one after 20,000 iterations (5080.57).

**Table 2:** Optimization results with all Optimization Algorithms

| | Solver algorithms | | xlOptimizer algorithms | |
|---|---|---|---|---|
| | **Single point GRG** | **Multistart GRG** | **DE** | **PSO** |
| A1 (in$^2$) | 30.73067 | 30.52222 | 30.55354 | 32.28475 |
| A2 (in$^2$) | 0.10000 | 0.10000 | 0.10000 | 0.10000 |
| A3 (in$^2$) | 23.94943 | 23.20055 | 23.17546 | 23.54453 |
| A4 (in$^2$) | 14.72451 | 15.22289 | 15.24921 | 14.68662 |
| A5 (in$^2$) | 0.10000 | 0.10000 | 0.10000 | 0.10000 |
| A6 (in$^2$) | 0.10000 | 0.55103 | 0.55361 | 0.10000 |
| A7 (in$^2$) | 8.53892 | 7.45715 | 7.45930 | 8.52972 |
| A8 (in$^2$) | 20.95103 | 21.03627 | 21.02740 | 20.33665 |
| A9 (in$^2$) | 20.83663 | 21.52784 | 21.51037 | 20.75112 |
| A10 (in$^2$) | 0.10000 | 0.10000 | 0.10000 | 0.10000 |
| Weight (lb) | 5076.67 | 5060.84 | 5060.87 | 5080.57 |
| Max Stress (ksi) | 20.36570 | **25.00002** | 24.99954 | 20.43916 |
| Max Displ. (in) | **2.00000$^+$** | **2.00001** | 2.00000$^-$ | **2.00000$^+$** |
| Solution time (s) | 0.797 | 11.5 | 69.098 | 73.073 |

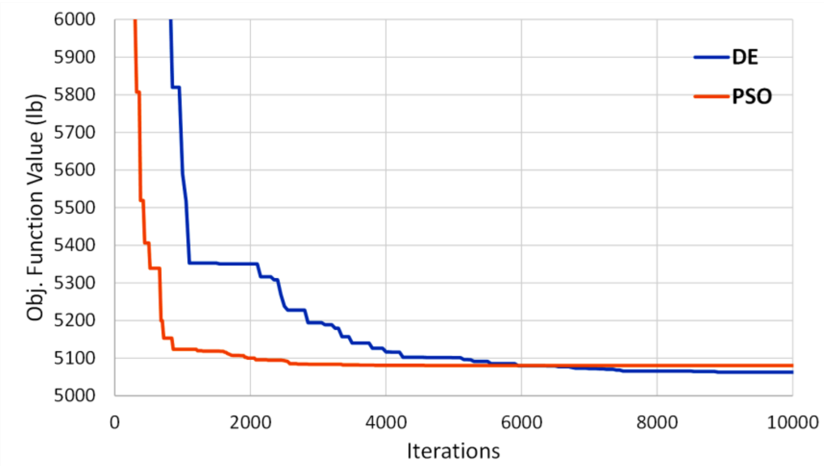


**Fig. 2:** The convergence histories for the DE and PSO algorithms

## 5 Conclusion

Spreadsheets have become very powerful and popular tools in the engineering community as they are easy to use, provide excellent results, and do not require any coding skills. Large engineering firms rely on spreadsheet calculations for the design of elements of various projects. Traditionally, the design was done with trial and error, where the engineer would try different sets of parameters before ending up with an acceptable good solution, while only expert programmers could have access to optimization algorithms. The addition of Solver and the availability of easy-to-use and powerful external tools such as xl Optimizer empowers the engineer who is now able to design elements optimally, without having to be an expert in optimization procedures or coding. The Solver is a robust and reliable tool, available in every Excel installation, offering two main optimization algorithms (a mathematical and an evolutionary one), while xl Optimizer goes far beyond, offering a host of metaheuristic optimization algorithms for a wide range of applications and the ability to deal with multi-objective problems. The addition of such tools gives tremendous power to Spreadsheet calculations. In this work, we showed that both Solver and xl Optimizer can be used to optimize 2D truss structures efficiently, using either mathematical or metaheuristic algorithms.

The next step in our research is applying the spreadsheet-based optimization techniques to larger scale problems Papadrakakis et al., (2001), 3D truss problems, frame problems, and multi-objective structural problems, while exploring additional metaheuristics, such as GA, EPSO, Simulated Annealing, Artificial Bee colony, and others.

## Acknowledgement

The support of Qatar University is acknowledged, through the Student Grant entitled “Design Optimization of Truss Structures using Swarm Intelligence Methods” (QUST-2-CENG-2022-681).

## References

Bhatti, M. A. (2005). "Structural Design Optimization using the MS Excel Solver and Computer Algebra Systems." in *Structures Congress*, 2005, 1-9. DOI: https://doi.org/10.1061/40753(171)83.

Charalampakis, A. E. & Dimou, C. K. (2010). "Identification of Bouc–Wen hysteretic systems using particle swarm optimization." *Computers & Structures*, *88*(21), 1197-1205. DOI: https://doi.org/10.1016/j.compstruc.2010.06.009.

Charalampakis, A. E. & Tsiatas, G. C. (2019). "Critical Evaluation of Metaheuristic Algorithms for Weight Minimization of Truss Structures." *Frontiers in Built Environment*, *5*. DOI: http://doi.org/10.3389/fbuil.2019.00113.

Christy, C. T. (2006). *Engineering with the spreadsheet: Structural engineering templates using excel*. DOI: http://doi.org/10.1061/9780784408278.

El-Sayed, M. E. M. & Jang, T. S. (1994). "Structural optimization using unconstrained nonlinear goal programming algorithm." *Comput. Struct.*, *52*(4), 723-727. DOI: https://doi.org/10.1016/0045-7949(94)90353-0.

Galante, M. (1992). "Structures optimization by a simple genetic algorithm." *Numerical Methods In Engineering And Applied Sciences*, Barcelona, Spain, 862-870.

Georgioudakis, M. & Plevris, V. (2018). "A Combined Modal Correlation Criterion for Structural Damage Identification with Noisy Modal Data." *Advances in Civil Engineering*, 2018(3183067), 20. DOI: https://doi.org/10.1155/2018/3183067.

___. (2020a). "A comparative study of differential evolution variants in constrained structural optimization." *Frontiers in Built Environment*, *6*(102), 1-14. DOI: https://doi.org/10.3389/fbuil.2020.00102.

___. (2020b). "On the Performance of Differential Evolution Variants in Constrained Structural Optimization." *Procedia Manufacturing*, *44*, 371-378. DOI: https://doi.org/10.1016/j.promfg.2020.02.281.

Ghasemi, M. R., Hinton, E. & Wood, R. D. (1997). "Optimization of trusses using genetic algorithms for discrete and continuous variables." *Eng. Computations, 16*(3), 272-303. DOI: https://doi.org/10.1108/02644409910266403.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning*, Addison-Wesley Longman Publishing Co., Boston, Massachusetts.

Haftka, R. T. & Gürdal, Z. (1992). *Elements of Structural Optimization*, Kluwer Academic Publishers.

Kao, C. Y., Hung, S. L. & Setiawan, B. (2020). "Two Strategies to Improve the Differential Evolution Algorithm for Optimizing Design of Truss Structures." *Advances in Civil Engineering*, 2020, 8741862. DOI: https://doi.org/10.1155/2020/8741862.

Kennedy, J. & Eberhart, R. (1995). "Particle swarm optimization." *IEEE International Conference on Neural Networks*, Piscataway, NJ, USA, 1942–1948.

Lagaros, N. D., Plevris, V. & Kallioras, N. A. (2022). "The Mosaic of Metaheuristic Algorithms in Structural Optimization." *Archives of Computational Methods in Engineering*. DOI: https://doi.org/10.1007/s11831-022-09773-0.

Lagaros, N. D., Plevris, V. & Papadrakakis, M. (2005). "Multi-objective Design Optimization Using Cascade Evolutionary Computations." *Comput. Methods Appl. Mech. Engrg.*, *194*(30-33), 3496-3515. DOI: https://doi.org/10.1016/j.cma.2004.12.029.

Memari, A. M. & Fuladgar, A. (1994). "Minimum Weight Design of Trusses by BEHSAZ Program." *Proceedings of 2nd International Conference on Computational Structures Technology*, Athens, Greece, August 30 - September 1.

Ozcep, F. (2010). "SoilEngineering: A Microsoft Excel® spreadsheet© program for geotechnical and geophysical analysis of soils." *Computers and Geosciences*, *36*(10), 1355-1361. DOI: https://doi.org/10.1016/j.cageo.2010.01.015.

Pandit, A. (2015). *Water engineering with the spreadsheet: A workbook for water resources calculations using excel*. DOI: https://doi.org/10.1061/9780784414040.

Papadrakakis, M., Lagaros, N. D. & Plevris, V. (2002). "Multi-objective Optimization of Skeletal Structures under Static and Seismic Loading Conditions." *Engrg. Optim.*, *34*(6), 645-669. DOI: http://dx.doi.org/10.1080/03052150215716.

Papadrakakis, M. et al. (2001). "Large Scale Structural Optimization: Computational Methods and Optimization Algorithms." *Archives of Computational Methods in Engineering*, *8*(3), 239-301. DOI: http://dx.doi.org/10.1007/BF02736645.

Perez, R. E. & Behdinan, K. (2007). "Particle Swarm Optimization in Structural Design." in *Swarm Intelligence: Focus on Ant and Particle Swarm Optimization*, F. T. S. Chan and M. K. Tiwari, eds., Itech Education and Publishing, Vienna, Austria, 373-394.

Plevris, V. (2009). "Innovative computational techniques for the optimum structural design considering uncertainties." National Technical University of Athens, Athens, Greece.

Plevris, V., Batavanis, A. & Papadrakakis, M. (2011). "Optimum design of steel structures with the Particle Swarm Optimization method based on EC3." *3rd International Conference on Computational Methods in Structural Dynamics and Earthquake Engineering (COMPDYN 2011)*, Corfu, Greece, 25-28 May 2011.

Plevris, V. & Papadrakakis, M. (2011). "A Hybrid Particle Swarm – Gradient Algorithm for Global Structural Optimization." *Comput.-Aided Civ. Infrastruct. Eng.*, *26*(1), 48-68. DOI: https://doi.org/10.1111/j.1467-8667.2010.00664.x.

Rady, M. & Mahfouz, S. Y. (2022). "Effects of Concrete Grades and Column Spacings on the Optimal Design of Reinforced Concrete Buildings." *Materials*, *15*(12), 4290. DOI: https://doi.org/10.3390/ma15124290.

Shahnam, N. (2003). "Utilization Of Excel In Solving Structural Analysis Problems." Nashville, Tennessee, 2003/06/22. DOI: https://doi.org/10.18260/1-2--12122.

Storn, R. & Price, K. (1995). "Differential Evolution - A Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous Spaces." Berkeley, CA.

**Cite as**: Plevris V., Elias I.H.M. & Charalampakis A.E., "Optimum Design of Plane Trusses Using Mathematical and Metaheuristic Algorithms on a Spreadsheet", *The 2nd International Conference on Civil Infrastructure and Construction (CIC 2023)*, Doha, Qatar, 5-8 February 2023, DOI: https://doi.org/10.29117/cic.2023.0109