Applied Mathematics E-Notes, 24(2024), 118-130 © ISSN 1607-2510
Available free at mirror sites of http://www.math.nthu.edu.tw/∼amen/

Resolving Issues Related To The Overlapping Of Geometric Figures Through The Utilization Of A Hybrid Computational Methodology*

Jamal Gsim†, Soukaina Regragui‡, Mohamed Zeriab Es-Sadek§, Mourad Taha Janan¶

Received 12 June 2023

Abstract

This paper deals with an industrial problem, consisting of generating objects randomly in a container without overlapping each other, this type of problem is an integral part of the so-called space layout problems that implement design optimization. However, the main idea is always the same: given a set of components and a container, layout optimization consists in finding all the positioning variables of the components to minimize certain objectives, while respecting certain constraints. In this article a new method is adopted which consists in modelling this problem to an optimization problem and solving it by using a hybrid approach, based on the coupling between a genetic algorithm and the projected gradient method, which allows to efficient generate objects (in our case study: generate different cylinders randomly) in a container without touching each other.

1 Introduction

In software development, randomly generating objects within a space is a frequent challenge. However, this approach can lead to significant overlap between these objects. This paper proposes a novel method to address this issue.

Our approach formulates the problem as an optimization task. We then tackle it using a hybrid approach that combines a genetic algorithm with the projected gradient method [1]. While genetic algorithms excel at finding solutions in complex problems (non-linear and non-convex), they might not always guarantee the absolute best solution, especially in very large search spaces. To address this limitation, the projected gradient method is incorporated. This deterministic method, however, is limited to finding local optima in non-convex problems.

2 Modelling

2.1 General Case

Considering two objects, A and B, initially created without overlap as depicted in Figure 1, we can determine their overlap by calculating the distance between them (denoted by d). A value of d equal to zero indicates complete overlap between the objects.

The distance between two points, $x = (t_1, t_2, t_3)$ in A and $y = (t_4, t_5, t_6)$ in B, can be calculated using the following formula:

$$d = \sqrt{(t_1 - t_4)^2 + (t_2 - t_5)^2 + (t_3 - t_6)^2}.$$

*Mathematics Subject Classifications: 20F05, 20F10, 20F55, 68Q42.
†Department of Mathematics and Computer Science, ENSAM, Mohammed V University in Rabat, Morocco
‡Laboratory of Applied Mechanics and Technologies, ENSAM, Mohammed V University in Rabat, Morocco
§Department of Mathematics and Computer Science, ENSAM, Mohammed V University in Rabat, Morocco
¶Laboratory of Applied Mechanics and Technologies, ENSAM, Mohammed V University in Rabat, Morocco

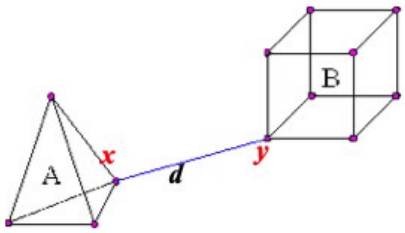


Figure 1: Nearest distance between two objects.

In the context of distinct objects A and B (Figure 1), the separation between them can be quantified as the minimum distance between any point within set A, denoted by $x = (t_1, t_2, t_3) \in A$, and any point within set B, denoted by $y = (t_4, t_5, t_6) \in B$. Mathematically, this distance is minimized subject to the constraints: $x \in A$ and $y \in B$:

$$\begin{cases} \min \sqrt{(t_1 - t_4)^2 + (t_2 - t_5)^2 + (t_3 - t_6)^2} \\ \text{subject:} \\ (t_1, t_2, t_3) \in A, \\ (t_4, t_5, t_6) \in B. \end{cases}$$

2.2 Case of Cylinders

Considering cylinders A and B (Figure 2), the constraint that $(t_1, t_2, t_3) \in A$ and $(t_4, t_5, t_6) \in B$ lies within their respective volumes, satisfying the equations that define cylinders A and B.

As established, a cylinder can be generally represented by a 7-dimensional vector

$$X = (x_1, x_2, x_3, x_4, x_5, x_6, x_7) \in \mathbb{R}^7.$$

Here, $O_1(x_1, x_2, x_3)$ and $O_2(x_4, x_5, x_6)$ denote the centers of the circular bases, and x_7 represents the radius. The equations of the cylinder's axis can then be derived from these elements:

$$\begin{cases} x_6 t_2 - x_5 t_3 + x_5 x_3 - x_6 x_2 = 0, \\ x_6 t_1 - x_4 t_3 + x_4 x_3 - x_6 x_1 = 0. \end{cases}$$

The center-to-center distance between the cylinders can be expressed by the following equation:

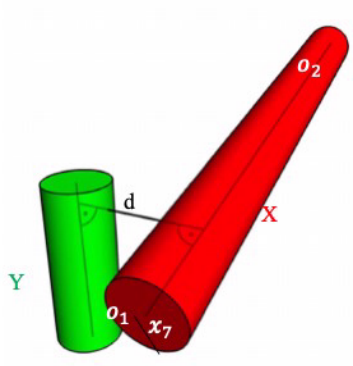


Figure 2: Distance between two cylinders.

$$\begin{cases} \min \ \sqrt{(t_1-t_4)^2+(t_2-t_5)^2+(t_3-t_6)^2} \\ \text{subject:} \\ x_6t_2 - x_5t_3 + x_5x_3 - x_6x_2 = 0, \\ x_6t_1 - x_4t_3 + x_4x_3 - x_6x_1 = 0, \\ y_6t_5 - y_5t_6 + y_5y_3 - y_6y_2 = 0, \\ y_6t_4 - y_4t_6 + y_4y_3 - y_6y_1 = 0. \end{cases} \tag{1}$$

Overlap exists between the two objects if the distance separating them $d \leq x_7 + y_7$, otherwise, there is no overlap.

3 Resolution Method

Problem (1) presents itself as a global optimization challenge. Deterministic algorithms, such as the projected gradient algorithm or the interior point method [2], offer solutions, but they can only guarantee a local optimum, especially for non-linear and non-convex problems like the one at hand.

Evolutionary algorithms provide another avenue for tackling this type of problem. Their effectiveness in handling non-linear and non-convex optimization scenarios has been established. However, they may struggle to converge on a satisfactory solution when dealing with a vast search space.

Recognizing these limitations, researchers have explored combining deterministic and evolutionary algorithms to capitalize on their respective strengths. This study proposes a new hybrid algorithm that integrates the projected gradient algorithm with a genetic algorithm to address Problem (1).

First, we will provide a concise review of both individual methods. Subsequently, we will unveil the proposed hybrid approach, designed to leverage the strengths of each technique to achieve an optimal solution.

3.1 Projected Gradient Algorithm

Rosen [3] proposed a technique for adapting optimization algorithms designed for unconstrained problems to handle problems with constraints. This method involves projecting the motion onto the boundary of the feasible region during each iteration, ensuring that the resulting solution remains within the valid space. To lay the foundation for the projected gradient method [4], a comprehensive analysis of this approach is necessary.

$$\begin{cases} \min \ f(x) \\ \text{subject:} \\ g_i(x) \leq 0 \quad i \in I_1 = \{1, 2, \ldots, p\}, \\ g_i(x) = 0 \quad i \in I_2 = \{p+1, \ldots, m\}. \end{cases} \tag{2}$$

Denoting by I_0 the set of constraints that are saturated at the specific point x:

$$I_0(x) = \{i \in I_1 \ : \ g_i(x) = 0\} \cup I_2 = \{i_1, \ldots, i_q\}.$$

We note:

$$g_0(x) = \left[g_{i_1}, \ldots g_{i_q}\right]^t.$$

We denote the Jacobian of function g_0 evaluated at point x by $J(x)$. The projection matrix we will employ has the following form:

$$P_0 = I - J^t.[J.J^t]^{-1}.J.$$

The following algorithm outlines the sequential steps involved in performing the Projected Gradient technique:

Step 1. Choose a starting point x^0

Step 2. While (stop condition not verified)
x^k the current point
Determine $J = J(x^k)$

Step 3. Calculate the projection matrix P_0

$\bar{y} = -P_0.\nabla f(x^k)$

$u = -[J.J^t]^{-1}.J.\nabla f(x^k)$

$u_i = \min_{j=1,\ldots,n} \{u_j\}$

Step 4. If $(\bar{y} = 0)$

If $(u_i < 0)$

Calculate $A_0^{'}$ the matrix obtained by removing row i from A_0, then calculating $P_0^{'}$

$\bar{y} = -P_0^{'}.\nabla f(x^k)$

Else

x^k satisfies the conditions of KKT

It should be recalled that the KKT (Karush-Kuhn-Tucker) conditions are a set of optimality conditions that must be satisfied by any feasible solution to a constrained optimization problem. These conditions provide the necessary conditions for a point to be an optimal solution when inequality and equality constraints are satisfied.

3.2 Genetic Algorithm

Pioneered by John Holland at the University of Michigan in the 1970s, the genetic algorithm (GA) is an optimization technique inspired by natural selection and genetic inheritance [5, 6]. By mimicking the process of evolution, the GA iteratively refines solutions to complex optimization problems. Leveraging principles like "survival of the fittest" and crossover of genetic material, the GA efficiently explores the search space, leading to progressively better solutions across generations. This makes it a powerful tool for tackling diverse optimization challenges in numerous fields. Genetic algorithms (GAs) are a class of optimization techniques recognized for their high robustness, particularly well-suited for addressing problems with challenging features. They excel in scenarios where:

1. The initial solution is not readily available.
2. Variables encompass diverse types, including both continuous and discrete values.
3. The optimization criterion relies on an external computational process rather than a well-defined mathematical function.

This robustness stems from GAs' proficiency in exploring intricate search spaces. This allows them to effectively handle situations with non-intuitive initializations and diverse variable types. Furthermore, their adaptability to "black-box" optimization problems, where the optimization criterion is evaluated through computationally expensive simulations or external processes, positions them as a valuable tool across a wide spectrum of real-world applications. In essence, GAs provides a powerful approach for optimizing problems characterized by varying complexities and uncertainties, making them a versatile choice for tackling real-world optimization challenges.

The Figure 3 illustrates the core principle of GAs. The initial population, either randomly chosen by the designer or generated through another computational process, undergoes evaluation based on the constraints and objectives defined in the optimization problem. The algorithm terminates upon meeting the stopping criteria. Conversely, if these criteria are not met, genetic operators are applied to manipulate the population, generating a new population of improved individuals that satisfy the problem's requirements. Notably, the individuals in the $(n+1)$th generation are considered the "children" of the individuals in the nth generation, who are referred to as the "parents" [7, 8, 9].

3.3 Hybrid Algorithm

This article proposes a novel hybridization approach that combines a genetic algorithm with a deterministic algorithm [10]. This integration exploits the complementary strengths of both methods, enabling us to effectively leverage their individual advantages. Specifically, the hybridization offers the following benefits:

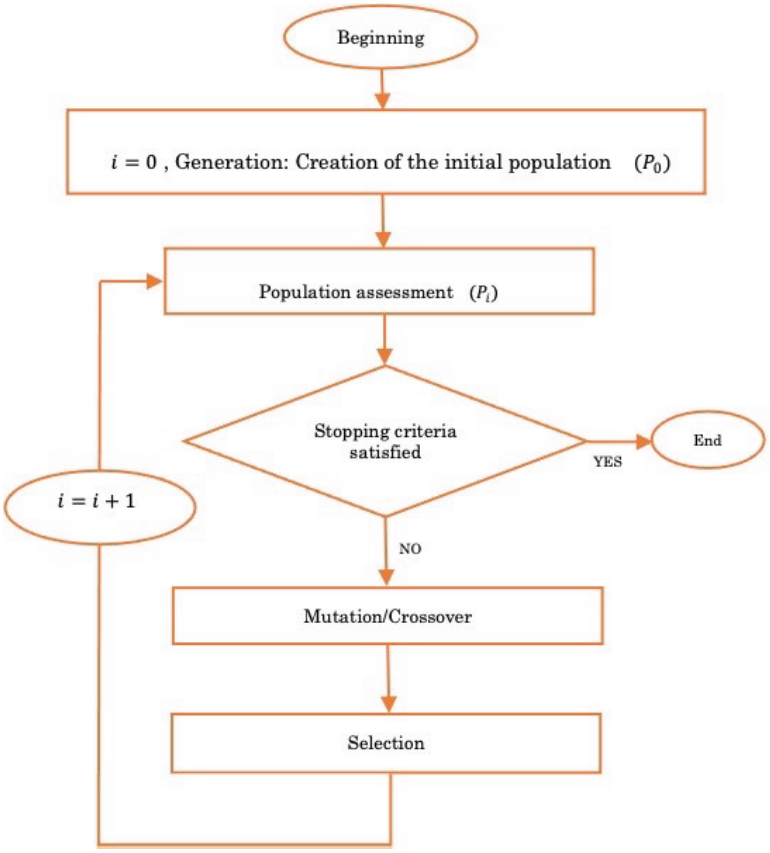


Figure 3: General operation of a genetic algorithm.

1. The genetic algorithm possesses a remarkable ability to efficiently identify promising regions containing the optimal solution. By incorporating genetic operators like crossover and mutation, it effectively navigates the search space, increasing the likelihood of finding such regions.

2. On the other hand, the deterministic algorithm excels in terms of convergence speed, accuracy, and computational efficiency when the initial solution is in the vicinity of the optimum. Its deterministic nature allows for a more precise and rapid exploration of the local region around a starting point, which is particularly advantageous for fine-tuning solutions near the global optimum.

By synergistically harnessing the strengths of both algorithms, our proposed hybrid approach aims to improve the overall optimization process. This provides a powerful and versatile tool for tackling complex optimization problems with diverse characteristics and challenges. Through empirical evaluations and illustrative examples, we demonstrate the effectiveness and superiority of our hybrid method in achieving high-quality solutions efficiently.

Our approach to achieving this objective involves customizing the genetic algorithm's operators. Specifically, we modify the core operators–generation, crossover, mutation, and selection–as follows:

Generation: We generate n random points. To guarantee their feasibility (i.e., adherence to problem constraints), we employ the correction operator introduced by Rosen [11]. This operator utilizes a projection matrix to rectify any infeasible points.

Crossover: To generate new solutions via crossover, we randomly select two parent points, X and Y, from

the population. We then create two offspring points, X_new and Y_new, using the following formulas:

$$X_{new} = rand * X + (1 - rand) * Y,$$

$$Y_{new} = (1 - rand) * X + rand * Y.$$

Here, "rand" represents a random coefficient between 0 and 1 that governs the blending of parental characteristics. To guarantee that the newly generated points remain within the feasible region and comply with problem constraints, we apply the projected gradient correction operator. This step ensures the offspring adhere to the constraints and avoid violating any boundary conditions set by the problem.

Mutation: For each point X, we utilize the Projected Gradient method with X as the starting point. This method guides the optimization process towards the feasible region while respecting the constraints. This ensures the updated solution remains within the domain defined by the constraints, allowing exploration of regions that satisfy both the objective function and the constraints.

Selection: After applying crossover and mutation, we meticulously select the n most promising individuals from the population based on their performance. We favor individuals that have undergone advantageous genetic variations and improvements during the evolutionary process. These chosen individuals constitute the next generation, contributing to the ongoing population refinement towards superior solutions.

3.4 Proof of Convergence

Within each iteration denoted by k, we define the population of candidate solutions as P_k. Each solution within this population is represented by X_i^k, where the indice k indicates the iteration and i denotes the specific solution's index within the population.
Furthermore, we introduce the concept of error at iteration k, denoted by e_k. This error term quantifies the discrepancy between the optimal solution X^* and the best solution currently identified within the population (P_k). Mathematically, the error is defined as the minimum value of the distance function applied to all solutions (X_i^k) in the population P_k, where the distance function calculates the distance between each solution and the optimal solution X^*:

$$e_k = \min_{X_i^k \in P_k} ||X_i^k - X^*||.$$

We now proceed to analyze the iterative process of the algorithm, specifically focusing on the transition from iteration k to $k + 1$. During this transition, the following key step occurs:

$$e_{k+1} = \min_{X_i^{k+1} \in P_{k+1}} ||X_i^{k+1} - X^*||.$$

This equation represents the minimization of the distance between each point X_i^{k+1} in the updated population P_{k+1} and the optimal solution X^*.

It is important to note the modifications implemented in the genetic algorithm operators to achieve this:

Generation: Random points are initially generated and then corrected to ensure they satisfy the problem constraints and remain within the feasible region.

Crossover: Offspring points are generated by linearly combining parent points. These offspring are then corrected to remain within the feasible region.

Mutation: The projected gradient method leverages the best points from the previous population as starting points to explore local optima.

Stopping Criteria: The hybrid algorithm terminates if the change in the objective function between subsequent iterations falls below a predefined threshold, indicating insufficient progress.

3.4.1 Convergence Argument

We analyze the error reduction between consecutive iterations. Let e_k denote the error associated with solution X_i^k at iteration k, and X^* represents the optimal solution. The inequality below holds:

$$e_{k+1} \leq \min_{X_i^{k+1} \in P_{k+1}} ||X_i^{k+1} - X^*|| \leq \min_{X_i^k \in P_k} ||X_i^k - X^*|| = e_k.$$

This guarantees that the error either decreases ($e_{k+1} < e_k$) or remains constant ($e_{k+1} = e_k$) between iterations, indicating convergence towards the optimal solution X^*. The algorithm employs a hybrid approach, leveraging genetic algorithm components for efficient exploration of the search space and deterministic components for refining solutions towards optimality.

3.4.2 Conclusion

The proposed hybrid algorithm is theoretically expected to converge based on the arguments presented. The modifications to the genetic algorithm operators, specifically those ensuring feasibility and constraint satisfaction, are hypothesized to contribute to the overall convergence of the algorithm. The incorporation of a selection operator that prioritizes promising individuals is likely to further enhance the refinement process. These features collectively suggest that the hybrid algorithm has the potential to be a powerful optimization tool.

However, it is important to acknowledge that the practical success of the hybrid algorithm may be contingent upon several factors. These factors include, but are not limited to, the careful tuning of hyperparameters, the specific characteristics of the optimization problem being addressed, and the underlying complexity of the optimization landscape.

4 Numerical Experiments

In order to demonstrate the effectiveness, efficiency, and robustness of the proposed algorithm, we have chosen the following problems as test cases.

4.1 Academic Problem

Let's consider the following optimization problem [12]:

$$\begin{cases} \min 168x_1x_2 + 3651.2x_1x_2x_3^{-1} + 40000x_4^{-1} \\ \text{Subject to:} \\ 1.0425x_1x_2^{-1} \leq 1 \\ 0.00035x_1x_2 \leq 1 \\ 1.25x_1^{-1}x_4 + 41.63x_1^{-1} \leq 1 \\ 40 \leq x_1 \leq 44 \\ 40 \leq x_2 \leq 45 \\ 0.1 \leq x_3 \leq 1.4 \end{cases} \tag{3}$$

Table 1 presents a comparative analysis of the results achieved by our proposed approach against those

Ref.	x^*	Solution
Present study	[40.6831 42.6887 68.8200 1.1083]	$4.2000e^{+05}$
[12]	[43.02 44.85 66.39 1.11]	$6.2337e^{+05}$
[13]	[43.08 44.99 66.41 1.10]	$4.6848e^{+05}$

Table 1: Results comparison for problem 3

reported in existing literature. It is clear that the hybrid algorithm is more efficient and give the best solution.

4.2 Optimal Design of a Speed Reducer

This study evaluates the performance of the proposed algorithm and compares it to the algorithm presented in [14], which was implemented using GAMS technology. This study aims to identify the optimal system parameters that minimize the weight of a speed reducer (Figure 4). The system is defined by seven design variables:

- x_1: Face width of the gear tooth.
- x_2: Teeth module (reciprocal of diametral pitch).
- x_3: Number of pinion teeth.
- x_4: Shaft length 1 (distance between bearings).
- x_5: Shaft length 2 (distance between bearings).
- x_6: Shaft diameter 1.
- x_7: Shaft diameter 2.

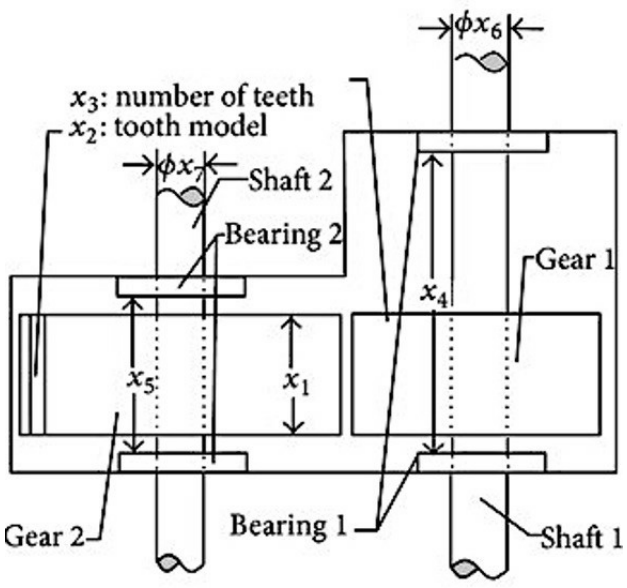


Figure 4: Golinski Speed reducer.

The mathematical model for weight minimization will be presented subsequently:

$$\begin{cases} \min 0.7854x_1x_2^2\left(3.3333x_3^2+14.933x_3-43.0934\right)-1.508x_1\left(x_6^2+x_7^2\right)+7.4777\left(x_6^3+x_7^3\right) \\ \quad +0.7854\left(x_4x_6^2+x_5x_7^2\right) \\ \text{Subject to:} \\ \frac{27}{x_1x_2^2x_3}-1\leq 0, \quad \frac{397.5}{x_1x_2^2x_3^2}-1\leq 0 \\ \frac{1.93x_4^3}{x_2x_3x_6^4}-1\leq 0, \quad \frac{1.93x_5^3}{x_2x_3x_7^4}-1\leq 0 \\ \frac{\sqrt{\left(\frac{745x_4}{x_2x_3}\right)^2+16900000}}{110x_6^3}-1\leq 0, \quad \frac{\sqrt{\left(\frac{745x_5}{x_2x_3}\right)^2+15750000}}{85x_7^3}-1\leq 0 \\ \frac{x_2x_3}{40}-1\leq 0, \quad \frac{5x_2}{x_1}-1\leq 0 \\ \frac{x_1}{12x_2}-1\leq 0, \quad \frac{1.5x_6+1.9}{x_4}-1\leq 0, \quad \frac{1.1x_7+1.9}{x_5}-1\leq 0 \\ 2.6\leq x_1\leq 3.6, \quad 0.7\leq x_2\leq 0.8, \quad 17\leq x_3\leq 28, \quad 7.3\leq x_4\leq 8.3 \\ 7.8\leq x_5\leq 8.3, \quad 2.9\leq x_6\leq 3.9, \quad 5.0\leq x_7\leq 5.5 \end{cases}$$

Table 2 presents a comparison of the results obtained using our proposed approach with those from existing literature for the same problem. This comparison underscores the feasibility and efficiency of our algorithm. Furthermore, the results demonstrate superior performance and efficiency of our approach compared to the previously established methods. These findings serve as strong evidence for the effectiveness and superiority of our proposed algorithm.

Ref.	x^*	Solution
Present study	[2.6 0.7 17.5605 7.5029 7.8 3.0191 5.1975]	2.581 e^{+03}
[15]	[3.5 0.7 17 7.3 7.8 3.35 5]	2.823 e^{+03}
[14]	[3.5 0.7 17 7.3 7.8 3.3502 5.2866]	2.996 e^{+03}

Table 2: Results comparison for the optimal design of a speed reducer

4.3 Pressure Vessel Optimization Problem

In this example, we will evaluate our hybrid algorithm on another optimization problem: the pressure vessel design problem (Figure 5). This problem aims to minimize the total cost, which includes the costs of welding and forming materials. The key decision variables in this optimization problem are:

- L: Length of the cylindrical segment of the vessel
- Ts: Thickness of the cylindrical shell
- R: Inner radius of the vessel
- T_h: Thickness of the spherical head

Assuming $x_1 = Ts$, $x_2 = T_h$, $x_3 = R$ and $x_4 = L$ as the design variables, the optimization problem is:

$$\begin{cases} \min 0.6224x_1x_3x_4+1.7781x_2x_3^2+3.1611x_1^2x_4+19.8621x_1^2x_3 \\ \text{Subject to:} \\ 0.0193x_3-x_1\leq 0 \\ 0.00954x_3-x_2\leq 0 \; x_4-240\leq 0 \\ 750\times 1728-\pi x_3^2x_4-43\pi x_3^3\leq 0 \\ x_1,x_2\in[1\times 0.0625,\ 99\times 0.0625] \\ x_3,x_4\in[10,\ 240] \end{cases}$$

Table 3 presents a comparison of the results obtained using our proposed approach with those from existing literature for the same problem.

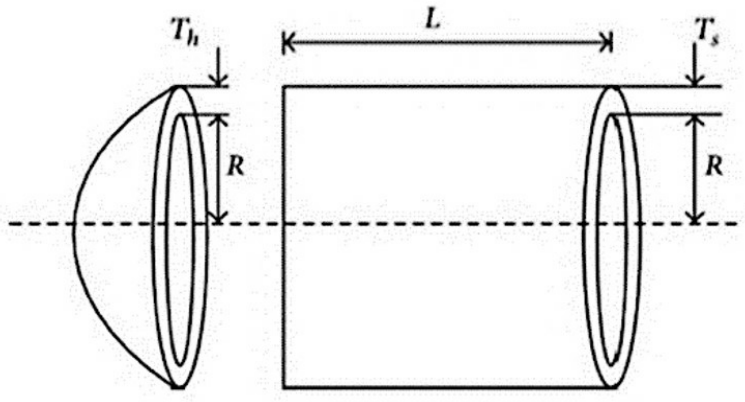


Figure 5: The pressure vessel design problem.

Ref.	x^*	Solution
Present study	[0.74, 0.375, 38.865, 221.18]	5772.5
[16]	[0.7683, 0.3797, 39.8096, 207.2250]	5868.76
[17]	[0.75, 0.375, 38.8601, 221.3654]	5850.383
[18]	[1.125, 0.625, 58.2901, 43.6927]	7199.35

Table 3: Results comparison for the pressure vessel design problem

4.4 Compression/Tension Spring Design Problem

In this work, we evaluate the performance of our proposed algorithm against established methods [19, 20] by applying it to the compression/tension spring design problem (Figure 6). This problem aims to minimize the weight of a spring while satisfying constraints such as shear stress, minimum deflection, limitations on the outer diameter, and surge frequency. The design variables considered are the number of active coils (P), the mean coil diameter (D), and the wire diameter.Assuming $x_1 = d$, $x_2 = D$ and $x_3 = P$, as the design

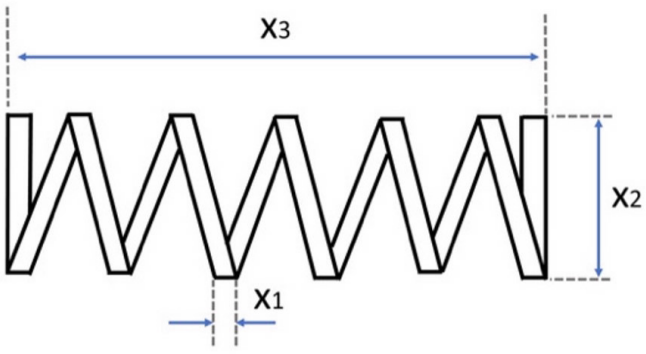


Figure 6: Schematic of the compression/tension spring design problem.

variables, the compression/tension spring design problem can be expressed as follows:

$$\begin{cases} \min\,(x_3{+}2)x_2{x_1}^2 \\ \text{Subject to:} \\ 1-\frac{x_2^3x_3}{71{,}785x_1^4}\le 0 \\ \frac{4x_2^2-x_1x_2}{12{,}566\left(x_2x_1^3-x_1^4\right)}+\frac{1}{5{,}108x_1^2}-1\le 0 \\ 1-\frac{140.45x_1}{x_2^2x_3}-x_2\le 0\ ,\ \frac{x_2+x_1}{1{,}5}\le 0 \\ x_1\in\ [0.05,\ 2.0] \\ x_2\in\ [0.25,\ 1.3] \\ x_3\in\ [2.0,\ 15.0] \end{cases}$$

Table 4 presents a comparison of the results obtained using our proposed approach with those from existing literature for the same problem. This study evaluates the performance of the hybrid algorithm through

Ref.	x^*	Solution
Present study	[0.051, 0.357, 11.140]	0.0122
[19]	[0.051796, 0.359283, 11.140516]	0.012667
[20]	[0.051583, 0.354190, 11.438675]	0.012665

Table 4: Results comparison for the compression/tension spring design problem

benchmark design optimization examples. These results demonstrate the algorithm's effectiveness in tackling practical optimization problems.

5 Results of the Overlapping Problem

The generation of non-overlapping objects in 3D space presents a significant challenge in various practical applications. Software-based generation often introduces the risk of object overlap, leading to inaccurate and unreliable results. To address this issue, we propose a novel hybrid algorithm that leverages the strengths of different optimization techniques. This algorithm solve the optimization problem (1) for each new cylinder with the other existing cylinders to ensure the generation of non-overlapping cylinders within a defined cuboid.

Our hybrid approach aims to deliver robust and meaningful outcomes by efficiently handling the complexities of spatial arrangements. To evaluate its efficacy, we conducted experiments using a 3D geometry populated with randomly distributed cylinders. This scenario replicates a realistic setting to assess the performance of our method.

During the testing phase, cylinders with varying diameters and lengths were generated. Each newly created cylinder was meticulously positioned within a cuboid with predefined height, width, and length. This deliberate placement aimed to prevent overlap between the generated cylinders themselves, as well as between the cylinders and the cuboid walls.

The implementation of our hybrid algorithm with the inclusion of non-overlapping constraints successfully addressed the challenge of random object generation in 3D space. The experimental results demonstrate the effectiveness of our approach in managing complex spatial arrangements, offering a reliable solution for practical applications.

Considering the total volume of the generated cylinders as V_{Cyl} and the volume of the brick as V_{Brick}, we aim to evaluate our algorithm across various cylinder fill rates within the brick. The fill rate, denoted by $T_{Filling}$, is defined as the ratio of V_{Cyl} to V_{Brick} ($T_{Filling} = \frac{V_{Cyl}}{V_{Brick}}$). Specifically, we will test the algorithm at fill rates of 1%, 5% and 10%.

This section presents the numerical results and visualizations obtained by applying the hybrid algorithm to the specific problem at hand. We evaluate the performance and efficiency of the proposed method in

addressing the overlapping issue. A cuboid with dimensions $(100, 60, 200)$ is employed as a case study.

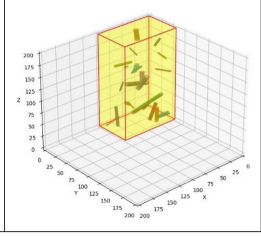
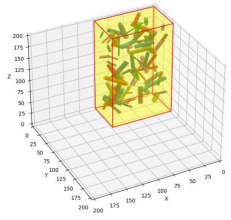
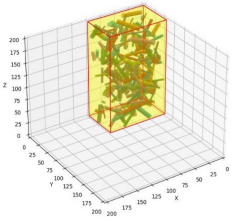
Filling rate	1%	5%	10%
Number of cylinders	22	84	189
Graphical representation			

Table 5: Results of the overlapping problem.

6 Conclusion

This study demonstrates the effectiveness of the proposed hybrid algorithm in generating non-overlapping, distinct cylinders across all three filling rate scenarios. This successful approach addresses the challenge of creating cylinders with varying diameters and lengths that do not intersect. The promising results obtained here serve as a foundation for further research endeavors aimed at exploring and extending this approach to a broader range of shapes and geometries.

References

[1] J. B. Rosen, The gradient projection method for nonlinear programming Part 1: Linear constraints, SIAM J. Appl. Math., 8(1960), 181–217.

[2] M. Z. Es-Sadek, Contribution à l'optimisation Globale: Approche Déterministe et Stochastique et Application, Mathématiques Générales [math.GM]. INSA de Rouen, Université Mohammed V-Agdal, 2009.

[3] D.-Z. Du, F. Wu and X. -S. Zhang, On Rosen's gradient projection methods, Ann. Oper. Res., 24(1990), 11–28.

[4] J. B. Rosen, The gradient projection method for nonlinear programming Part 2: Nonlinear constraints, SIAM J. Appl. Math., 9(1961), 514–553.

[5] J. H. Holland, Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence, MIT Press, Cambridge, MA, USA, 1992.

[6] D. E. Goldberg, Genetic Algorithms in Search Optimization and Machine Learning, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[7] J. Zhang, H. S. H. Chung and J. Zhong, Adaptive crossover and mutation in genetic algorithms based on clustering technique, Proceedings of the 7th annual conference on Genetic and evolutionary computation, 2005, 1577–1578.

[8] T. P. Hong, H. S. Wang, W.-Y. Lin and W.-Y. Lee, Evolution of appropriate crossover and mutation operators in a genetic process, Applied Intelligence, 16(2002), 7–17.

[9] J. Jilkova and Z. Raida, Influence of Multiple Crossover and Mutation to the Convergence of Genetic Optimization, In MIKON 2008, XVII International Conference on Microwaves, Radar and Wireless Communications in Poland, 2008.

[10] Y. Belkourchia, L. Azrar and M. Z. Es-Sadek, Hybrid optimization procedure applied to optimal location finding for piezoelectric actuators and sensors for active vibration control, Appl. Math. Model., 62(2018), 701–716.

[11] American Society of Mechanical Engineers, ANSI Y14.5, Dimensioning and tolerancing ANSI Y14.5. New York, NY; 1994.

[12] M. Rijckaert and X. Martens, Comparison of generalized geometric programming algorithms, J. Optim. Theory Appl., 26(1978), 205–241.

[13] S. Qu, K. Zhang and F. Wang, A global optimization using linear relaxation for generalized geometric programming, Eur. J. Oper. Res., 190(2008), 345–356.

[14] J. Golinski, An adaptive optimization system applied to machine synthesis, Mechanism and Machine Synthesis, 8(1973), 419–436.

[15] N. Andrei, Nonlinear optimization applications using the GAMS technology, Springer Optimization and Its Applications, 81(2013), 67–70.

[16] A. R. Hedar and M. Fukushima, Derivative-free filter simulated annealing method for constrained continuous global optimization, Journal of Global Optimization, 35(2006), 521–549.

[17] G. G. Dimopoulos, Mixed-Variable Engineering Optimization Based on Evolutionary and Social Metaphors, Computer Methods in Applied Mechanics and Engineering, 2007.

[18] O. Hasançebi, S. K. Azad and O. Hasançebi, An Efficient Metaheuristic Algorithm for Engineering Optimization: SOPT, International Journal of Optimization in Civil Engineering, 2012.

[19] L. C. Cagnina, S. C. Esquivel and C. A. Coello, Solving engineering optimization problems with the simple constrained particle swarm optimizer, Informatica, 32(2008), 319–326.

[20] O. Adekanmbi and P. Green, Conceptual Comparison of Population-Based Metaheuristics for Engineering Problems, The Scientific World Journal, 2015.