HEURISTIC ALGORITHMS USED IN MULTIDISCIPLINARY DESIGN OPTIMIZATION

A Graduate Project Report submitted to Manipal Academy of Higher Education in partial fulfilment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

In

Aeronautical Engineering

Submitted by

Sanjana Srivastava

Under the guidance of

Professor Manikandan M.
Assistant Professor – Senior Scale

DEPARTMENT OF AERONAUTICAL AND AUTOMOBILE ENGINEERING

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

AUGUST 2021

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

Manipal
31//08/2021

CERTIFICATE

This is to certify that the project titled **HEURISTIC ALGORITHMS USED IN MULTIDISCIPLINARY DESIGN OPTIMIZATION** is a record of the bonafide work done by **SANJANA SRIVASTAVA** (170933064) submitted in partial fulfilment of the requirements for the award of the Degree of Bachelor of technology (B Tech) in **AERONAUTICAL ENGINEERING** of Manipal Institute of Technology, Manipal, Karnataka, during the academic year 2020 -2021.

Mr. Manikandan M. *Assistant Professor – Senior Scale* *Aero & Auto, M.I.T, MANIPAL*	**Dr. Satish Shenoy B** *Prof & HOD, Aero & Auto* *M.I.T, MANIPAL*

ACKNOWLEDGMENTS

The work conducted in this project is submitted to Manipal Academy of Higher Education in partial fulfilment of the requirement for the award of the degree of Bachelor of Technology in Aeronautical Engineering under the department of Aeronautical and Automobile Engineering.
I would like to express my gratitude to the director of MIT Manipal, Dr D Srikanth Rao and Dr Satish Shenoy B, the head of my department for providing me with this opportunity to develop and hone my skills in the research domain and inspiring me to pursue research of my own in the future.
I convey my sincere gratitude to Professor Manikandan M for his expertise and invaluable guidance for the execution of the project and making of this report in all stages.
I would also like to convey my thanks to Ms Thara Reshma for her support and feedback with the evaluation of this report.
I am profoundly grateful to the Department of Aeronautical and Automobile Engineering at MIT Manipal, for offering me the proper infrastructure and necessary facilities and required to aid and complete this report successfully.

ABSTRACT

Multidisciplinary design optimization is an up-and-coming field of engineering and computational science, finding applications across nearly all streams of science and technology as well as sociology. The approach works better than traditional methods as it can accommodate the interdisciplinary interactions of the different spheres involved in an optimization problem and successfully incorporate these interdependencies in the computational process. Heuristic algorithms are being increasingly used in the solution of MDO problems due to their abilities of finding approximate solutions in much lesser complex and time-consuming problems as compared to traditional methods. This project entails development of the main objective function and aims to evaluate three different heuristic algorithms – particle swarm optimization, genetic algorithm, and simulated annealing – and provide a better insight into their usage in optimization with large numbers of parameters as well as constraints. Multiple iterations of each algorithm are carried out, tabulated, and compared for a deeper analysis of their usefulness and robustness. This project makes use of MATLAB software to code the algorithms and perform their iterations.

LIST OF TABLES

LIST OF FIGURES

Contents

CHAPTER 1
INTRODUCTION

1.1. Multidisciplinary design optimization (MDO)

Solving an engineering problem generally requires expertise in various aspects spanning across different branches of science and technology. Multidisciplinary design optimization (MDO) is a branch of engineering that allows the incorporation of all the relevant fields simultaneously into a singular design problem. MDO is considered a superior technique of optimization as it can account for the interactions between the various disciplines in computing the solution as opposed to individually optimizing each different aspect of the problem and then compiling the results. The concept's inception first began in the 1980s as a successor to the largely successful methods of structural optimization. Since then, MDO has evolved from utilizing gradient-based methods to more recent non gradient-based methods as well as broader techniques such as evolutionary algorithms, decomposition methods, memetic algorithms, and more. The branch has shown a wide range of applications in numerous fields such as aerospace and automobile engineering, naval architectures, electrical engineering, and aircraft design.

MDO comprises of formulation of a problem statement with proper selection of the given design features, relevant constraints, an objective function, and possibly mathematical or geometric models of the problem. The formulated problem is then solved using an optimization technique of choice, which may be linear, gradient-based, population-based, etc. This paper deals with the formulation of an objective function for an engineering problem – namely, to minimize the mass of an airship with the constraints of weight and energy balance – and to solve the same with the help of heuristic algorithms. The objective function is converted from a constrained problem to an unconstrained problem by using the penalty function method, which is a computational method that incurs heavy penalties as and when the constraints are being violated. The objective function is solved using three main heuristic algorithms – Particle Swarm Optimization (PSO), Genetic Algorithm (GA), and Simulated Annealing (SA) – and the results are further compared to determine the most accurate optimization method for the given problem.

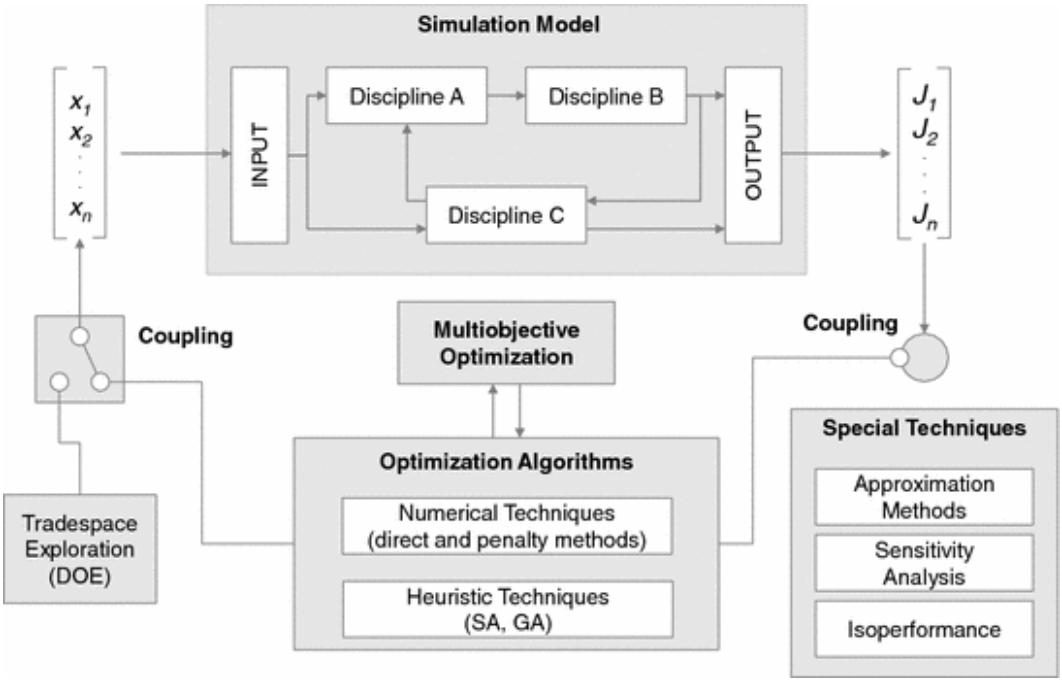


Figure 1: *Basic flowchart of MDO [1].*

1.2. Heuristic algorithms

Heuristic algorithms are newer algorithms that are used in place of traditional optimization methods for a faster and more efficient solution when traditional methods may be too slow or unable to find an exact solution. These algorithms involve a trade-off in aspects such as accuracy, precision, or optimality, for speed to ensure a faster solution for the same problem. A classic example of a heuristic algorithm is known as the Travelling Salesman Problem (TSP) first described by Jon Bentley. The problem is traditionally hard and time-consuming to solve traditionally, so the heuristic algorithm proposed searches for an approximately good solution rather that an optimum solution in a better and more practical amount of time by picking the next best step regardless of its impact on future steps.

Particle Swarm Optimization (PSO)

Particle swam optimization is an iterative computational method first introduced by Kennedy and Eberhart in 1995 [2]. The algorithm was computed as an inspiration from social behaviour, namely from the movement of a swarm of bees or a school of fish. PSO is also a derivate of evolutionary algorithms with ties to genetic algorithm. The algorithm takes a population (or a swarm) of particles and allows them to move in the sample space based on the governing formulae. The particles are moved in accordance with their own individual best value (pbest) as well as the global best value of the population (gbest). The process occurs iteratively till a satisfactory result is achieved.

PSO pseudo code for minimizing a function [3]:

1. A population array is initialised with random values of velocity and position in the n-dimensional sample space.

2. Beginning of loop:
3. The fitness of each particle is determined in accordance with the fitness function.
4. The fitness value of the particle is compared with its best value, pbest. If pbest > current value, then let pbest = current value.
5. After initial fitness evaluation, the best fitness value is determined an assigned to gbest.
6. The velocity and position of each particle is updated as per the general PSO equation:

$$v(id) = w.v(id) + c1.rand\big(p(id) - x(id)\big) + c2.Rand(p(gd) - x(id))$$

$$x(id) = x(id) + v(id)$$

Where,

d = dimension

$c1$ = self-adjustment weight

$c2$ = social-adjustment weight

w = inertia weight

7. Loop is exited after stopping condition is met (for example, maximum number of iterations).
8. End of loop.

Parameters c1 and c2 are randomly generated constants also known as acceleration coefficients. They are responsible for the magnitude values of forces that direct the particles towards the pbest and gbest values. The inertia weight, w, has shown high performance with relatively high starting values like 0.8 or 0.9, and then progressively lowered as iterations converge [2]. The selection of these parameters has significant effects on the outcome of the Particle Swarm Algorithm.

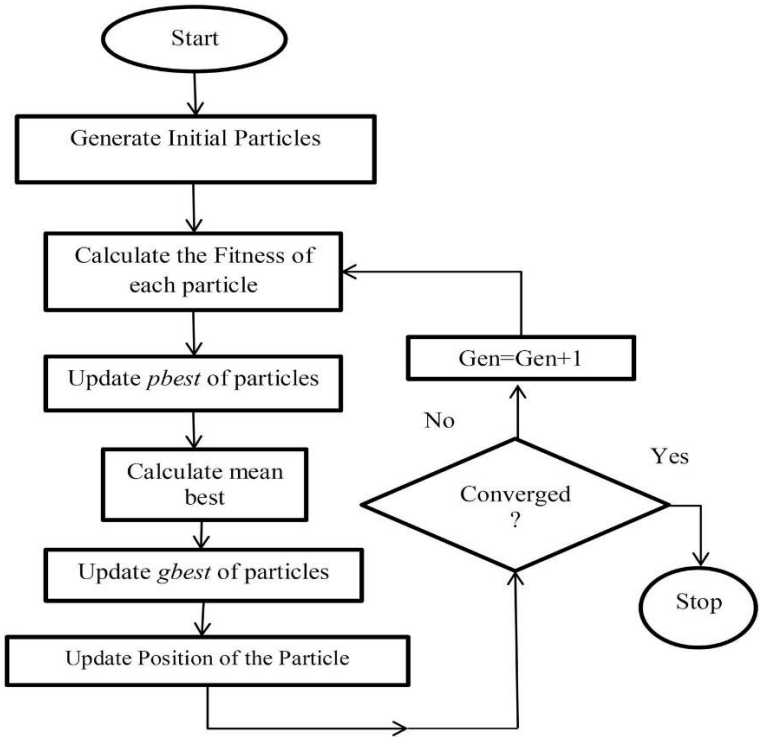


Figure 2: *Flowchart detailing PSO algorithm [4].*

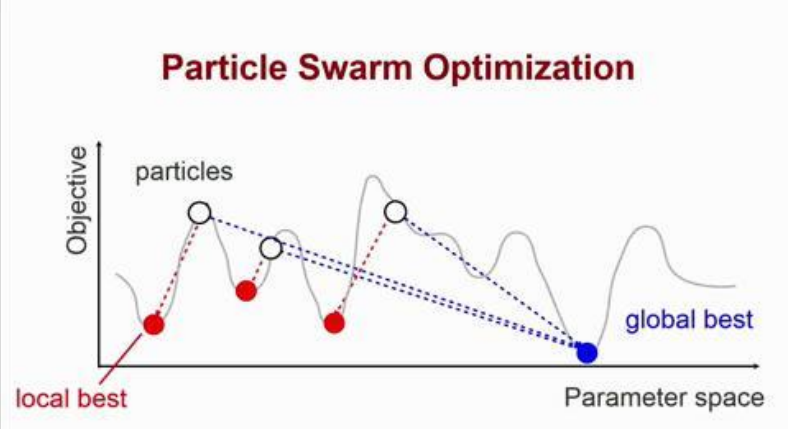


Figure 3: *Graphical representation of the working of PSO [5].*

Genetic Algorithm (GA)

Genetic algorithm is an evolutionary algorithm inspired by the biological phenomenon of Darwin's natural selection and survival of the fittest, incorporating biological parameters of updation like selection, mutation, and crossovers. The basic framework of genetic algorithm is to first randomly initialise a population of chromosomes encoded upon a scheme like binary, hexadecimal, octal, etc. and compute the fitness of each chromosome. Two chromosomes with the best fitness values are chosen to be the 'parents' of the next generation, and a genetic operator (like single-point crossover) is applied to produce the new offspring. A mutation operator is then applied on the offspring to produce 'fitter' offspring for the next generation. This process is repeated till a new generation of chromosomes is formed.

GA Pseudo Code [6]:

1. An initial population of n chromosomes is created randomly within the sample space.
2. Beginning of loop:
3. Fitness value of each chromosome is computed according to the fitness function specified.
   while iteration < maximum number of iterations:
5. Select two chromosomes from initial population with best fitness value.
6. Apply the crossover operator on the selected chromosomes.
7. Apply the mutation operator on the new offspring.
8. Replace older generation with newer formed generation of offspring.
9. Return value of fittest chromosome.
10. End of while.
11. Loop is exited after stopping condition (like satisfactory fitness value or maximum number of generations) is met.
12. End of loop.

The four major defining parameters of the outcome of genetic algorithm are encoding scheme, selection, crossover, and mutation. It is necessary for the parameters to be chosen based upon type of problem and outcome expected. Heuristic operators may also be added in addition to genetic operators to improve the speed of the algorithm. GA has been widely used in a variety

of applications, most prominently - layout problems, network designs, encryption, video or image processing, and wireless networking [6].

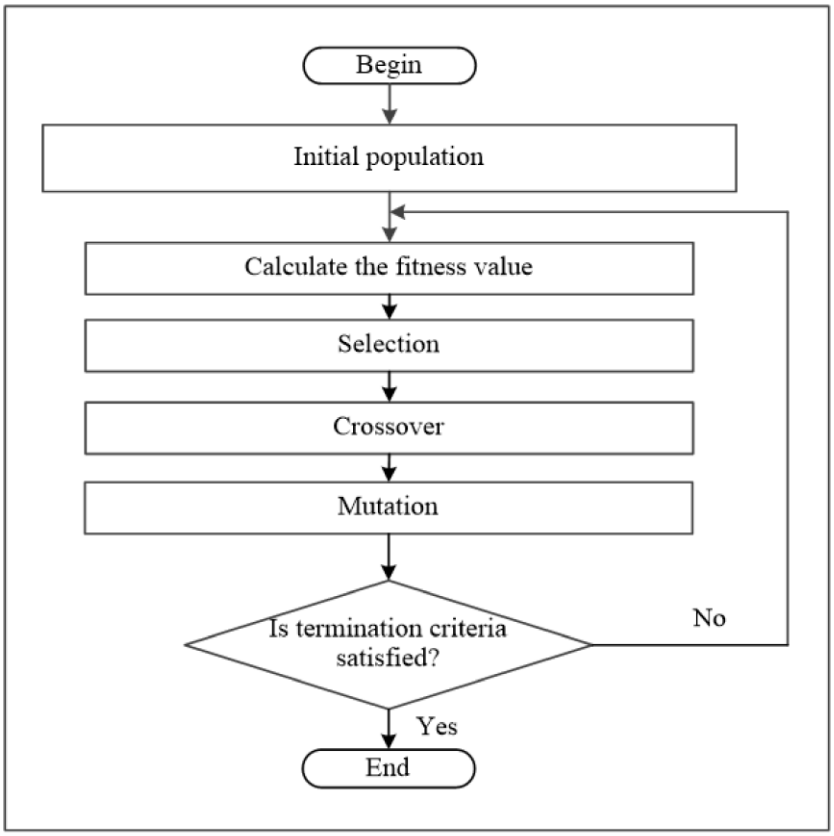


Figure 4: *Basic flowchart of Genetic Algorithm [7].*

Simulated Annealing (SA)

Simulated annealing is a heuristic optimization method that is used to find the global optimum of a function across a very large sample space which is generally discrete. It is derived from the chemical process of annealing, which is a thermodynamic process in which a material undergoes controlled heating and cooling to reduce its structural defects and increase its constituent crystal size. The basic principle of annealing is that the material at higher temperatures is weaker in structure and more susceptible to structural changes, which then become permanent as the material cools down and the molecular structure hardens. The primary equation used in simulated annealing is analogous to the thermodynamic equation that calculates the probability of increase in energy magnitude:

$$P(\Delta E) = e^{-\frac{\Delta E}{k*t}}$$

Where ‘k’ is Boltzmann’s constant and ‘t’ is any temperature.

SA Pseudo Code [8]:

1. First, an initial solution ‘s0’ at initial temperature ‘t0’ must be initialized.

2. A temperature reduction function, ‘alpha’, is created in accordance with any of the three main rules of temperature reduction:

i) Linear reduction: $t = t - \propto$

ii) Geometric reduction: $t = t * \propto$

iii) Slow-decrease: $t = \frac{t}{1+\beta * t}$, where ‘beta’ is an arbitrary constant.

3. Beginning of loop:

4. A solution is chosen from the map of N solutions in the neighbourhood of the old solution, and the difference in cost between this new solution and the initial solution is calculated.

5. If the difference calculated is greater than 0, the new solution is accepted. Else if the difference is less than 0, a random number from 0 to 1 is generated and accepted if the value is less than that calculated by the energy magnitude equation. The equation for SA is summed up as follows:

$$P = \begin{cases} 1 & if \Delta c \leq 0 \\ e^{-\Delta c / t} & if \Delta c > 0 \end{cases}$$

Where delta c is the change in temperature and ‘t’ represents the current temperature. ‘P’ represents the probability of accepting the new solution.

6. The iterations are repeated, and temperature is decreased according to the reduction factor ‘alpha’ till the stopping criteria is met – such as a final temperature or minimum difference between successive solutions.

7. End of loop.

Iterations of simulated annealing can also be performed by changing the initial solution set ‘s0’. Simulated annealing is a simple and easy to use algorithm in problems where other techniques may get stuck on local minimas or are not applicable to a wide range of solutions. However, one drawback to SA is that the algorithm requires finetuning of a large number of parameters.

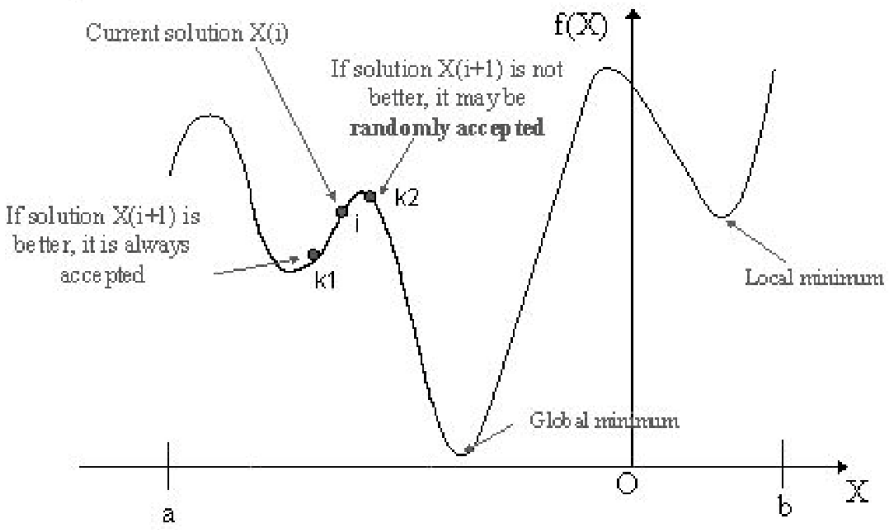


Figure 5: *Principal of Simulated Annealing [9].*

CHAPTER 2
LITERATURE REVIEW

2.1. General discussion

This paper deals with the comparison of different heuristic algorithms to find the most optimum algorithm that solves the given objective function. The goal of the algorithms is to minimize the mass of the airship while keeping in mind the constraints of weight-balance and energy-balance to maintain the stability of the airship. Multidisciplinary design optimization problems generally take a long time to compute results, and a lot of current research is based on improving speed and robustness of these problems by experimenting with different type of optimization algorithms. Another problem faced in MDO is that as of the current research, no optimization technique is able to guarantee a global optimum solution. Some algorithms, such as those based on gradients, have shown to result in local optimum solutions with high efficiencies, but are still susceptible to the same problem. A plethora of research has been conducted over the years to predict the behaviour of certain algorithms in different engineering situations.

2.2. Heuristic Algorithms Applied to Multidisciplinary Design Optimization of Unconventional Airship Configuration [10]

This project is mainly inspired by the work of Ceruti et al., where similar heuristic algorithms were applied to an aerospace engineering problem and then the results were compared to find the most optimum solution. The paper begins with defining the design goals and objectives of the airship, designed with two semi-ellipsoids. The design is optimized to fulfil the objectives of stability, payload requirements, design speed and altitude, and maintain the weight and buoyancy equilibrium. The 3-view representation of the airship and ten configurations are chosen to conduct CFD analysis to find lift and drag coefficients are depicted in Figures 6 and 7 respectively.

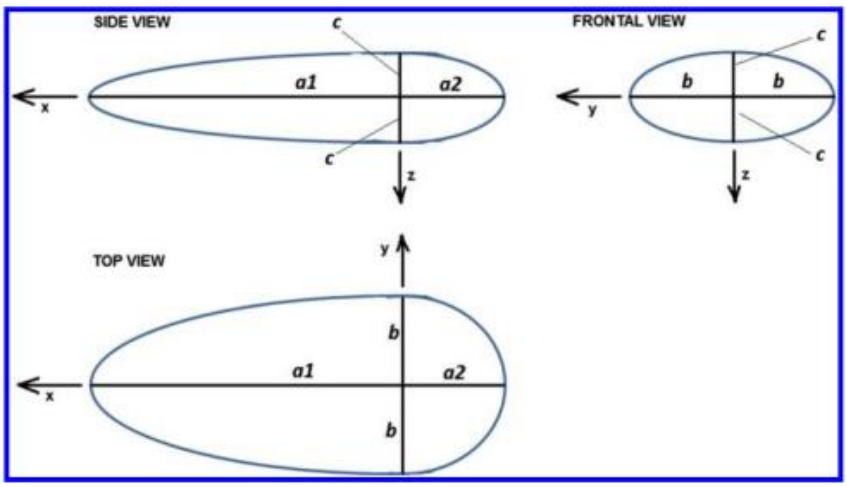


Figure 6: *3-view diagram of airship [10].*

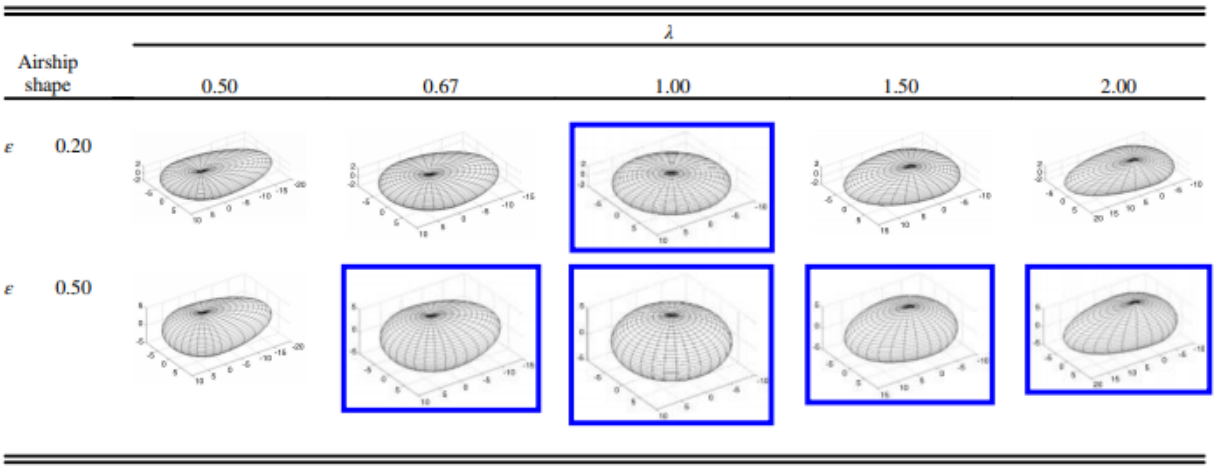


Figure 7: *Ten configurations of airship chosen for CFD analysis [10].*

The basic workings of the six different heuristic algorithms chosen for optimization – Monte Carlo, simulated annealing, differential evolution, GA, PSO, and Imperialist Optimization - are then elaborated. A mathematical model is then designed with the aim to optimize six variables, namely – volume, eccentricity, ratio of area of solar films to surface area, height to width ratio, and horizontal tail scaling. The model developed explains the weight breakdown, energy storage requirements, and power requirements alongside stability. Weight and buoyancy are then estimated, along with the design goals mentioned above to properly compile and define a fitness function (or objective function). The fitness function achieved primarily aims to minimize the mass of the airship.

$$F = (m_t)^u \cdot \frac{w_1 C_1 + w_2 C_2 + w_3 C_3}{w_1 + w_2 + w_3}$$

Experimentation is carried out for three different scenarios – low average speed with medium operational altitude, significantly higher average speed with the same altitude, and high speed with higher operational altitude. These conditions come with differing values of air drag and density, keeping the payload constant. The algorithms were run for 2100 different combinations of parameters each. The final values of fitness obtained for each algorithm have been compared in Figure 8.

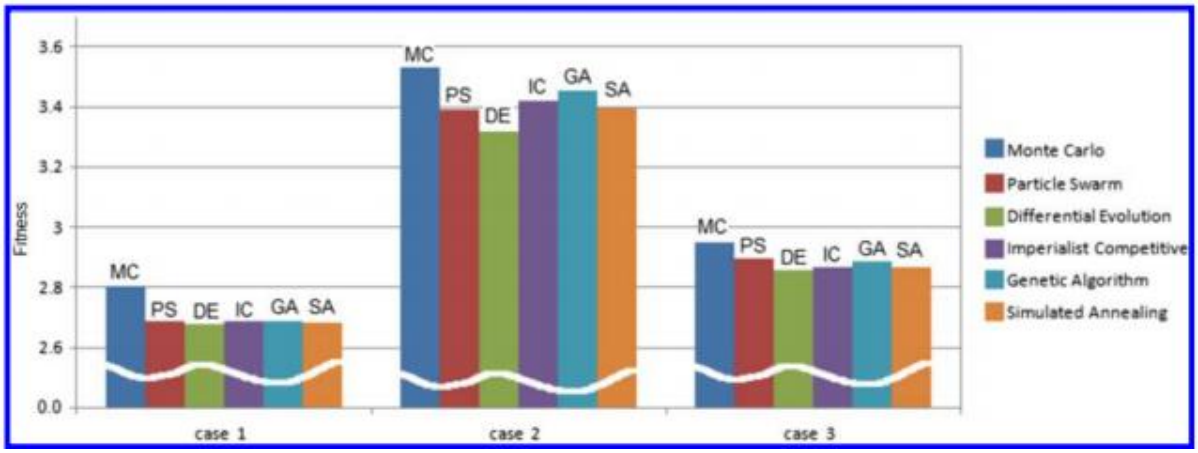


Figure 8: *Comparison of the six different algorithms used [10].*

It was finally concluded that the best optimization results were shown with differential evolution, GA, and PSO, and Monte Carlo algorithm performed the worst in all the three case studies.

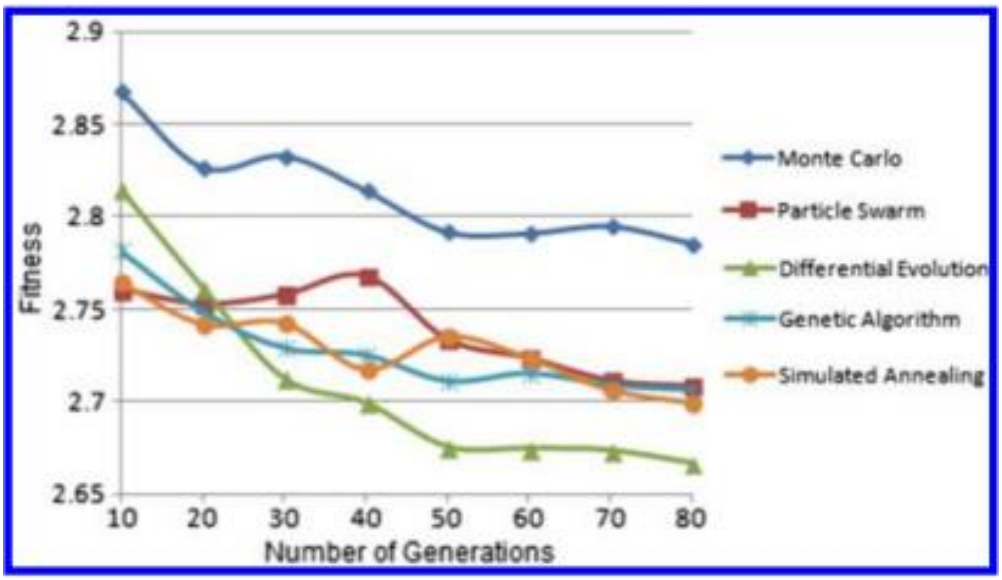


Figure 9: *Fitness v/s Number of generations for the six algorithms [10].*

2.3. Multi-Objective, Multidisciplinary Design Optimization Methodology for Distributed Satellite Systems [11]

Jilla et al. applied and compared four different optimization techniques to a multi-objective MDO for the mathematical model of a distributed satellite system. The paper describes the importance of a properly structured conceptual design of satellite systems, as failure to develop an efficient and optimal structure – due to large number of variables, interdependence between variables, etc. - can result in great increase in life-cycle costs due to constant modifications to improve efficiency at later stages, when changes become much more expensive to integrate. Hence a methodology is presented to provide a better understanding of initial design choice to

reduce long-term life-cycle costs and produce an optimized and efficient space system. The first step is to formulate the objective function, which combines the design variables along with the cost functions, which is then to be minimized. The conversion of standard optimization parameters to specific DSS parameters is shown below in Figure 10.

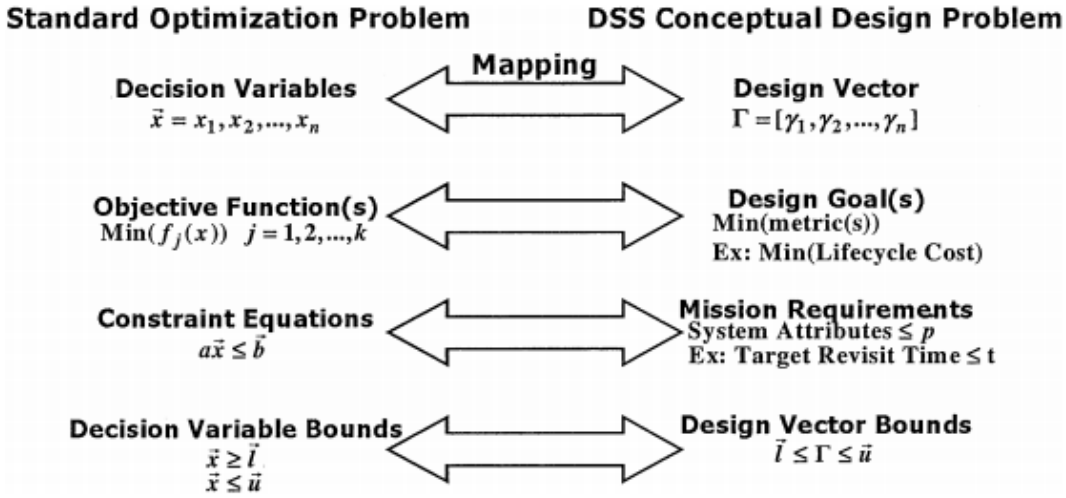


Figure 10: *Conversion of standard optimization parameters to DSS parameters [11].*

The formulation process entails the seven steps detailed in the multi-objective multidisciplinary design optimization systems architecture (MMDOSA) methodology. The first step is to create the generalized information network analysis model (GINA). The GINA model is a framework that portrays the DSS architecture to an information transfer network to visualise physically different architectures with the same quantitative variables. The metrics involved in the model are quality of service metrics, performance, life-cycle cost, and adaptability, and are then integrated into a multidisciplinary system to produce optimum values. Next, preliminary studies are performed by changing just one variable in the baseline function to investigate each parameter individually. A random sample is then selected from the tradespace to be analysed by different optimization techniques. Both single and multi-objective techniques are investigated in the MMDOSA methodology, and it is found that heuristic algorithm simulated annealing performs the best. The simulated annealing algorithm is then formulated trying to maximize performance while minimizing the life-cycle cost of the system. The algorithm is briefly elaborated in Figure 11.

Step	Description
1	Choose a random design vector $\mathbf{\Gamma}_i$, select initial system temperature, and outline cooling schedule.
2	Evaluate $E(\mathbf{\Gamma}_i)$ via the GINA simulation model.
3	Perturb the current design vector $\mathbf{\Gamma}_i$ to obtain a neighboring design vector $\mathbf{\Gamma}_{i+1}$.
4	Evaluate $E(\mathbf{\Gamma}_{i+1})$ via the GINA simulation model.
5	If $E(\mathbf{\Gamma}_{i+1}) < E(\mathbf{\Gamma}_i)$, $\mathbf{\Gamma}_{i+1}$ is the new current design vector.
6	If $E(\mathbf{\Gamma}_{i+1}) > E(\mathbf{\Gamma}_i)$, then accept $\mathbf{\Gamma}_{i+1}$ as the new current design vector with a probability $\exp(-\Delta/T)$ where $\Delta = E(\mathbf{\Gamma}_{i+1}) - E(\mathbf{\Gamma}_i)$; otherwise, $\mathbf{\Gamma}_i$ remains the current design vector.
7	Reduce system temperature according to the cooling schedule.
8	Repeat steps 3–7 until the algorithm terminates.

Figure 11: *Simulated Annealing algorithm [11].*

Three different SA algorithms were formulated and analysed via a test matrix run 100 times each – single-objective SA, multi-objective SA with single solution, and multi-objective SA with multiple solutions – to find the approximate best global Pareto boundary. The basis of Pareto optimization is that it is deemed impossible for an individual variable to be improved without degrading other variables in the sample space. The results concluded that the single solution multi-objective SA performed the worst of all three, the single-objective SA performed the best up to 75 iterations, and multi-objective multi-solution performed the best at the higher iterations. The variations of the three SA algorithms are shown in Figure 12.

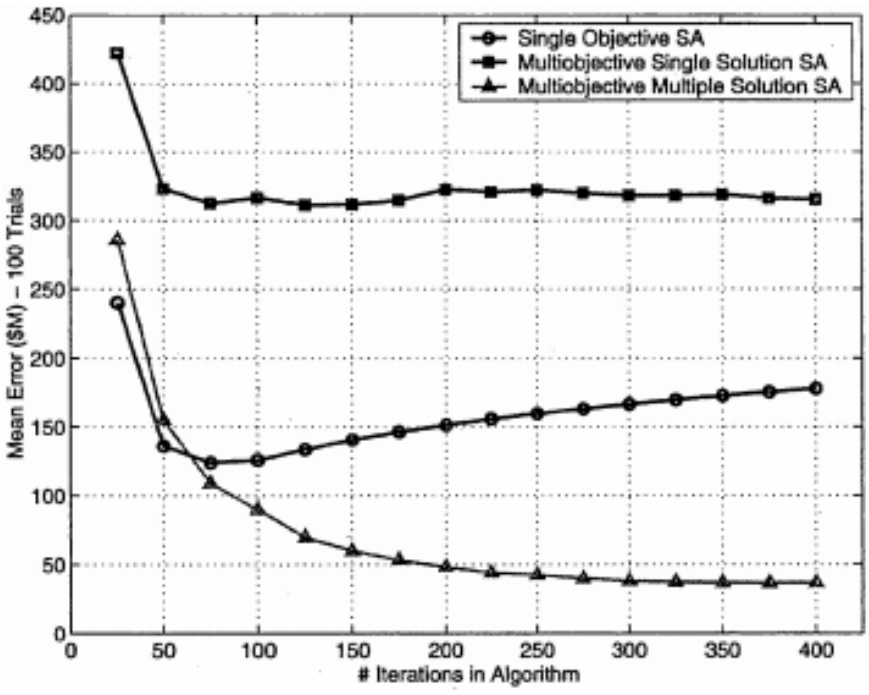


Figure 12: *Variation of the 3 different SA algorithms [11].*

The results from the algorithms are iteratively analysed, and the algorithm is fine-tuned for optimization by possible changes in the GINA model like changing cooling schedules, warm starting a new run with the previously obtained optimum solution or running even more trials. With this, the algorithm can confidently converge on the best architectures in the DSS tradespace.

2.4. Application of Three Bioinspired Optimization Methods for the Design of a Nonlinear Mechanical System [12]

Borges et al. investigated three different evolutionary algorithms applied to a nonlinear dynamic vibration absorber (nDVA) system to maximize its bandwidth of suppression. DVAs are used to reduce noise and vibrations in different engineering structures like aircrafts, ships, robots, etc. The three algorithms used are – Bees Colony (BCA), Firefly Colony (FCA), and Fish Swarm (FSA). Each algorithm is based on the biological phenomenon of its name, for example, BCA works on new information being shared between bees of a colony after a certain area has been scouted for nectar or pollen, resulting in repeated exploring of the best regions as well as newer regions being searched. Similarly, FCA works on the principle of individual fireflies being attracted to the brightest (best) firefly in the pack. Lastly, FSA is based on random search, mimicking four distinct behaviours of school of fish, namely – random, searching, swarming, and leaping. The system to be optimized is elaborated as 2-DOF, nondamped, with a mass fit to the ground with a linear string and a second mass affixed to the configuration with a nonlinear spring. After deriving the equation of motion of the system and representing in matrix form in terms of mass, damping, and stiffness matrices, the final algebraic expression, nonlinear with four equations and variables, is derived to be:

$$\left(1+\mu-\omega_1^2\right)u_1+\mu u_2-2\zeta_1\omega_1 v_1 - \frac{3\varepsilon_1\left(u_1^2+v_1^2\right)u_1}{4}+\beta\omega_1^2=0,$$

$$\mu u_1+\left(\mu-\mu\rho^2\omega_1^2\right)u_2-\mu\left(2\zeta_2\rho\omega_1^2 v_2+\frac{3\varepsilon_2\left(u_2^2+v_2^2\right)u_2}{4}\right)$$

$$\left(\omega_1^2-1-\mu\right)v_1-\mu v_2-2\zeta_1\omega_1 u_1+\frac{3\varepsilon_1\left(u_1^2+v_1^2\right)v_1}{4}=0,$$

$$\mu v_1+\left(\mu\rho^2\omega_1^2-\mu\right)v_2 - \mu\left(2\zeta_2\rho\omega_1^2 u_2-\frac{3\varepsilon_2\left(u_2^2+v_2^2\right)v_2}{4}\right)=0.$$

Numerically solving for the four variables u1, u2, v1, and v2 will finally lead to the vibration amplitudes of both masses, given by:

$$X_i = \left(u_i^2 + v_i^2\right)^{0.5} \quad i = 1, 2$$

As the mechanical parameters are sensitive to physical characteristics, geometric constraints, as well as nonlinearity parameters, the dependence is analysed with a sensitivity analysis.

2.4.1. Bees Colony Algorithm (BCA)

The BCA algorithm extends over a large sample space and expands across multiple directions simultaneously. The algorithm contains the elements of memorizing, learning, as well as communication of data – giving rise to the term swarm intelligence. The parameters that must be fine-tuned are number of bees, number of regions selected, number of best (most populated) regions, number of bees for best regions and regions otherwise, and termination criteria. A basic overview of BCA used for optimization is elucidated in figure.

(1) Initialize population with random solutions
(2) Evaluate fitness of the population
(3) While (stopping criterion not met)
(4) Select sites for neighborhood search
(5) Recruit bees for selected sites and evaluate fitnesses
(6) Select the fittest bee from each site
(7) Assign remaining bees to search randomly and evaluate their fitnesses
(8) End while

Figure 13: *Bees Colony Algorithm (BCA) [12].*

2.4.2. Firefly Colony Algorithm (FCA)

The bioluminescence of fireflies has been analysed to be performing three basic functions – communication, portraying as bait, and to act as a warning to potential predators. The brightness of a firefly is considered as an objective function, as attraction is said to be directly proportional to the brightness of a potential mate. The algorithm also considers all fireflies to be unisex, i.e they have equal chance of being attracted to another firefly. As each firefly will be attracted to brighter mates (better solutions), the sample space is efficiently covered. A randomized movement step is also incorporated in case a firefly finds no mates with higher brightness. The final objective function is defined as:

$$x_i^t = x_i^{t-1} + \lambda\left(x_j^{t-1} - x_i^{t-1}\right) + \alpha\,(\text{rand} - 0.5)\,,$$

Where i is the initial firefly, j the brighter firefly, t is given time, lambda is a factor of attractiveness, and alpha is path randomness consideration.

2.4.3. Fish Swarm Algorithm

In FSA, considering each fish as a possible solution, the food density in a region relates to the objective function, and the weight of the fish is an indication of the food collected and eaten, hence directly relating to success. The objective function defined for each of the four operators is as follows:

i. Individual moment operator:

$$x_i(t+1) = x_i(t) + \text{rand} \times s_{\text{ind}},$$

Where x is the last position of the fish, and s a weighted parameter.

ii. Food operator:

$$W_i(t+1) = W_i(t) + \frac{\Delta f_i}{\max(\Delta f)},$$

Where w is the weight of the fish and f is the objective function.

iii. Instinctive collective moment operator:

$$I_d(t) = \frac{\sum_{i=1}^{N} \Delta x_i \Delta f_i}{\sum_{i=1}^{N} \Delta f_i},$$

$$x_i(t+1) = x_i(t) + I_d(t).$$

Where I is the new direction of the fish and x is the position.

iv. Noninstinctive collective moment operator:

$$B(t) = \frac{\sum_{i=1}^{N} x_i W_i(t)}{\sum_{i=1}^{N} W_i(t)}.$$

$$x(t+1) = x(t) - s_{\text{vol}} \times \text{rand} \times \frac{x(t) - B(t)}{d(x(t), B(t))},$$

Where B is the average position of all fish, d is a calculator of Euclidean distance between the current position and centroid, and s controls the displacement of the fish.

The results of the three algorithms are then compared with results obtained from GA and PSO, each algorithm run for 50 iterations each. It was finally concluded that the results obtained from each of three chosen algorithms performed satisfactorily when compared to results obtained from GA and PSO.

CHAPTER 3
METHODOLOGY

The code of the aerospace engineering problem and the relevant algorithms used for its solution have been written and executed on MATLAB. As MATLAB allows for handling large sample spaces, numerical solutions, graph plotting, as well as in-built evolutionary algorithm codes, it was chosen as the preferred platform for convenient execution of this project. The toolboxes used in this project are – Global Optimization Toolbox, Optimization Toolbox, and SRGTS Toolbox by Vienna. This project entails two major steps – first, to properly fine-tune each algorithm used in accordance with the constraints and requirements of the design problem, and second to compare the results of each algorithm in terms of accuracy and execution time to properly conclude the most optimum algorithm for the problem statement.

3.1. Inputs for main body code (Trilobe_Design_Code)

The nomenclature of the design variables used in this code are defined as follows:

m = Ratio of distance of max section from the nose to length
R0 = Radius of curvature at nose of hull
R1 = Radius of curvature at tail of hull
Cp = Prismatic coefficient
L/D = Fineness ratio
f = Relative distance (Lateral)
g = Relative distance (Vertical)

The framework of this project is based on designing a tri-lobed airship and optimizing its parameters using three different heuristic algorithms. The first input to be received from the user is the shape of the airship, which can be ellipsoidal, NPL (shape determined by the National Physics Laboratory), or general. This input is stored in the variable 'Shapeselection'. The different design parameters pertaining to each shape are given in Table 1.

Shapeselection	**Ellipsoidal (1)**	**NPL (2)**	**General (3)**
Mmin	0.500	0.431939	0.3
Mmax	0.500	0.431939	0.6
r0min	0.500	0.5886	0.5
r0max	0.500	0.5886	1.0
r1min	0.500	0.42485	0
r1max	0.500	0.42485	0.5
cpmin	0.666	0.66675	0.55
cpmax	0.666	0.66675	0.70
12dmin	4.999	4	3
12dmax	4.999	4	6

Table 1: *Design parameters pertaining to each shape.*

Next, the user is given four choices of city of operation – 1. Mumbai, 2. Delhi, 3. Kolkata, and 4. Chennai. The input is stored in the variable 'city', and the different latitude (XLAT) and longitude (XLONG) values pertaining to each choice are given in Table 2.

City	Mumbai (1)	Delhi (2)	Kolkata (3)	Chennai (4)
XLONG (deg)	72.88	77.23	88.36	80.24
XLAT (deg)	19.07	28.65	22.57	13.07

Table 2: *Longitude and latitude values pertaining to each city.*

The final input to be taken from the user is the choice of an energy storage – namely, 1) a rechargeable battery, or 2) a fuel cell, stored in the variable 'storage'. Aside from the user-taken inputs, a number of pre-defined variables are also initialised, relating to constants of flight conditions, aircraft configuration, battery specifications, etc. The variables with their defined values are given in Table 3.

Parameter	Variable Name	Value
Airship Yaw angle (deg)	YAW	90
Minimum Altitude (km)	Hmin	0
Maximum Altitude (km)	Hmax	1
Day of operation	IDAY	209
Solar cell efficiency	ETA_SC	0.12
Power to Payload (W)	P_Pay	1000
Energy density of battery (Wh/kg)	Rho_batt	400
Energy density of fuel cell (Wh/kg)	Rho_fc	1000
Percentage purity of helium (97%)	K_pur	0.97
Specific mass of envelope material – vectran (kg/m^2)	Rho_en	0.2

Table 3: *Pre-defined variables for main body code.*

Next, the payload value in kg is taken as input from the user in the variable 'M_pay'. Then, the selection of required objective function to be minimised must be done. The user can choose between five different variables to minimize – 1) volume of the envelope, 2) area of the array, 3) mass of the airship, 4) surface area of the envelope, or 5) drag coefficient of the envelope. The selection is stored in the variable 'Objectivefunction'. Finally, the user must input the decision of the desired algorithm to be used – 1) Particle Swarm Algorithm (PSO), 2) Genetic Algorithm (GA), or 3) Simulated Annealing (SA), the value of which is stored in variable 'Solver'. The user is then allowed to input their preferred name for the folder to store the data. The function 'optimizeTrilobe' is then called to begin the process of optimization.

For the purpose of this paper, the input values chosen to solve the engineering problem are:

Shapeselection = 1 (Ellipsoidal)

city = 1 (Mumbai)

storage = 1 (Rechargeable Battery)
M_pay = 1400 (kg)
Objectivefunction = 3 (Mass of the airship)

3.2. Solver function (optimizeTrilobe)

Following the assignment of variable bounds as per the selection of shape in variable 'Shapeselection', more bounds are then defined in the code for the remaining variables, as follows in Table 4:

u6min	0
u6max	1
u7min	0
u7max	1
u8min	0
u8max	1
u10min	0
u10max	1
fmin	0.1
fmax	0.3
gmin	0.0
gmax	0.2
w1min	0.0
w1max	0.9
w2min	0.3
w2max	0.3
w3min	0.0
w3max	0.9
w4min	0.0
w4max	0.9

Table 4: *Minimum and maximum bounds for remaining variables.*

The bounds are then compiled into arrays of 'lb' and 'ub'. The number of variables 'nvars' is then assigned as the length of the bound arrays. Next, the different options to fine-tune the options and hyperparameters for each algorithm must be defined, before beginning the trials for optimization.

3.2.1. Particle Swarm Algorithm (1)

First, the options for using hybrid function 'patternsearch' are defined using the 'hybridopts' command as follows in Table 5:

Parameter	Value
InitialMeshSize	nvars*500
ScaleMesh	false
AccelerateMesh	true
MeshTolerance	1e-7

Table 5: *PSO hybrid function parameters.*

Next, the hyperparameters for PSO are defined as follows, within the command 'optimoptions' in Table 6:

Parameter	Value
SwarmSize	nvars*10
MaxIterations	100
PlotFcn	pswplotbestf
MinNeighborsFraction	1
SelfAdjustmentWeight	0 to 2.0
SocialAdjustmentWeight	0 to 2.0
UseVectorized	false
HybridFcn	{patternsearch, hybridopts}

Table 6: *Hyperparameters for PSO.*

After tuning the hyperparameters, MATLAB's inbuilt function 'particleswarm' is called to begin the iterations of the algorithm, sending the objective function 'Optim_Trilobe', the number of variables, the bounds of the variables, and the options, and returning the best value of the function (fbest) as well as its occurrence (xbest). 15 iterations of PSO are carried out, and the values of self and social adjustment weight are uniformly increased from 0 to 2.

3.2.2. Genetic Algorithm (2)

The options for fine-tuning genetic algorithm are initialized with the help of both 'hybridopts' and 'gaoptimset'. The hybrid function 'patternsearch' is used with the same options as those given in table. Then, the hyperparameters are initialized using 'gaoptimset', the values of which are given as follows in Table 7:

Parameter	Value
PlotInterval	1
PopulationSize	nvars*10
Generations	10

EliteCount	0 to 3
CrossoverFraction	0.1 to 0.9
MutationFcn	mutationuniform, 0.01
TolFun	1e-8
MigrationFraction	0.01 to 0.1
PlotFcns	gaplotbestf
HybridFcn	{patternsearch, hybridopts}

Table 7: *Hyperparameters for Genetic Algorithm.*

MATLAB's inbuilt function 'ga' is then called, passing the number of variables, objective functions, bounds, and the options. The function then returns the optimum values fbest and xbest. 13 iterations of the genetic algorithm are carried out, varying three variables in uniform increments – the elite count, crossover fraction, and the migration fraction.

3.2.3. Simulated Annealing (3)

Simulated annealing is defined in the function 'simann', taking inputs of the objective function, bounds, starting values 'x', initial temperature, temperature reduction factor – recommended at a value of 0.85, number of ns loop iterations before temperature reduction – recommended at a value of 5, and the number of times function is run before stepsize adjustment – recommended at 20. The function then returns the optimum values of x as 'xopt'. For SA, the parameters to be changed with every iteration are taken as the starting values 'x', that range from 0.1 to 0.9 for the 16 different variables across 7 iterations. The convergence criterion is taken as 1e-6, and maximum number of function evaluations allowed is assigned as 12000000.

3.3. Objective function definition (Optim_Trilobe)

This file details the formation of the objective function to be optimized, giving the choice of five different variables as mentioned in the previous section. A general variable 'u' is used to assign values to the 16 different variables used in the optimization process. The variable 'u' contains firstly a randomly initialized value and then is assigned as the best value of the previous iteration for the succeeding iterations. The nomenclature of all the variables used as well as their formulae definition is detailed in Table 8.

Parameter	**Variable Name**	**Formula**
Ratio of distance of max section from the nose to length	M	u(1)
Radius of curvature at nose of hull	r0	u(2)
Radius of curvature at tail of hull	r1	u(3)

Prismatic coefficient	cp	u(4)
Fineness ratio	12d	u(5)
Length of the airship (m)	L	$Lmin + u(6) * Lmax$
Starting point of the solar array (m)	XS	$u(7) * (L/2)$
Ending point of the array (m)	XF	$(L/2) + u(8) * (L/2)$
Operating altitude (km)	ALT	$15 + 5 * u(9)$
Angle of array (deg)	Angle	$Amin + Amax * u(10)$
Width of the single lobe (m)	D	$L/12d$
Outer lobe distance (m)	F	$u(11) * D$
Relative distance between lobes in vertical direction (m)	G	$u(12) * (D/2)$
Weighting factor 1	w1	0.1 + u(13)
Weighting factor 2	w2	0.1 + u(14)
Weighting factor 3	w3	0.1 + u(15)
Weighting factor 4	w4	0.1 + u(16)

Table 8: *Nomenclature and formulae definition for objective function.*

Further constant parameters relating to efficiency terms and flight angles are also assigned fixed values as given in Table 9.

Parameter	**Variable Name**	**Value**
Altitude (km)	H	ALT
Packing efficiency	E_pack	0.95
Component efficiency	E_comp	0.95
Convert efficiency	Eta_conv	0.9
Gear efficiency	Eta_gear	0.98
Propulsion efficiency	Eta_prop	0.72
Power density of propulsion system (W/kg)	Rho_prop	440
Specific mass of the solar array (kg/m^2)	Rho_pv	0.3
Acceleration due to gravity	gravity	9.8065
Latitude angle	PHI	XLAT

Table 9: *Fixed values of efficiency parameters and flight angles.*

3.3.1. Geometry

The shape of the airship is defined using Gertler's equation, which he first described for the formation of a submarine hull shape [7]. The Gertler shape coefficients 'A', 'B' and 'a' (A\B)

are then defined for the airship followed by meshing parameters 'm', 'n', and 'p' around the longitudinal and circumferential of the airship. To begin defining the geometry, the radius of curvature 'r_p1' is first defined with the equation below:

$$r_{p1} = D * \sqrt{\sum_{i=1}^{6} a(i) * (\frac{v1}{L})^i}$$

Next, the volume of the Gertler shape is found with the help of function 'Volume_gertler' by passing the length, array 'u', as well as the number of lobes. The width of the shape is also defined as:

$$Width = D + (2 * f)$$

Then, the grid dimensions are created for the three lobes of the airship – one central lobe, and two outer lobes. Next, the wetted area estimation is carried out using the function 'SA_total'. The wind speed model is then calculated with function 'wind_speed', atmospheric properties at the operating altitude like pressure and density are then found using function 'atmosisa'. Then, the coefficient of drag of the hull (CDHull) is estimated the surrogate model function 'CDSurrogate'. The density of helium at sea level (rho_g) is assigned as 0.1692. Density ratio (sigma) is then defined as:

$$sigma = \frac{rho\_OA}{1.225}$$

The density of helium with purity (rho_He) must also be assigned as:

$$rho_{He} = \left(k_{pur} * rho_g + \left(1 - k_{pur}\right) * 1.225\right) * sigma$$

3.3.2. Lift and drag estimation

Next, variables relating to aerodynamic lift are defined and assigned values as follows in Table 10:

Parameter	**Variable Name**	**Formula/Value**
Buoyancy of Airship	Lift	$(rho_{OA} - rho_{He}) * gravity * Volume$
Planform area	[s_plan, AS]	Proj_Area(a,L,D,f)
Aspect ratio of the envelope	AR	$\frac{(Width)^2}{s_{plan}}$
Parameter based on volume and surface area	NL	2.4

Drag due-to-lift factor	k_lift	$(-0.0145 * (\frac{1}{AR})^4 + 0.182 * (\frac{1}{AR})^3 - 0.514 * (\frac{1}{AR})^2 + 0.838 * (\frac{1}{AR} - 0.053)/NL$
Angle of attack in deg	alpha_deg	2
Lift-curve slope in 1/deg	C_L_alpha	$\frac{2 * \pi * AR}{2 + \sqrt{(4 + AR^2)}} * \frac{\pi}{180}$
Lift coefficient	C_LV	$C_{L_{alpha}} * alpha_{deg} * NL$
Aerodynamic lift	L_aero	$0.5 * rho_{OA} * (V_{air})^2 * (Volume)_3^2 * C_{LV}$
Drag coefficient due to lift	Cd_lift	$k_{lift} * (C_{LV})^2 * Cdlift_{yes}$

Table 10: *Aerodynamic lift properties.*

Similar parameters are then defined for the total drag in Table 11.

Parameter	Variable Name	Formula/Value
Ratio of CD total/CDhull	N	1.1
Total coefficient of drag for airship	CD_total	$(N * CDhull) + Cd_{lift}$
Total drag of airship	Dtotal	$0.5 * rho_{OA} * V_{AIR}^2 * (CD_{total}) * Volume_3^2$
Thrust Power required	P_thrust	$\frac{(Dtotal * V_{AIR})}{(ETA_{prop} * ETA_{gear})}$
Power Required for payload + drag	P_total	$P_{thrust} + P_{pay}$

Table 11: *Aerodynamic drag properties.*

3.3.3. Power estimation

The inputs for power estimation are also defined and given appropriate values in Table 12, starting with variable 'q' = 100.

Parameter	Variable Name	Formula/Value
Solar constant (W/m^2)	ISC	1367
Mean Anomaly	Gamma	$2 * \pi * (IDAY - 1)/365$

Declination Angle	delta	$\frac{180}{\pi} * (0.006918 - 0.399912 * \cos(Gamma) + 0.070257 * \sin(Gamma) - 0.006758 * \cos(2 * Gamma) + 0.000907 * \sin(2 * Gamma) - 0.002697 * \cos(3 * Gamma) + 0.001480 * \sin(3 * Gamma))$
True Anomaly	Zi	$Gamma + 0.0344 * \sin(Gamma) + 0.000349 * \sin(2 * Gamma)$
Correction factor for solar radiation at top of atmosphere	Eo	$(1.017 + 0.0174 * \cos(Zi))^2$
Extra-terrestrial normal solar intensity	I_SUN	ISC*Eo
AOD at measurement site	tau_m	0.2688
Water vapour column at measurement site	Omega_m	1.4160
AOD at different altitude	tau	$tau_m^{(-0.691*H)}$
Water vapour column at different altitude	Omega	$Omega_m^{(-0.44*H)}$
Sunrise hour angle	wSRH	$-acosd(-tand(PHI) * tand(delta))$
Sunset hour angle	wSSH	-wSRH
Sunrise time in hour	tSRH	wSRH/15+12
Sunset time in hour	tSSH	wSSH/15+12
Day time duration in hour	Nd	(tSSH-tSRH)
Night time duration in hour	Nt	24 – Nd
Array of hour angle	w	linspace(wSRH,wSSH,q)
Solar Elevation Angle (Sen, 2008)	alpha	asin $(sind(delta) * sind(PHI) + cosd(delta) * cosd(PHI) * cosd(w))$
Relative air mass	m_R	$(\sin(alpha) + 0.15 * (3.885 + alpha)^{-1.253})^{-1}$
Absolute air mass	m_A	$m_R * (\frac{P_{OA}}{101320})$

Direct Normal Radiation	I_DN	$I_{SUN}^{(-0.103*m_A^{0.571})} - (0.081 * (Omega.m_R)^{0.213}) - (tau^{0.91} * m_R^{0.87})$
Scattered/diffused radiation	I_dh	$(0.143 + 0.113 * \sin(alpha) - 0.0485 * Omega + tau) * (I_{SUN} - I_{DN}) * \sin(alpha)$
Total Incidence	In	$I_{DN} + I_{dh}$

Table 12: *Parameters for power estimation.*

The power estimation deals with changing the azimuth angle 'si' depending on the hour angle 'w', and then the incidence angles for all three axes of reference are calculated using trigonometric qualities with angles alpha, si, and the yaw angle of the airship. Then power estimation is carried out for all the three lobes, by first defining their surface normals and then calculating the power based on the incidence angles. Finally, the total power supplied is determined as the sum of the individual power of each of the three lobes.
Next, efficiency terms for the battery system and fuel cell are defined in Table 13.

Parameter	Variable Name	Formula/Value
Battery Efficiency	eta_bat	0.9
Battery Input Line Efficiency	eta_IL	0.98
Battery Output Line Efficiency	eta_OL	0.98
Electricity Transmission Efficiency	eta_T	0.98
Round-Trip Efficiency	eta_RTE	0.9
Total efficiency of battery system	eta_BS	$eta_{bat} * eta_{IL} * eta_{OL} * eta_T * eta_{RTE}$
Efficiency of fuel cell	eta_fc	$0.55 * eta_T$

Table 13: *Efficiency terms for the battery system and fuel cell.*

3.3.4. Mass estimation

The initial variables considered in mass estimation of the airship are initialized in Table 14:

Parameter	Variable Name	Formula/Value
Mass of Propulsion System	M_prop	$\frac{P_{thrust}}{Rho_{prop}}$
Molecular Mass of Helium	MW_he	4.003
Molecular Mass of Air	MW_air	28.97
Mass of lifting gas	M_he	$rho_{OA} * \left(\frac{MW_{he}}{MW_{air}}\right) * Volume$
Mass of hull	M_en	$1.2 * 1.26 * Rho_{en} * SA_{LOBE}$
Area of the septum in m^2	Area_sep	$2 * 0.75 * AS$
Weight of the septum in kg	M_Sep	$4 * Rho_{en} * Area_{sep} * 1.06$

Table 14: *Parameters for mass estimation.*

Based on input for the energy storage device in variable 'storage' the mass of either fuel cell or rechargeable battery storage system is is defined. If rechargeable battery (1) is selected, then the installation factor 'In_fac' is considered as 1.15, and the mass is defined as:

$$M_{stor} = In_{fac} * (\frac{P_{total} * NT}{Rho_{batt} * eta_{BS}})$$

And if fuel cell (2) is selected, then the installation factor 'I_fac' is taken as 1.25, and mass of the storage system is:

$$M_{stor} = I_{fac} * (\frac{P_{total} * NT}{Rho_{fc} * eta_{fc}})$$

The final parameters achieved after mass estimation are defined in Table 15:

Parameter	Variable Name	Formula/Value
Mass of Solar array	M_array	$1.3 * Rho_{pv} * Area_{total}$
Mass of energy subsystem	M_energy	$M_{array} + M_{stor}$
Mass of other components (kg)	M_cont	$0.25 * (M_{en} + M_{tail} + M_{energy} + M_{prop})$
Mass of structure system (kg)	M_struc	$M_{he} + M_{en} + M_{tail} + M_{cont}$
Total Mass (kg)	M_total	$1.006 * (M_{pay} + M_{struc} + M_{energy} + M_{prop} + M_{Sep})$
Energy Required	P_req	$P_{total} * Nd + (\frac{P_{total} * NT}{Eta_{conv}})$

Table 15: *Mass estimation final parameters.*

3.3.5. Constraints

One of the most important steps in defining the objective function is to define the constraints that bound the function. The constraints for the engineering problem are defined as follows:

Parameter	Variable Name	Formula/Value
Total lift of airship (N)	Lift_total	$Lift + L_{aero}$
Power extra (N)	P_extra	$real(P_{req} - P_{sup})$
Lift extra (N)	Lift_extra	$real((M_{total} * gravity) - Lift_{total})$
Energy balance	gg(1)	P_{extra}
Weight-lift balance	gg(2)	$Lift_{extra}$
Weight factors	hh(1)	$(w1 + w2 + w3 + w4) - 1$

Table 16: *Objective function constraints.*

3.3.6. Exterior Penalty function method

The exterior penalty function method is an optimization method which converts a constrained optimization problem into an unconstrained problem with the addition of a term known as a penalty function. The penalty function contains a parameter of penalty that is multiplied to the objective function when it is in violation of a constraint. The parameter remains zero when the constraints are not violated. So, a parameter variable 'penalty' is defined and variable 'scale_factor' is initialized with a value of 10^5. The variable 'penalty' is scaled depending on the weight factors 'hh(1)' and energy balance and weight-lift balance 'gg(1)' and 'gg(2)' respectively defined in the previous section. Depending on the input for variable 'Objectivefunction' relating to minimization of different variables, the objective function is finally defined as follows:

For volume of the envelope (1),

$$MinObj = Volume + penalty * scale_{factor}$$

For area of the array (2),

$$MinObj = Area_{total} + penalty * scale_{factor}$$

For mass of the airship (3),

$$MinObj = M_{total} + penalty * scale_{factor}$$

For surface area of the envelope (4),

$$MinObj = SA_{LOBE} + penalty * scale_{factor}$$

For drag coefficient of the envelope (5),

$$MinObj = w1 * (\frac{CDhull}{0.0535}) + w2 * (\frac{Area_{total}}{10891}) + w3 * \left(\frac{M_{total}}{63372}\right) + w4 * (\frac{SA_{LOBE}}{45408}) + penalty * scale_{factor}$$

3.4. Postprocess

Finally, the function postprocess takes the most optimum value of the objective function, 'fbest' and displays the output in the following format:

The length of the envelope is estimated to be 'L' m.
The width of the envelope is estimated to be 'Width' m.
The fineness ratio is estimated to be 'L/Width'.
The starting point of solar array is estimated to be 'XS' m.
The ending point of solar array is estimated to be 'XF' m.
The angle of array is estimated to be 'Angle' deg.
The operating altitude is estimated to be 'ALT' km.
The wind speed at operating altitude is estimated to be 'V_AIR' m/s.
The length of the array is estimated to be 'Array_length' m.
The area of the array is estimated to be 'Area_total' m^2.
The wetted area of the envelope is estimated to be 'SA_LOBE' m^2.

The hull drag coefficient is estimated to be ‘CDhull’.
The total drag coefficient is estimated to be ‘CD_total’.
The drag is estimated to be ‘Dtotal’ N.
The buoyant lift is estimated to be ‘Lift’ N.
The aerodynamic lift is estimated to be ‘L_aero’ N.
The weight is estimated to be ‘Weight’ N.
The array mass is estimated to be ‘M_array’ kg.
The other component mass is estimated to be ‘M_cont’ kg.
The envelope mass is estimated to be ‘M_en’ kg.
The energy mass is estimated to be ‘M_energy’ kg.
The fin mass is estimated to be ‘M_fin’ kg.
The helium mass is estimated to be ‘M_he’ kg.
The energy storage mass is estimated to be ‘M_stor’ kg.
The structure mass is estimated to be ‘M_struc’ kg.
The total mass is estimated to be ‘M_total’ kg.
The empty weight is estimated to be ‘W_empty’ kg.
The objective value is estimated to be ‘MinObj’.
The total power is estimated to be ‘P_total/1000’ KW.
The power required is estimated to be ‘P_req/1000’ KW.
The excess power during day is estimated to be ‘(P_req – P_total)/1000’ KW.
The power supplied is estimated to be ‘P_sup/1000’ KW.
The extra power is estimated to be ‘P_extra/1000’ KW.
The lift extra is estimated to be ‘Lift_extra’ N.
The volume is estimated to be ‘Volume’ m^3.

Where the variable names display the final optimum value of the variables respectively. The function then generates the envelope and array and finalizes the coordinates for the fin design. The file is saved as a pdf and the image generated is saved as a png file. The output is then exported to an excel file and a folder to save the data is created.

CHAPTER 4
RESULT ANALYSIS

Multiple iterations of the three algorithms are carried out varying the hyperparameters of each algorithm and the results are then tabulated to further analyse the solutions. The objective function and constraints applicable are the same for all iterations and is defined as mentioned in the previous section.

4.1. Particle Swarm Algorithm Results

For PSO, the parameters varied are the self-adjustment weight (c1) and social adjustment weight (c2), from 0.0 to 2.0 uniformly in 15 iterations.

S.No.	**Parameter**	**w/0 C1, C2**	**C1=0, C2=0**	**C1=0.1, C2=0.1**	**C1=0.3, C2=0.3**
1	Length of envelope (m)	430.49	434.23	425.64	426.05
2	Width of envelope (m)	137.78	138.98	136.23	136.35
3	Fineness ratio	3.12	3.12	3.12	3.12
4	Starting point of solar array (m)	1.91	106.01	37.54	36.35
5	Ending point of solar array (m)	295.19	328.33	317.38	405.39
6	Angle of array (deg)	49.92	84.16	54.62	30.23
7	Operating altitude (km)	20	20	19.81	19.86
8	Wind speed at operating alt (m/s)	23.4004	23.4004	23.4507	23.4359
9	Length of array (m)	293.28	222.2	279.85	369.04
10	Area of array (m^2)	35236	39449	36158	35130
11	Wetted area of envelope (m^2)	145081	147614	141832	142092
12	Hull drag coefficient	0.0786	0.0786	0.0786	0.0786
13	Total drag coefficient	0.0877	0.0877	0.0877	0.0877
14	Drag (N)	45390.35	46732.11	46458.66	46157.52
15	Buoyant lift (N)	2313568	2374411	2303914	2293806
16	Aerodynamic lift (N)	25448	25892	25741	25570
17	Weight (N)	2339016	2400301	2329650	2319375
18	Array mass (kg)	13742	15385	14102	13701
19	Other component mass (kg)	31937	32845	31946	31758
20	Envelope mass (kg)	43872	44638	42890	42969
21	Energy mass (kg)	77427	80181	78657	77798
22	Fin mass (kg)	8228	8371	8043	8058
23	Helium mass (kg)	38993	40019	38830	38660
24	Energy storage mass (kg)	63685	64796	64556	64097
25	Structure mass (kg)	117790	120541	116392	116166

26	Total mass (kg)	238517	244766	237562	236514
27	Empty weight (kg)	237117	243366	236162	235114
28	Objective value	238517	244766	237562	236514
29	Total power (kW)	1524	1551	1545	1534
30	Power required (kW)	38458	39129	38983	38707
31	Excess power during day (kW)	36934	37578	37438	37172
32	Power supplied (kW)	38458	39129	38984	38707
33	Extra power (kW)	0.0501	0.0662	0.1447	0.0785
34	Extra lift (N)	0.1863	2.1473	4.1821	0.6899
35	Volume (m^3)	3205503	3289801	3098421	3107007

Table 17: *PSO Results (1).*

S.No.	**C1=0.3, C2=0.5**	**C1=0.7, C2=0.5**	**C1=0.8, C2=0.8**	**C1=1, C2=1**	**C1=1.2, C2=1.0**	**C1=1.2, C2=1.2**	**C1=1.5, C2=1.5**
1	410.19	423.78	426.7	428.03	428.46	421.78	430.32
2	134.47	135.64	136.57	137	137.14	135	137.73
3	3.12	3.12	3.12	3.12	3.12	3.12	3.12
4	11.62	80.7	64.07	30.46	18.24	0.06	5.03
5	416.07	419.43	363.88	427.07	360.48	330.9	301.33
6	26.88	38.5	45.57	24.29	33.5	42.07	46.66
7	19.58	19.73	19.87	19.91	19.95	19.68	20
8	23.5483	23.4794	23.4328	23.4209	23.4112	23.5007	23.4004
9	404.45	338.74	299.81	396.61	342.24	330.84	296.3
10	36679	36447	35557	35638	34781	35695	35047
11	138212	140597	142539	143429	143719	139269	144966
12	0.0786	0.0786	0.0786	0.0786	0.0786	0.0786	0.0786
13	0.0877	0.0877	0.0877	0.0877	0.0877	0.0877	0.0877
14	47330.47	46731.27	46211.73	46144.09	45911.64	46748.78	45893.94
15	2297600	2301685	2300878	2307004	2299630	2287486	2310818
16	26218	25892	25604	25566	25437	25901	25428
17	232817	2327576	2326480	2332570	2325067	2313386	2336246
18	14305	14214	13867	13899	13565	13921	13668
19	32033	31974	31856	31913	31771	31802	31896
20	41795	42516	43104	43373	43461	42115	43838
21	80345	79228	78031	77936	77253	79018	77303
22	7838	7973	8083	8134	8150	7898	8221
23	38724	38793	38779	38882	38758	38554	38947
24	66040	65013	64164	64037	63689	65097	63635
25	114953	115902	116538	117030	116897	115007	117665
26	236967	237350	237239	237860	237094	235903	238234

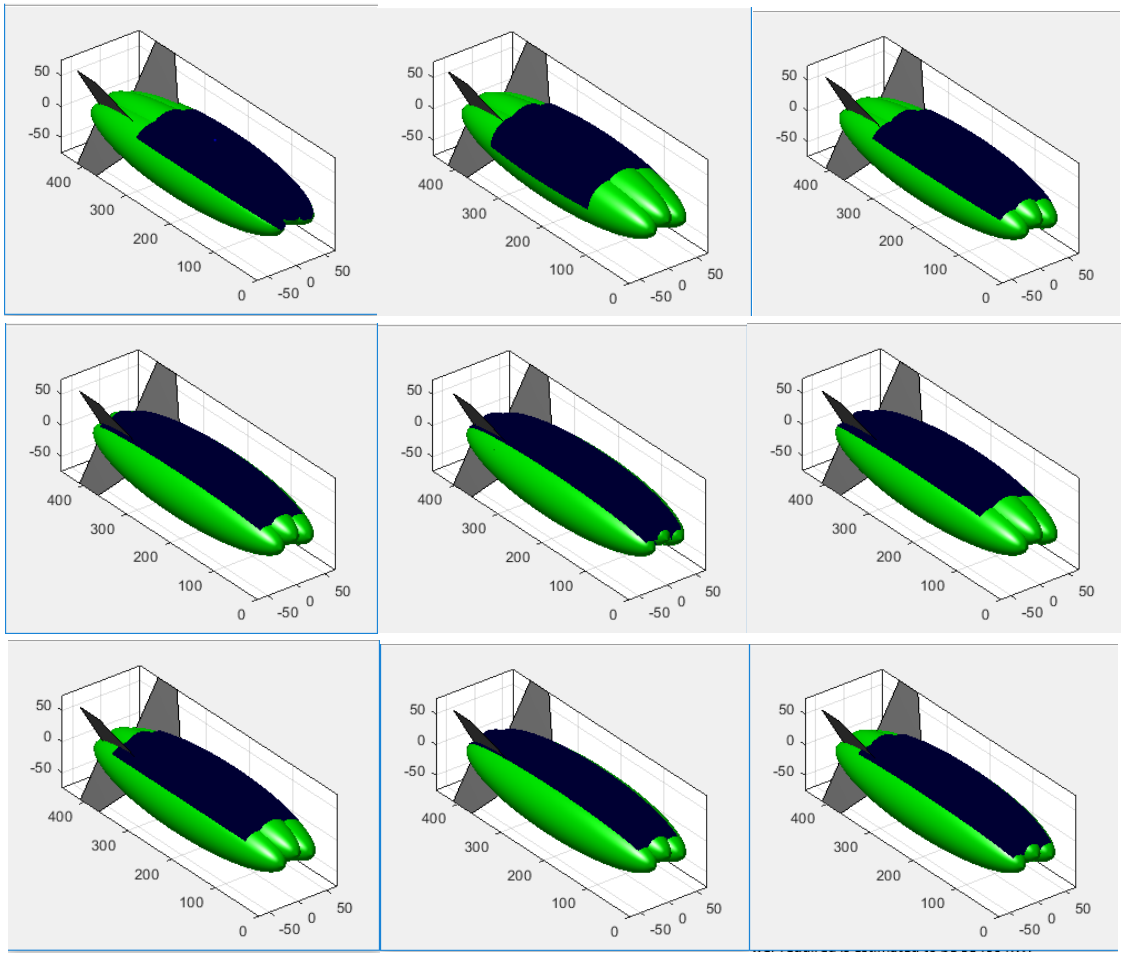
27	235567	235950	235839	236460	235694	234503	236834
28	236967	237350	237239	237860	237094	235903	238234
29	1581	1556	1536	1533	1524	1558	1523
30	39880	39260	38747	38671	38460	39310	38427
31	38299	37704	37211	37138	36936	37752	36904
32	39880	39260	38747	38671	38460	39310	38427
33	0.2415	0.1325	0.0119	0.0007	0.0103	0.0841	0
34	1.4277	0.7456	0.9221	0.1265	0.0524	1.6449	0.0011
35	2980609	3058038	3121628	3150925	3160472	3014837	3201693

Table 18: *PSO Results (2).*

S.No.	**C1=1.8, C2=1.8**	**C1=1.8, C=2.0**	**C1=2, C2=2**
1	429.84	434.15	426.98
2	137.58	138.95	136.65
3	3.12	3.12	3.12
4	0	0	18.56
5	382.55	236.48	425.95
6	26.39	89.79	22.64
7	20	20	19.87
8	23.4004	23.4004	23.4324
9	382.55	236.48	407.39
10	34518	39352	35777
11	144644	147556	142714
12	0.0786	0.0786	0.0786
13	0.0877	0.0877	0.0877
14	45791.92	46713.77	46260.02
15	2303117	2373013	2304564
16	25371	25882	25627
17	2328487	2398895	2330188
18	13462	15347	13953
19	31781	32824	31909
20	43740	44621	43157
21	76955	80118	78183
22	8203	8368	8093
23	38817	39995	38841
24	63493	64771	64230
25	117316	120478	116712
26	237443	244623	237617

27	236043	243223	236217
28	237443	244623	237617
29	1520	1550	1537
30	38342	39113	38787
31	36822	37563	37249
32	38342	39113	38787
33	0.0478	0.0149	0.128
34	1.5426	0.1546	2.5807
35	3191022	3287865	3127429

Table 19: *PSO Results (3).*



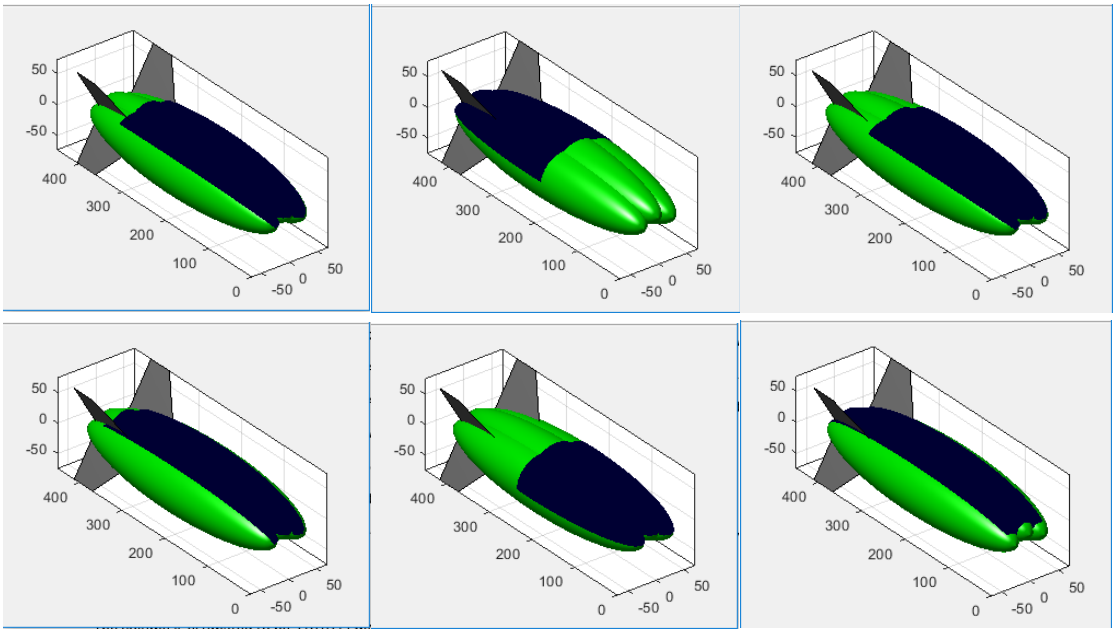


Figure 14: *PSO Results for all iterations.*

4.2. Genetic Algorithm Results

In the fine-tuning of Genetic Algorithm, three major hyperparameters were varied – the Elite Count (EC), from 0 to 3, the Crossover Fraction (CF) from 0.0 to 0.9, and the Migration Fraction (MF) from 0.01 to 0.1, uniformly across 13 iterations.

S.No.	**Parameter**	**{EC, CF=0.9, MR}**	**{ , 0.9, 0.01}**	**{0, 0.0, 0.01}**	**{0, 0.1, 0.02}**
1	Length of envelope (m)	453.68	477.24	445.29	475.26
2	Width of envelope (m)	140.95	139.16	139.96	138.45
3	Fineness ratio	3.22	3.43	3.18	3.43
4	Starting point of solar array (m)	86.44	88.03	35.44	34.2
5	Ending point of solar array (m)	392.09	415.27	325.38	410.75
6	Angle of array (deg)	56.95	64.97	61	51.53
7	Operating altitude (km)	19.91	19.8	19.83	19.77
8	Wind speed at operating alt (m/s)	23.421	23.454	23.4427	23.4638
9	Length of array (m)	305.65	327.24	289.94	376.55
10	Area of array (m^2)	41395	47280	40494	45418
11	Wetted area of envelope (m^2)	152953	159331	149738	157881
12	Hull drag coefficient	0.084	0.0875	0.0828	0.0875
13	Total drag coefficient	0.0936	0.0975	0.0923	0.0975
14	Drag (N)	53268.37	59456.44	51890.57	59257.6
15	Buoyant lift (N)	2592129	2842010	2523665	2815811
16	Aerodynamic lift (N)	26927	27352	26871	27224

17	Weight (N)	2619053	2869361	2550535	2843033
18	Array mass (kg)	16144	18439	15793	17713
19	Other component mass (kg)	35856	39212	34999	38842
20	Envelope mass (kg)	46253	48182	45281	47743
21	Energy mass (kg)	90062	101055	87867	100088
22	Fin mass (kg)	8791	9243	8569	9160
23	Helium mass (kg)	43688	47900	42534	47458
24	Energy storage mass (kg)	73918	82616	72074	82375
25	Structure mass (kg)	128890	138415	125746	137102
26	Total mass (kg)	267073	292598	260086	289913
27	Empty weight (kg)	265673	291198	258686	288513
28	Objective value	267073	292598	260086	289913
29	Total power (kW)	1769	1977	1725	1972
30	Power required (kW)	44637	49890	43524	49744
31	Excess power during day (kW)	42868	47912	41799	47772
32	Power supplied (kW)	44637	49890	43524	49744
33	Extra power (kW)	0.0218	0.1427	0.0228	0.3212
34	Extra lift (N)	2.6913	0.0955	0.2448	1.6125
35	Volume (m^3)	3540182	3816386	3406777	3764921

Table 20: *GA Results (1).*

S.No.	**{0, 0.2, 0.03}**	**{1, 0.3, 0.04}**	**{1, 0.4, 0.05}**	**{2, 0.5, 0.06}**	**{2, 0.6, 0.07}**	**{2, 0.7, 0.08}**
1	463.31	471.74	485.28	455.63	452.12	489.16
2	139.94	138.12	140.07	140.9	137.16	136.1
3	3.31	3.42	3.46	3.23	3.3	3.59
4	97.07	92.94	34.92	31.82	53.82	35.2
5	400.41	407.7	419.4	393.78	440.78	379.93
6	60.9	61.63	48.3	41.29	36.51	64.44
7	19.98	20	20	19.99	19.98	19.9
8	23.4035	23.4013	23.4009	23.4029	23.4037	23.4223
9	303.34	314.76	384.49	361.95	386.96	344.73
10	42936	44568	44953	39777	39372	48925
11	156832	159936	163242	153859	151894	164893
12	0.0847	0.0856	0.088	0.0841	0.0827	0.0886
13	0.0944	0.0954	0.098	0.0937	0.0922	0.0987
14	54451.67	56064.94	59191.24	52913.57	51368.74	61160.08
15	2666798	2744916	2861033	2584419	2532630	2942997

16	26639	26426	26839	26614	25836	26706
17	2693436	2771342	2887870	2611032	2558464	2969701
18	16745	17382	17532	15513	15355	19081
19	36760	37761	39214	35649	34885	40446
20	47426	48365	49364	46527	45933	49864
21	92248	95113	99593	88882	86586	103949
22	9031	9217	9478	8846	8724	9563
23	44946	46263	48220	43558	42685	49602
24	75503	77731	82062	73369	71231	84868
25	132393	135730	140237	128932	126652	143269
26	274658	282603	294485	266255	260895	302830
27	273258	281203	293085	264855	259495	301430
28	274658	282603	294485	266255	260895	302830
29	1807	1860	1964	1756	1705	2031
30	45594	46940	49555	44306	43015	51249
31	43787	45079	47591	42550	41310	49218
32	45594	46940	49555	44306	43015	51250
33	0.0674	0.0268	0.1736	0.0211	0.2081	0.1115
34	1.0249	0.7541	2.7848	1.0392	1.2566	1.9688
35	3685566	3800418	3962536	3573452	3499532	4016179

Table 21: *GA Results (2).*

S.No.	**{3, 0.8, 0.08}**	**{3, 0.9, 0.09}**	**{3, 0.9, 0.1}**
1	468.62	489.16	455.16
2	140.21	140.16	141.52
3	3.34	3.49	3.22
4	33.72	35.2	54.33
5	360.16	386.58	384.32
6	56.95	57.23	48.09
7	19.93	20	20
8	23.4149	23.401	23.4009
9	326.44	351.39	329.99
10	43516	46400	40114
11	158060	164778	154077
12	0.086	0.0884	0.0839
13	0.0959	0.0984	0.0936
14	56319.58	60036.33	52787.79
15	2727567	2903822	2583593

16	26922	26941	26714
17	2754489	2930761	2610306
18	16971	18096	15644
19	37595	39789	35654
20	47797	49829	46593
21	95101	101330	88833
22	9118	9571	8854
23	45971	48941	43544
24	78129	83233	73189
25	134598	142030	129003
26	280884	298859	266181
27	279484	297459	264781
28	280884	298859	266181
29	1870	1992	1752
30	47180	50262	44197
31	45310	48270	42445
32	47180	50263	44197
33	0.007	0.0987	0.0141
34	0.0842	1.5412	0.5829
35	3739347	4021235	3578120

Table 22: *GA Results (3).*

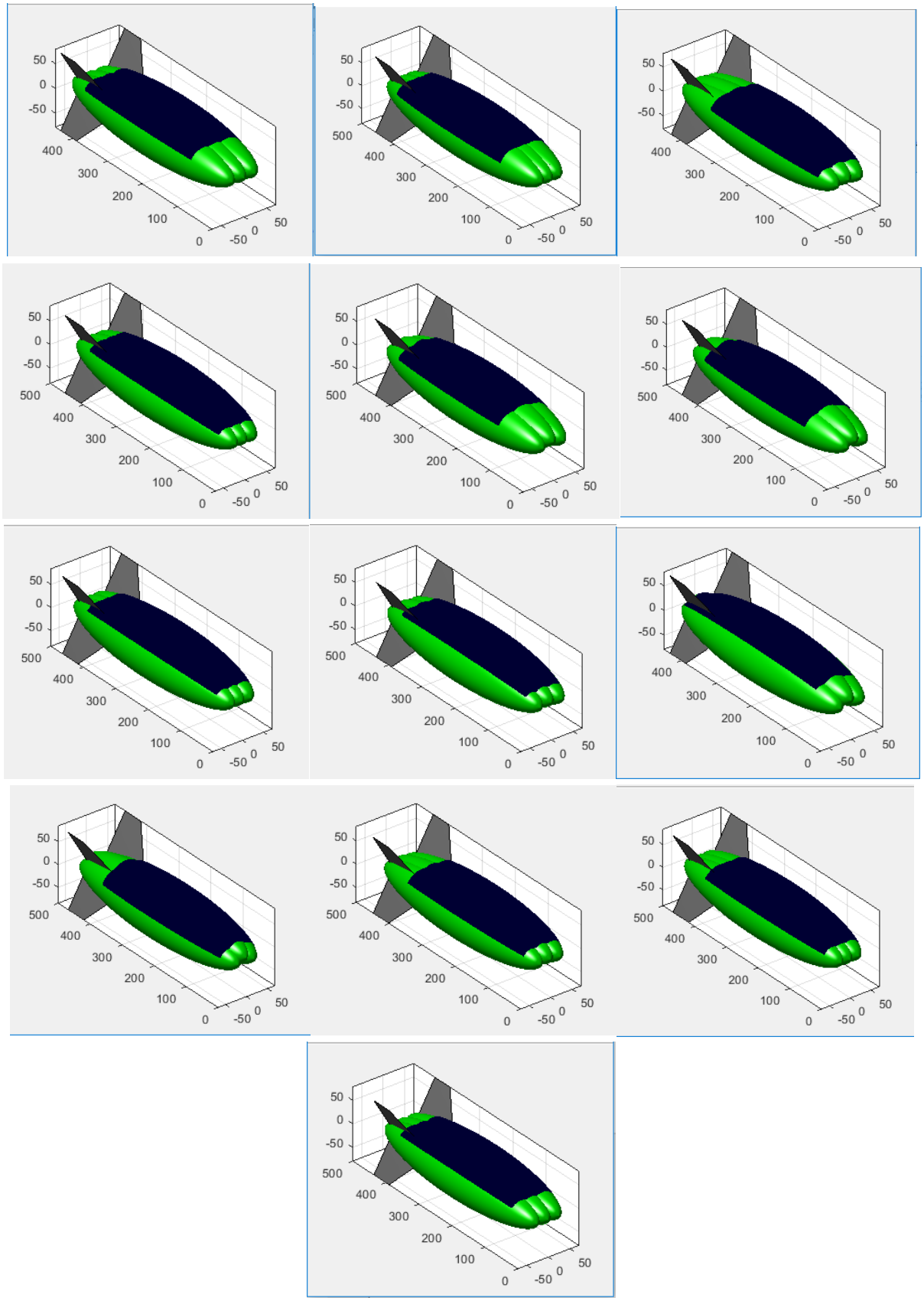
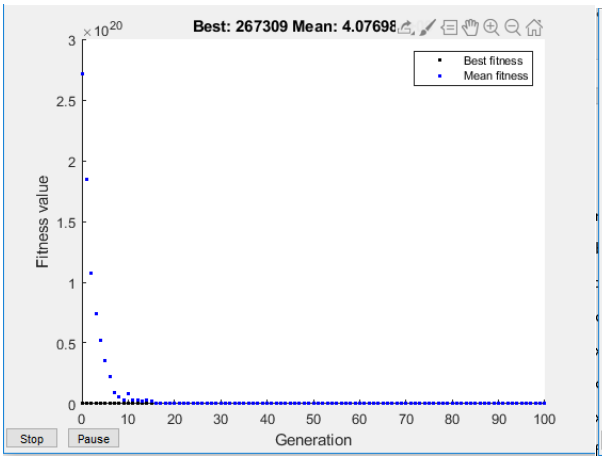
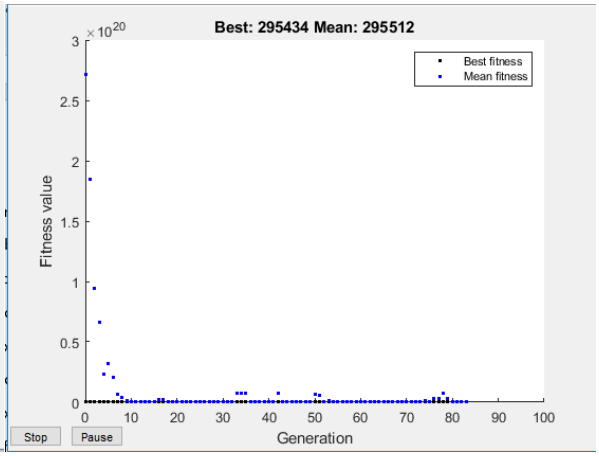
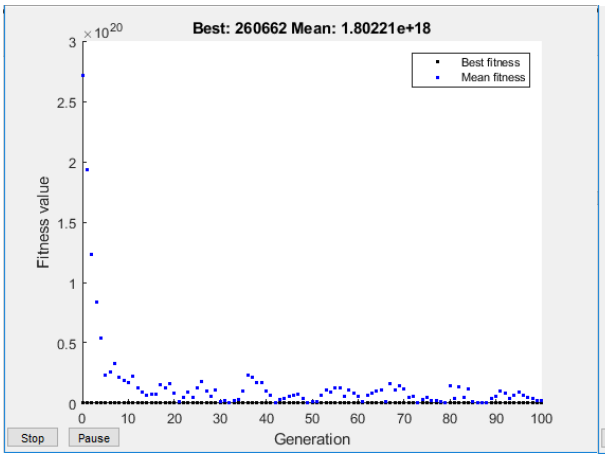
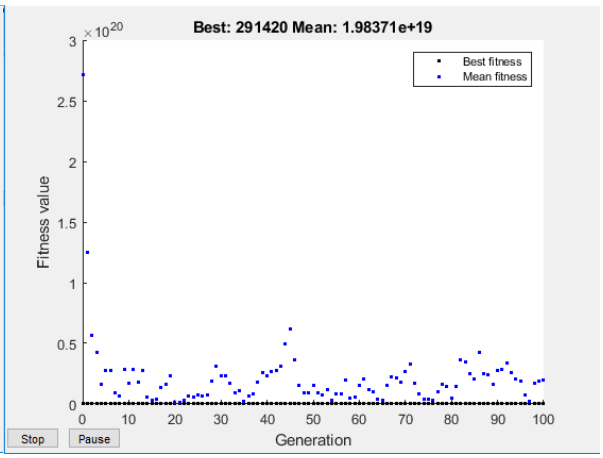


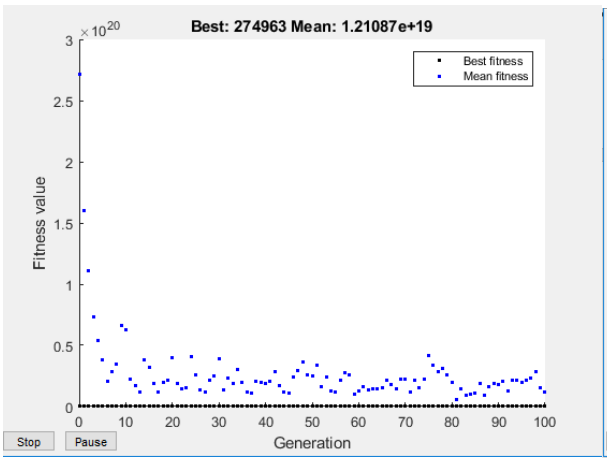
Figure 15: *GA results for all iterations.*

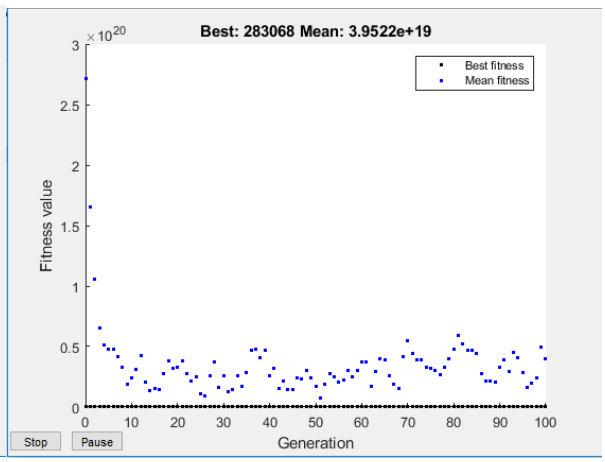

Best: 267309 Mean: 4.07698
Best fitness
Mean fitness
Fitness value
Generation
Stop
Pause

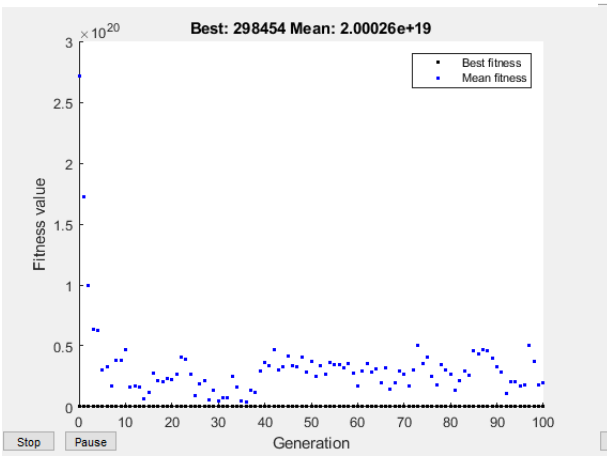

Best: 295434 Mean: 295512
Best fitness
Mean fitness
Fitness value
Generation
Stop
Pause

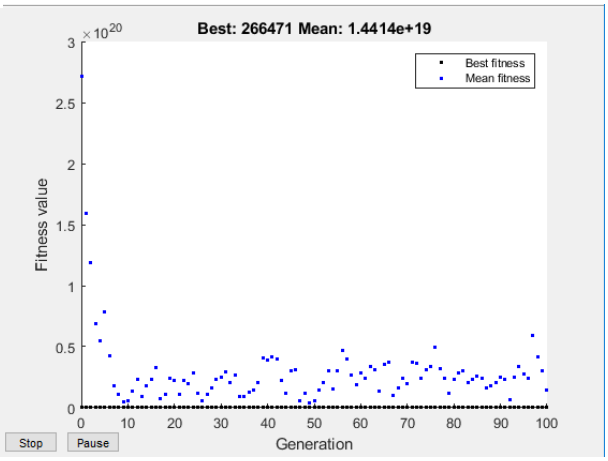

Best: 260662 Mean: 1.80221e+18
Best fitness
Mean fitness
Fitness value
Generation
Stop
Pause


Best: 291420 Mean: 1.98371e+19
Best fitness
Mean fitness
Fitness value
Generation
Stop
Pause


Best: 274963 Mean: 1.21087e+19
Best fitness
Mean fitness
Fitness value
Generation
Stop
Pause


Best: 283068 Mean: 3.9522e+19
Best fitness
Mean fitness
Fitness value
Generation
Stop
Pause


Best: 298454 Mean: 2.00026e+19
Best fitness
Mean fitness
Fitness value
Generation
Stop
Pause


Best: 266471 Mean: 1.4414e+19
Best fitness
Mean fitness
Fitness value
Generation
Stop
Pause

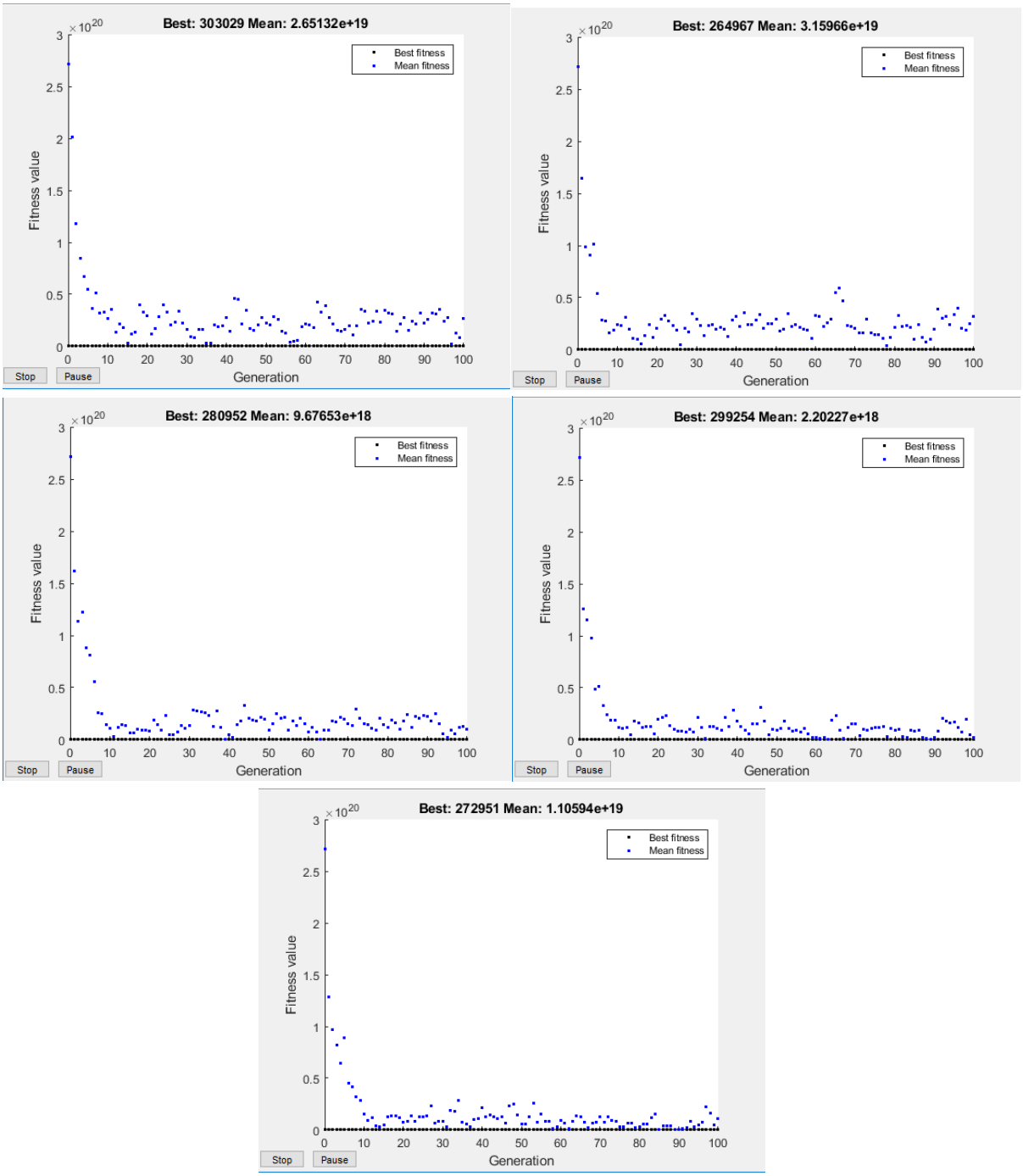


Figure 16: *Fitness curve for all GA iterations.*

4.3. Simulated Annealing Results

To fine-tune simulated annealing, the variables changed for each iteration are the starting values of the 16 design variables, taken as input argument array 'x'. The values range from 0.1 to 0.9 and are varied across 7 iterations.

S.No.	**Parameter**	[0.5,0.5,0.5,0.6, 5,0.3,0.7,0.1, 0.5,0.1,0.1,0.1,	[0.5,0.5,0.5,0.6, 5,0.3,0.7,0.1, 0.5,0.2,0.2,0.2,	[0.5,0.5,0.5,0.6, 5,0.3,0.7,0.2, 0.5,0.3,0.3,0.3,

		0.3,0.3,0.5,0.8]	0.4,0.4,0.5,0.9]	0.5,0.5,0.5,0.9]
1	Length of envelope (m)	425.27	426.38	431.39
2	Width of envelope (m)	136.11	136.47	138.07
3	Fineness ratio	3.12	3.12	3.12
4	Starting point of solar array (m)	18.82	35.68	54.75
5	Ending point of solar array (m)	298.32	406.61	294.19
6	Angle of array (deg)	54.85	29.3	71.79
7	Operating altitude (km)	19.8	19.88	19.97
8	Wind speed at operating alt (m/s)	23.4542	23.43	23.4068
9	Length of array (m)	279.51	370.92	239.44
10	Area of array (m^2)	36288	35109	37599
11	Wetted area of envelope (m^2)	141584	142324	145690
12	Hull drag coefficient	0.0786	0.0786	0.0786
13	Total drag coefficient	0.0877	0.0877	0.0877
14	Drag (N)	46466.77	46061.91	46379.58
15	Buoyant lift (N)	2301606	2292239	2339822
16	Aerodynamic lift (N)	25745	25521	25697
17	Weight (N)	2328422	2318821	2366632
18	Array mass (kg)	14152	13693	14664
19	Other component mass (kg)	31941	31740	32377
20	Envelope mass (kg)	42815	43039	44057
21	Energy mass (kg)	78729	77641	78989
22	Fin mass (kg)	8029	8071	8262
23	Helium mass (kg)	38792	38634	39436
24	Energy storage mass (kg)	64577	63948	64325
25	Structure mass (kg)	116258	116218	118837
26	Total mass (kg)	237437	236458	241333
27	Empty weight (kg)	236037	235058	239933
28	Objective value	237437	236458	241333
29	Total power (kW)	1546	1531	1540
30	Power required (kW)	38996	38617	38844
31	Excess power during day (kW)	37451	37086	37305
32	Power supplied (kW)	39093	38713	38950
33	Extra power (kW)	97.0347	96.5824	105.8833
34	Extra lift (N)	1070.8723	1060.9265	1112.614

35	Volume (m^3)	3090310	3114580	3225707

Table 23: *SA Results (1).*

S.No.	[0.6,0.6,0.6,0.6,6,0.3,0.7,0.3,0.5,0.3,0.3,0.3,0.5,0.5,0.5,0.9]	[0.7,0.7,0.7,0.7,6,0.3,0.7,0.3,0.5,0.3,0.3,0.3,0.6,0.6,0.5,0.8]	[0.8,0.8,0.7,0.7,6,0.4,0.7,0.3,0.5,0.4,0.4,0.3,0.7,0.6,0.5,0.8]	[0.7,0.7,0.6,0.6,6,0.4,0.6,0.3,0.5,0.5,0.5,0.3,0.6,0.6,0.4,0.9]
1	425.81	426.27	377.49	333.57
2	136.29	136.43	135.92	133.45
3	3.12	3.12	2.78	2.5
4	27.98	20.64	16.42	111.91
5	327.88	334.94	257.36	283.26
6	46.86	42.72	36.36	42.03
7	19.84	19.87	19.77	19.54
8	23.4405	23.433	23.4666	23.5681
9	299.91	314.29	240.94	171.34
10	35591	35311	26901	20531
11	141945	142249	124351	107890
12	0.0786	0.0786	0.0688	0.0584
13	0.0877	0.0877	0.0769	0.0656
14	46230.06	46121.57	35072.4	26093.07
15	2295491	2294021	1828573	1445124
16	25614	25554	24622	23744
17	2322177	2320647	1854102	1469648
18	13880	13771	10491	8007
19	31817	31775	25452	20198
20	42924	43016	37604	32626
21	78091	77810	59269	44464
22	8050	8067	6862	5728
23	38688	38664	30819	24356
24	64210	64039	48777	36457
25	116191	116248	96159	78902
26	236800	236644	189069	149865
27	235400	235244	187669	148465
28	236800	236644	189069	1498665
29	1537	1533	1167	873
30	38775	38672	29455	22015
31	37238	37139	28288	21143
32	38875	38770	29533	22083

33	99.7136	98.4954	77.2299	67.4938
34	1071.7475	1072.1152	906.6325	779.9388
35	3102146	3112097	2442100	1863506

Table 24: *SA Results (2).*

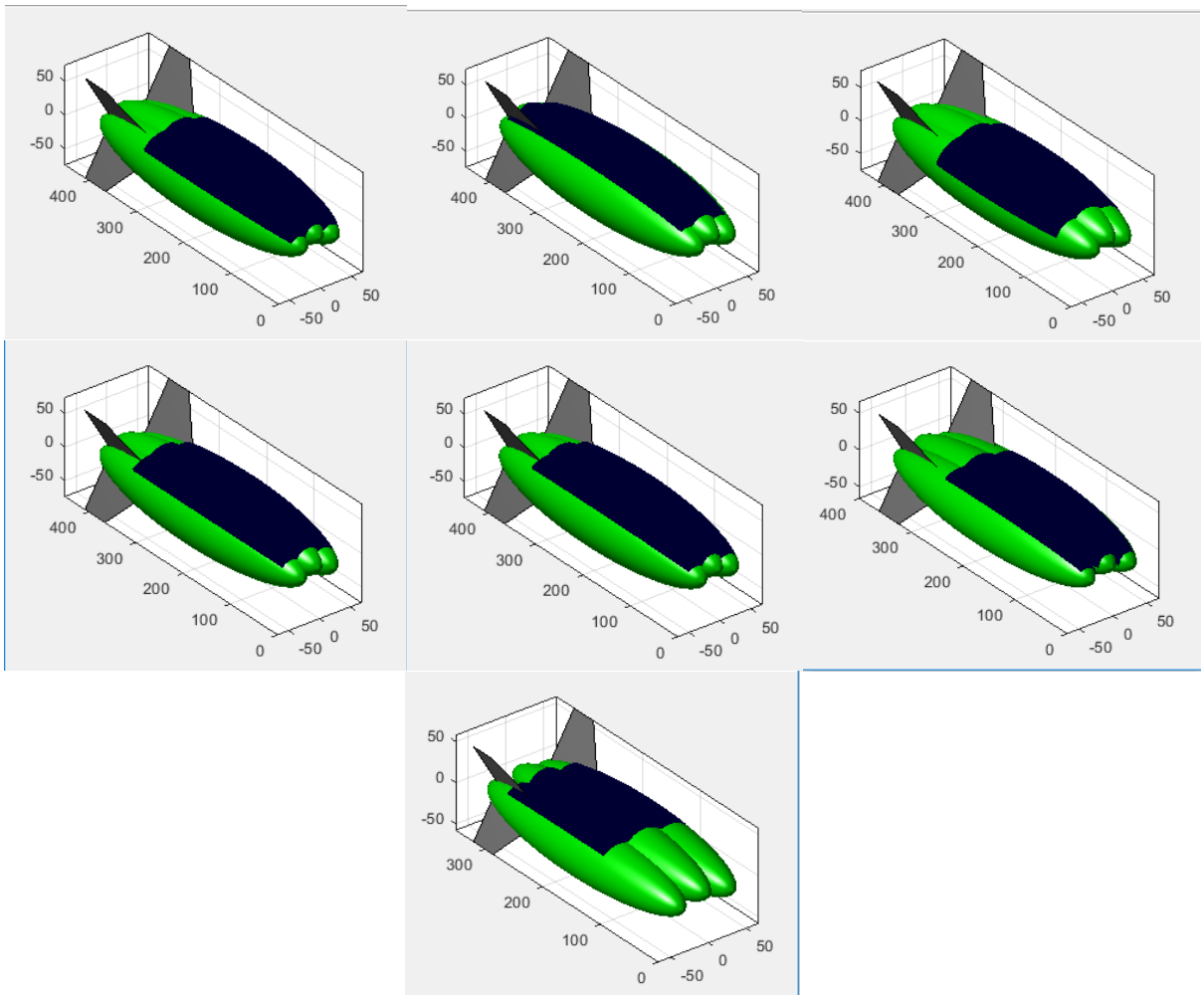


Figure 17: *SA results for all iterations.*

As clearly seen in the table above, the results from simulated annealing seem to start failing at higher initial values of the variable array ‘x’. For this reason, only 7 iterations of SA were able to be carried out before the computations failed.

CHAPTER 5
CONCLUSION AND FUTURE SCOPE OF WORK

The objective of this project was to utilize and compare heuristic algorithms in their application to multi-disciplinary design optimization. The engineering problem considered for this project was a minimization problem of the mass of an airship. The objective function was first formulated, taking into consideration all the various aerodynamic and mechanical factors involved in the computation of the minimization problem. After formulation of the objective function, three different heuristic algorithms were applied to carry out the optimization of the mass of the airship under the constraints of weight and energy balance. The algorithms considered in this paper were particle swarm optimization, genetic algorithm, and simulated annealing. These algorithms are based off natural phenomenon and use heuristics to achieve an approximate accurate solution to the problem. Multiple iterations of each algorithm were carried out, varying the hyperparameters to finetune each algorithm and measure the variations. Finally, the results consisting of various aerodynamic and power parameters were tabulated and presented. It was observed that Simulated Annealing (SA) performed the worse of all the algorithms, having the longest computation time and failing at greater variations of the parameters. Both GA and PSO performed well with minimal deviations of values for the objective function across the iterations.

The use of heuristic algorithms in MDO has been limited so far, due to the prominent drawbacks of no exact solutions and susceptibility to local optimal solutions in the sample space. Much work is being conducted on further tuning and bettering the evolutionary algorithms to achieve more accurate solutions and minimize deviations from the optimal solution. Metaheuristic methods are also being increasingly employed in similar experiments to better the optimization techniques and reduce dependency on randomized variables. The field of evolutionary algorithms is ever-growing and expanding its applications to all kinds of engineering and social problems.

REFERENCES

[1] Bil C. (2015) Multidisciplinary Design Optimization: Designed by Computer. In: Stjepandić J., Wognum N., J.C. Verhagen W. (eds) Concurrent Engineering in the 21st Century. Springer, Cham. *https://doi.org/10.1007/978-3-319-13776-6_15*.

[2] Kennedy, J., & Eberhart, R. (n.d.). Particle swarm optimization. Proceedings of ICNN'95 - International Conference on Neural Networks. *doi:10.1109/icnn.1995.488968.*

[3] Poli, R., Kennedy, J., & Blackwell, T. (2007). *Particle swarm optimization. Swarm Intelligence, 1(1), 33–57. doi:10.1007/s11721-007-0002-0***.**

[4] https://towardsdatascience.com/fine-tuning-the-strategy-using-a-particle-swarm-optimization-a5a2dc9bd5f1.

[5] https://deepai.org/machine-learning-glossary-and-terms/particle-swarm-optimization.

[6] Katoch, S., Chauhan, S.S. & Kumar, V. A review on genetic algorithm: past, present, and future. *Multimed Tools Appl* 80, 8091–8126 (2021). *https://doi.org/10.1007/s11042-020-10139-6.*

[7] Albadr, M. A., Tiun, S., Ayob, M., & AL-Dhief, F. (2020). *Genetic Algorithm Based on Natural Selection Theory for Optimization Problems. Symmetry, 12(11), 1758. doi:10.3390/sym12111758***.**

[8] https://towardsdatascience.com/optimization-techniques-simulated-annealing-d6a4785a1de7.

[9] Moins, S., 2002: Implementation of a simulated annealing algorithm for Matlab. Tech. Rep. LITH-ISY-3339-2002, Electronic Systems Dept., Linko¨ping Institute of Technology, Linko¨ping, Sweden, 34 pp. *[http://www.ep.liu.se/exjobb/isy/2002/3339/]*.

[10]. Ceruti, A., Voloshin, V., & Marzocca, P. (2014). *Heuristic Algorithms Applied to Multidisciplinary Design Optimization of Unconventional Airship Configuration. Journal of Aircraft, 51(6), 1758–1772. doi:10.2514/1.c032439.*

[11] Jilla, C. D., & Miller, D. W. (2004). *Multi-Objective, Multidisciplinary Design Optimization Methodology for Distributed Satellite Systems. Journal of Spacecraft and Rockets, 41(1), 39–50. doi:10.2514/1.9206.*

[12] Romes Antonio Borges, Fran Sérgio Lobato, Valder Steffen, "Application of Three Bioinspired Optimization Methods for the Design of a Nonlinear Mechanical System", *Mathematical Problems in Engineering*, vol. 2013, Article ID 737502, 12 pages, 2013. *https://doi.org/10.1155/2013/737502*.

[13] Goffe, W. L. (1996). *SIMANN: A Global Optimization Algorithm using Simulated Annealing. Studies in Nonlinear Dynamics & Econometrics, 1(3). doi:10.2202/1558-3708.1020.*

[14] Chipperfield, Andrew & Fleming, Peter & Pohlheim, Hartmut. (1994). A genetic algorithm toolbox for MATLAB. Proceedings of the International Conference on Systems Engineering. 200-207.

[15] https://www.nap.edu/read/2026/chapter/3.

[16] https://en.wikipedia.org/wiki/Simulated_annealing.
[17] https://www.geeksforgeeks.org/genetic-algorithms/.
[18] https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3.

PROJECT DETAILS

Student Details			
Name	Sanjana Srivastava		
Registration number	170933064	Roll. No	34
E-mail id	sanjana.srivas98@gmail.com	Phone no.	8767986051
Project Details			
Project title	Heuristic algorithms used in multidisciplinary design optimization		
Project duration	6 months	Date of final presentation	
Guide Details			
Name of Faculty	Prof. Manikandan M.		
Full contact address	Manipal Institute of Technology, Udupi - Karkala Rd, Eshwar Nagar, Manipal, Karnataka 576104, India		
E-mail id	manikandan.m@gmail.com		