PERSONAL PROGRAMMING PROJECT

TOPIC:

Truss Optimization based on FEM using Multi Objective Genetic Algorithm

Supervisor: Dr. Ing. Martin Abendroth

Name	Matriculation Number
Venkatasubramanian Sundaramoorthy Venkatasubramanian.sundaramoorthy@student.tu-freiberg.de	64143

April 15, 2021

Contents

List of Figures

1 Introduction

Genetic algorithms (GA) are based on biological evolution and include and adapt concepts such as chromosomes, genes, natural selection, fitness, crossover combinations, and mutation. Stochastic methods are used in genetic algorithms to identify random solutions in the design space and then guide them towards the best solution. In this problem, I combined a FEM problem of a truss with multi objective (stress and strain) and used a Genetic Algorithm to optimize the mass of the Truss. Natural sciences, mathematics, and computer science are examples of engineering problems where GA principles can be applied. This involves using Genetic Algorithms in engineering applications, data mining, and a variety of other image processing applications. As a result, I'm inspired to work on a project involving Genetic Algorithms.

2 Theoretical Background and Implementation

2.1 Main program

The aim of the optimization is to reduce the structure's total mass while staying below the maximum permissible stress and displacement. For plane truss problems, the finite element approach is used to evaluate stresses and displacements of candidate solutions. The penalty function is applied to nonconforming individuals to deal with nonlinear constraints. The mass of the truss (which must be minimized), but depending on the severity of the constraint violating, this value is then penalized by a certain amount.

Penalty Function:

The goal of penalty functions is to turn constrained problems into unconstrained ones by imposing an artificial penalty for breaking the constraint.

Figure 1: Penalty Function The

algorithm's general principle scheme is shown below.

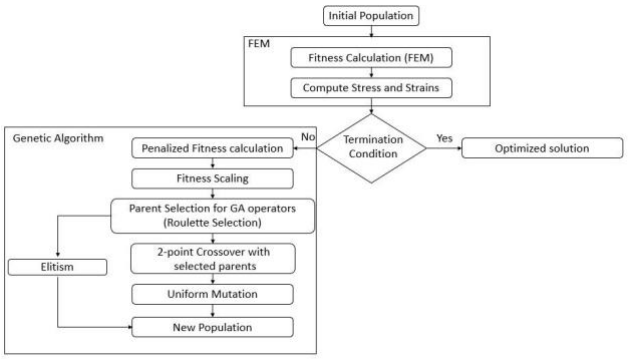

Figure 2: Main Program work flow

The algorithm begins by creating a population of candidate solutions at random within the design space (allowable parameters range). From here, the population is iterated in a loop (each loop giving birth to a new generation of individuals), ideally evolving to an optimal solution. Each individual in the population is assigned a fitness value based on the total mass of the proposed solution and how well it conforms to the overall stress and maximum permissible displacement using FEM.

2.2 FEM

Two-dimensional trusses, or two-dimensional cartesian trusses, are essentially two-dimensional bars oriented in two-dimensional cartesian structures. To translate the local element matrices (stiffness matrix, force vector) into the structural (global) coordinate system, a transformation of coordinate basis is needed. Trusses, like bars, only support compressive and tensile forces. At the nodes, all forces are applied.

Truss with optimizing parameter is shown below:

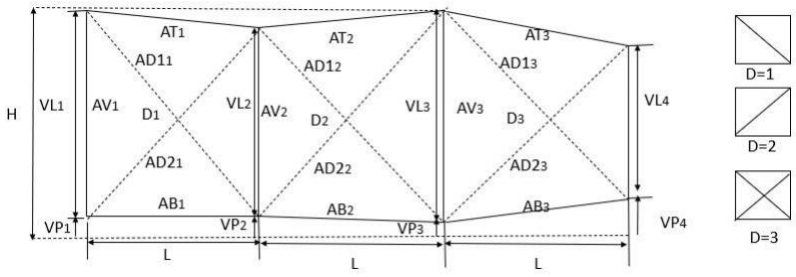

Figure 3: Truss structure to optimize

A variable number of spans n of adjustable size L x H are used to build the structure. Each span has eight parameters, which are denoted as follows (except for the (n+1) the span, which only has three parameters for the last vertical bar):

- VPi- Vertical position of the bottom node of the i-th span (between 0 and H-Lmin
- VLi- Length of the Verical bar in the i-th span (between Lmin and H)
- AVi- Cross-sectional area of the vertical bar of the i-th span (between Amin and Amax)
- ABi- Cross-sectional area of the Bottom bar of the i-th span (between Amin and Amax)
- ATi- Cross-sectional area of the Top bar of the i-th span (between Amin and Amax)
- Di- Diagonal value in the i-th bar (takes 1,2,3)
- AD1i- Cross-sectional area of the diagonal bar going down in the i-th span (between Amin and Amax)
- AD2i- Cross-sectional area of the diagonal bar going up in the i-th span (between Amin and Amax)

Work flow of the FEM model is shown below

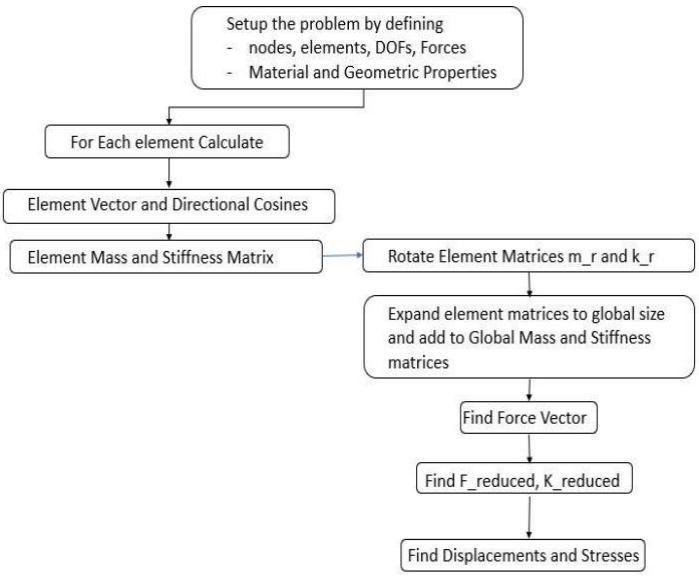

Figure 4: FEM program work flow

While the Genetic Algorithm and FEM seem to have a straightforward work flow, their implementation is quite difficult.

3 Methods and Implementation

3.1 FEM

3.1.1 Equations

The following equations are necessary to derive the governing equations of the Bar element: wave equation, Kinetic Energy, Strain Energy and Resultants

Wave equation:

$$[M]\ddot{u}(t) + [K]u(t) = F(t) \tag{1}$$

$$\ddot{u}(x,t) = C^2 u(x,t) \tag{2}$$

$$C^2 = E/\rho \tag{3}$$

Kinetic Energy:

$$T = \dot{\vec{q}}^T [m] \dot{\vec{q}} 1/2 \tag{4}$$

Strain Energy:

$$u = 1/2 \int_0^l EA{u_{,x}}^2(x,t)dx \tag{5}$$

Resutants :

$$[k] = EA \int_0^l \vec{\phi_{,x}} \vec{\phi_{,x}}^T dx \tag{6}$$

$$[m] = \rho A \int_0^l \vec{\phi}\vec{\phi}^T dx \tag{7}$$

$$\delta w = \int_0^l f(x,t)\delta u(x,t)dx \tag{8}$$

3.1.2 FEM formulation of Bar element:

Linear displacement equation is assumed:

$$u(x,t) = a_0 + a_1 X \tag{9}$$

After the element discretization the u(x,t) is in the following:

$$u(x,t) = \vec{\phi}(x)q(\vec{t}) \tag{10}$$

$$\vec{\phi}(x) = \begin{bmatrix} 1 - x/l \\ x/l \end{bmatrix} = \begin{bmatrix} \phi_1(x) \\ \phi_2(x) \end{bmatrix} \tag{11}$$

ϕ is the Shape function
The velocity is given by,

$$\dot{u}(x,t) = \vec{\phi}^T \dot{\vec{q}} \tag{12}$$

With these the mass matrix of the element is given by

$$[m] = \rho A \int_0^l \vec{\phi}\vec{\phi}^T dx \tag{13}$$

$$[m] = \rho A \int_0^l \begin{bmatrix} \phi^2 & \phi_1\phi_2 \\ \phi_1\phi_2 & \phi^2 \end{bmatrix} \tag{14}$$

$$[m] = \rho A l/6 \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \tag{15}$$

The Stiffness matrix of the element is given by

$$[k] = EA \int_0^l \vec{\phi_{,x}}\vec{\phi_{,x}}^T dx \tag{16}$$

$$\phi_{,x} = \begin{bmatrix} 1 - x/l \\ x/l \end{bmatrix} \tag{17}$$

$$[k] = EA \int_0^l \begin{bmatrix} \phi_{1,x}^2 & \phi_{1,x}\phi_{2,x} \\ \phi_{1,x}\phi_{2,x} & \phi_{2,x}^2 \end{bmatrix} \tag{18}$$

$$[k] = EA/l \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \tag{19}$$

After finding the element Stiffness and Mass matrices are extended to the Global coordinates by finding the rotation matrix for stiffness and the mass matrices

$$[M_G] = \sum_{i=1}^{no.of.ele} [m_{rG}] \tag{20}$$

$$[K_G] = \sum_{i=1}^{no.of.ele} [k_{rG}] \tag{21}$$

Where m_{rG} and k_{rG} are the matrices used to extend the elemental coordinates to the Global coordinates

$$[m_{rG}]_i = [B]_i^T [\tau_i]^T [mr_i][\tau_i][B_i] \tag{22}$$

$$[k_{rG}]_i = [B]_i^T [\tau_i]^T [kr_i][\tau_i][B_i] \tag{23}$$

where τ is the Transformation matrix

$$[\tau] = \begin{bmatrix} cos\theta & sin\theta & 0 & 0 \\ 0 & 0 & cos\theta & sin\theta \end{bmatrix} \tag{24}$$

The displacement and the Stress are calculated by using the formula mentioned below

$$[\vec{X}] = [K_G]^{-1}\vec{F} \tag{25}$$

$$\sigma = E\epsilon \tag{26}$$

3.2 Genetic Algorithm

The algorithm begins by creating a population of candidate solutions inside the design space at random (allowable parameters range). The GA operators of selection, crossover, and mutation are used to obtain the individuals in each new generation from the previous generation.

3.2.1 Encoding technique:

Trusses are encoded in chromosomes using a traditional technique that allows for simultaneous topology, shape, and size optimization. The phenotype, which is a parameterized structure, must be coded in the genotype. As a result, in the GA, each entity (candidate solution) is represented by its chromosome, which is a vector of genes, each of which contains the value of one parameter. (Razvan Cazacu etal.)

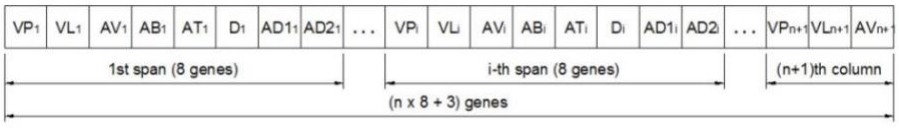

Figure 5: Gene arrangement in Chromosome

All of this data is put into an array and transferred to the optimizing function. By forming the full range of parameters from the fixed and optimized parameters, the algorithm then uses this array in the coding and decoding of the problem.

3.2.2 Fitness Evaluation:

This is the penalty function used for this Project:

$$PF = OF.(IP + P\,2((\sigma - \sigma_a)/\sigma_a)^2 \text{(Razvan Cazacu etal.)} \tag{27}$$

were

1. PF – penalized fitness,
2. OF – original fitness,
3. IP – initial penalty at the point of stress limit,
4. P2 –penalty scale factor at double the stress limit,

5. σ - actual maximum stress in the structure,
6. σ_a - maximum allowable stress for the chosen material.

For the stress restriction, the above function is used. Similar function for the displacement constraint, and the two penalties add up in the case of both stress and displacement constraint violations.

3.2.3 Genetic algorithm operators:

Parent Selection:

The GA operators are used to generate each new generation of individuals in the algorithm. Individuals from the older generation are chosen and used in replication (crossover) and mutation to create the new generation. These individuals are chosen at random but based on their scaled fitness, with a fitter individual having a higher probability of being chosen than a less fit one. The Roulette Wheel Selection method is the preferred method of selection, which simulates a roulette spin by creating a random number within a space where each individual occupies a portion proportional to its fitness.

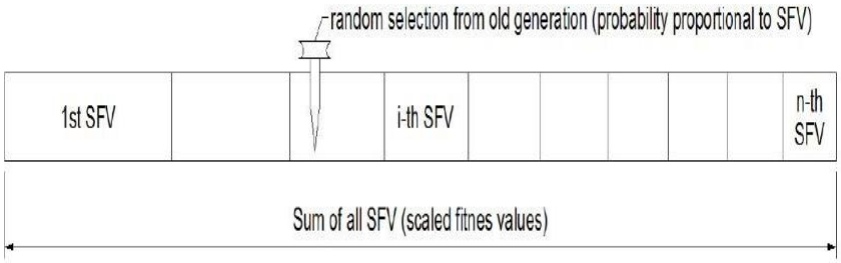

Figure 6: Parent Selection
(Razvan Cazacu etal.)

Elitism:

To retain the best entities, the algorithm employs elitism. This ensures that a portion of them will pass on to the next generation unaltered. Elite count of 2, which means that the two fittest individuals would almost certainly be found in the next generation, where they maybe replaced by even fitter children.

Crossover:

Two-point crossover is used to create a new generation for two chosen parents. For crossover, two integer values between 1 and the chromosome size are chosen at random. The algorithm uses these two values to recombine the genes of the two parents to produce two children, as seen in the image.

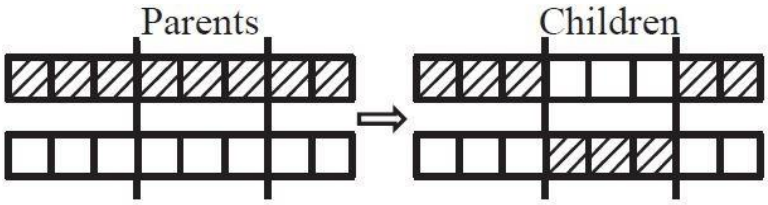

Figure 7: Crossover operation
(Razvan Cazacu etal.))

Mutation:

For each entity in the current population chosen for mutation, classic uniform mutation is used, and each gene has a probability pm of being mutated. A new value for the gene is then selected at random from a uniform distribution over the gene's possible range.

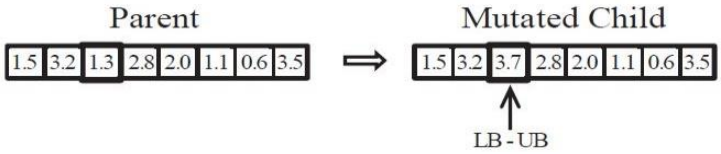

Figure 8: Mutation
(Razvan Cazacu etal.)

Genetic Algorithm Pseudocode:

```
Begin:
 Initial Population
 for (each Chromosome)
        FEM calculation
        Stress and Displacement Evaluation
 end for
 Penalized Fitness Calculation for each chromosome in the Population
 Fitness Scaling for Parent Selection
 set generation_number=0
 print the Population with their Fitness
 set generation number =1
 while (Until Stopping Criteria)
        Parent Selection (Roulette wheel)
        for (2)
               Generate Crossover points
        end for
        Two Point Crossover
        Mutation (Uniformly mutated Chromosome based on probability of mutation)
        Elitism (Count 2)
        Generate new population
        for (each Chromosome)
               FEM calculation
               Stress and Displacement Evaluation
        end for
        Penalized Fitness Calculation for each Gene in the Population
        Fitness Scaling for Parent Selection
 generation_number+=1
 end while
end
```

Figure 9: Genetic Algorithm

4 Programming Language and Libraries Used:

Implementation of the above program is done in Python with libraries Numpy and Matplotlib

5 Results and Discussions:

In this program, I must reduce the truss weight by optimizing the truss parameters while keeping tension and strains within acceptable limits. All the parameter of the truss are the optimizing parameter and for the computational time, I have selected the following material and GA parameters and stopping criteria with 40 iterations and a single constraintproblem

Parameters	Inputs
number of parameters	8
number of span	2
Maximum Length of the bar	18
Maximum Height of the bar	9
min value of area	5
max value of area	10
population size	40
number of constraints (stress (1) / strain (2) or both (3)	1
density of the element	0.1
stiffness of the element	10
Stress limit	25
Probability of mutation	0.6
size of restrained dofs	4
number of restrained dofs in the truss	1
number of restrained dofs in the truss	2
number of restrained dofs in the truss	3
number of restrained dofs in the truss	4
Forces[1] x	100
Forces[1] y	200
Forces[3] x	100
Forces[3] y	200
Forces[5] x	100
Forces[5] y	200

The results of the best solution is stored and the plot is shown below:

Figure 10: Optimized solution

A plot with the fitness value and the iteration is given below:

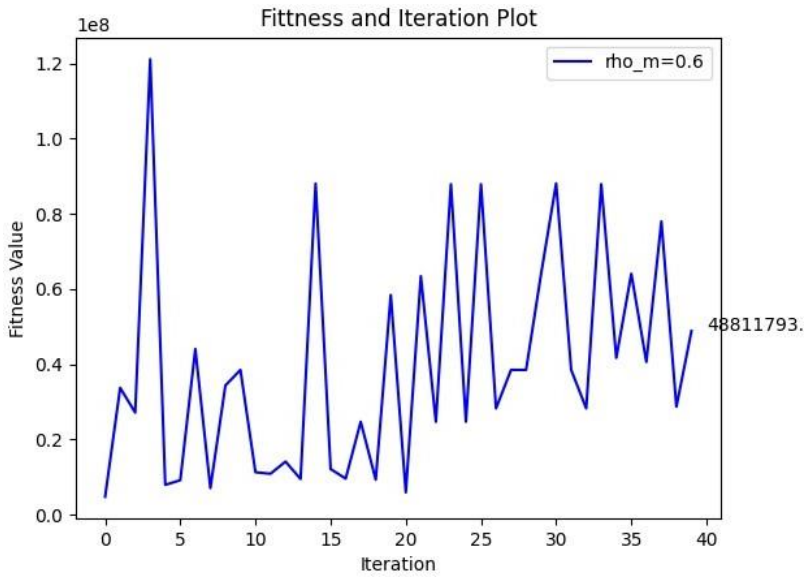

Figure 11: plot with fitness value and number of iterations

From the above graph it is clear that the Penalized fitness of the truss is minimized with number of iterations. For the computational time I have selected a population size of 40 with mutation probability 0.6. lower the mutation probability means there is higher chances of each gene in the chromosome to get mutated.

6 Testing:

Testing is the process of comparing the program's input to the expected output. I conducted Testing for the Genetic algorithm and the FEM-model here.

6.1 FEM:

6.1.1 Test Case:01

Rigid body Translation, With the FEM truss structure all the constraints were removed and Forces are applied in all the node in the structure. With these conditions, The Global Stiffness matrix will be a Singular matrix, which determines, no stresses are developed in thenodes.
The material parameter is given below:

Parameters	**Inputs**
number of parameters	8
number of span	2
Maximum Length of the bar	18 m
Maximum Height of the bar	9 m
Density	1 kg/m^3
Youngs Modulus	70000 Pa
Restrained DOF	0
Forces in all nodes	100 N

The below figure shows that the program shows the Stiffness matrix is singular

```
PS E:\Masters\Academics\PPP\TOPICS\Optimization\Programs> & C:/Users/nivas/AppData/Local/Progr
ams/Python/Python37/python.exe e:/Masters/Academics/PPP/TOPICS/Optimization/Programs/Genetic_a
lgorithm_for_rigid_body_translation.py
Enter the number of parameters in each bar 8
Enter the number of span in the truss structure 2
Enter the length of the bar 18
Enter the height of the bar 9
Enter the density of element 1
Enter the stiffness of element 70000
Input array is
  [[0, 18, 300, 300, 300, 3, 300, 300], [0, 18, 300, 300, 300, 3, 300, 300], [0, 18, 300, 0, 0
, 0, 0, 0]]
Enter the size of the restrained dofs array 0
Enter the value of force vector in 0 in X dir 100
Enter the value of force vector in 0 in Y dir 100
Enter the value of force vector in 1 in X dir 100
Enter the value of force vector in 1 in Y dir 100
Enter the value of force vector in 2 in X dir 100
Enter the value of force vector in 2 in Y dir 100
Enter the value of force vector in 3 in X dir 100
Enter the value of force vector in 3 in Y dir 100
Enter the value of force vector in 4 in X dir 100
Enter the value of force vector in 4 in Y dir 100
Enter the value of force vector in 5 in X dir 100
Enter the value of force vector in 5 in Y dir 100
Determinat of K before reducing 0.0
Determinant of K after reducing  0.0
```

Figure 12: Rigid body Translation

6.1.2 Test Case: 02

a) I got the stress and the Displacements from a scientific paper (Matlab Codes for Finite Element Analysis FEM Analysis of 2D trusses Chapter 4, Page No: 62) and checked with my FEM program whether same results are generated for a specific material specification. The material parameter is given below:

Parameters	**Inputs**
number of parameters	8
number of span	2
Maximum Length of the bar	3000 mm
Maximum Height of the bar	3000 mm
Density	1 kg/m^3
Youngs Modulus	70000 Pa
Restrained DOF	3 [1,2,10]
Forces[4]	-50000 N
Forces[8]	-50000 N
Forces[12]	-50000 N

The results from the FEM program is shown below:

```
PS E:\Masters\Academics\PPP\TOPICS\Optimization\Programs> & C:/Users/nivas/AppData/Local/Programs/Python/Pytho
n37/python.exe e:/Masters/Academics/PPP/TOPICS/Optimization/Programs/Genetic_Algorithm_FEM_verification.py
Enter the number of parameters in each bar 8
Enter the number of span in the truss structure 2
Enter the length of the bar 3000
Enter the height of the bar 3000
Enter the density of element 1
Enter the stiffness of element 70000
Input array is
  [[0, 3000, 300, 300, 300, 3, 300, 300], [0, 3000, 300, 300, 300, 3, 300, 300], [0, 3000, 300, 0, 0, 0, 0, 0]
]
Enter the size of the restrained dofs array 3
Enter the numbers of the restrained dofs in the truss 1
Enter the numbers of the restrained dofs in the truss 2
Enter the numbers of the restrained dofs in the truss 10
Enter the value of force vector in 0 in X dir 0
Enter the value of force vector in 0 in Y dir 0
Enter the value of force vector in 1 in X dir 0
Enter the value of force vector in 1 in Y dir -50000
Enter the value of force vector in 2 in X dir 0
Enter the value of force vector in 2 in Y dir 0
Enter the value of force vector in 3 in X dir 0
Enter the value of force vector in 3 in Y dir -100000
Enter the value of force vector in 4 in X dir 0
Enter the value of force vector in 4 in Y dir 0
Enter the value of force vector in 5 in X dir 0
Enter the value of force vector in 5 in Y dir -50000
Determinat of K before reducing -18.904960966374144
Determinant of K after reducing  2.326759571826466e+35
Global_displacements are
 [  0.           0.           7.14285714  -9.03863712   5.24707717
 -16.2964926    5.24707717 -20.08805255  10.49415433   0.
   3.35129719  -9.03863712]
Global strains are
 [-0.00301288  0.00174903 -0.00063193  0.00089368 -0.0024735  -0.00126385
  0.00174903 -0.00063193 -0.0024735   0.00089368 -0.00301288]
Total time elasped 16.140482187271118 in seconds
PS E:\Masters\Academics\PPP\TOPICS\Optimization\Programs>
```

Figure 13: Result from FEM program

The below graph shows the curve fit of displacements at each node and the stresses in each element in FEM program and the Paper.

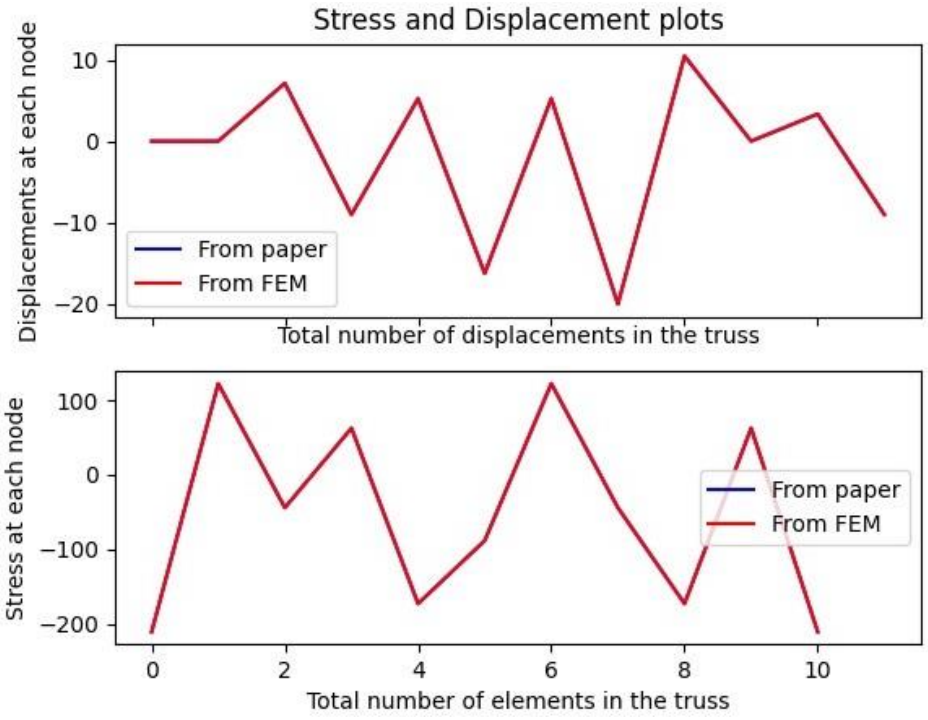

Figure 14: Result from FEM program

6.1.3 Test Case: 03

FEM program is compared with the Manually solved results with specific material parameters
With span size 2

Parameters	Inputs
number of parameters	8
number of span	2
Maximum Length of the bar	18 m
Maximum Height of the bar	9 m
Density	0.284 kg/m^3
Youngs Modulus	3e7 Pa
Restrained DOF	4 [5,6,9,10]
Forces[3]	100 N
Forces[4]	-200 N
Forces[7]	200 N
Forces[8]	-400 N
Forces[11]	300 N
Forces[12]	-600 N

Manual Calculation is done in Excel and the work flow of the calculation is mentioned below:

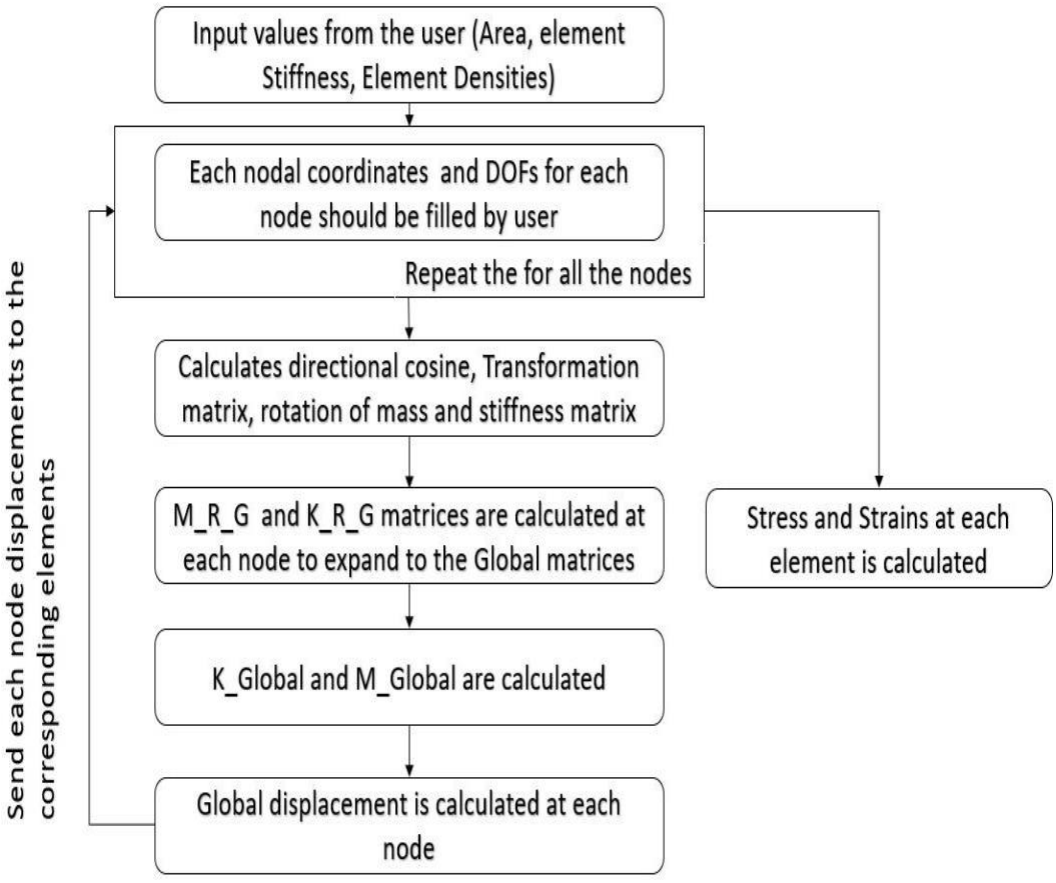

Figure 15: Work flow of the Manual Calculation

The results from the Manual Calculation and the FEM program are compared and the Stress and strain plot is shown below:

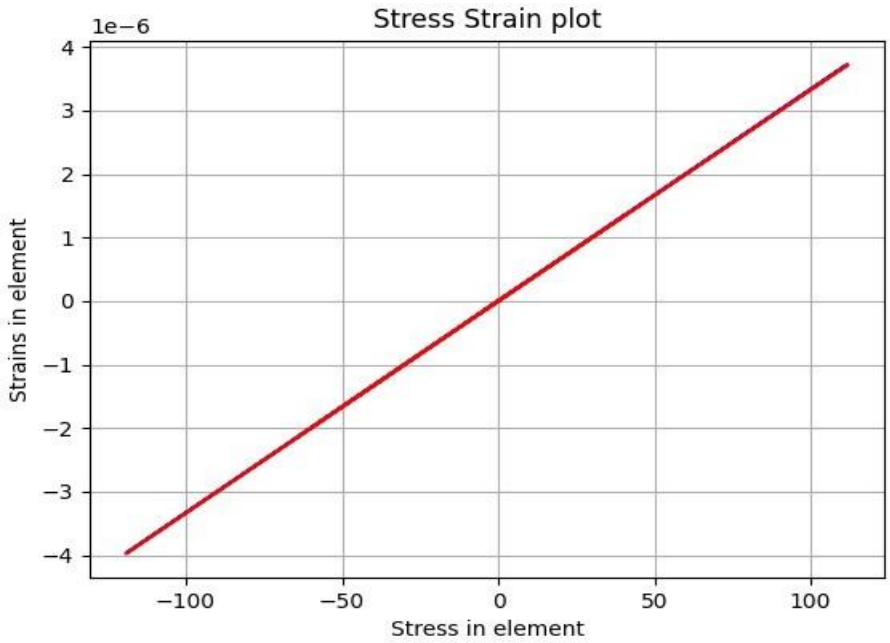

Figure 16: FEM program and Manual calculation stress and strain graph

To Justify the FEM, the curve fits with the Manual calculation and the result from the paper. So, FEM program is valid.

6.2 Genetic Algorithm:

6.2.1 Test Case: 04

I have taken a simple problem to find two parameter which maximize the function with constrains and checking with the algorithm whether the algorithm optimizes the function with the constraints and gives the same results. The optimization is compared with the SciPy and the GA algorithm.

function to maximize: $x_1 * x_2$

constraints: $20 - 2x_1 - 2x_2$

With population size of 40 and varying the probability of mutation the following result is obtained

mutation probability	Output and number of iterations	Expected Output
0.6	5,5 216	5,5
0.7	5,5 233	5,5
0.8	5,5 81	5,5

The fitness value of each chromosome in the population, as well as the number of iterations needed to find the best solution, are depicted in the graph below:

When comparing the results obtained by varying the mutation probability, the number of iterations is decreased if the mutation probability is higher, indicating that a higher probability value means further crossover and allowing the algorithm to find the best solution sofar.

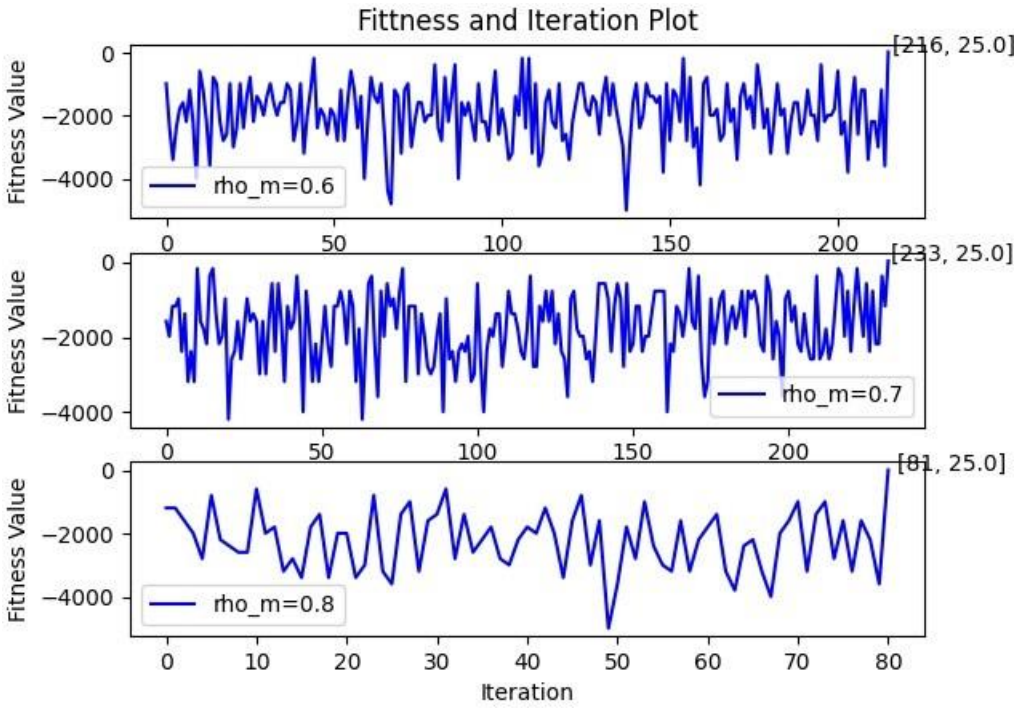

Figure 17: Fitness and Number of iterations with varying mutation probability

6.2.2 Test Case: 05

All of the parameters do not have to be optimized for every problem.
To validate the encoding technique, the algorithm is applied to one of the most common benchmark problems for plane truss optimization. Only the sectional areas are subject to optimization, as node locations and bar connections are fixed. It's assumed that the bars are made of aluminum. The problem's static scheme is given below:

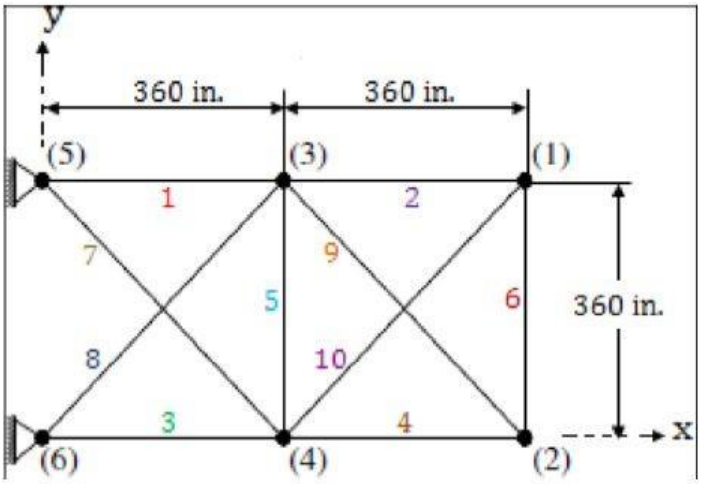

Figure 18: Benchmark Truss problem
Baykasoğlu Adil1 etal)

The material parameter for the Benchmark problem is mentioned below:

Parameters	Inputs
number of parameters	8
number of span	2
Maximum Length of the bar	360
Maximum Height of the bar	360
Minimum value of area	0.1
Maximum value of area	35
Population size	50
Number of constraints	3
Density	0.1
Youngs Modulus	10
Stress limit	25
Strain limit	2
Mutation probability	0.6
size of restrained dofs	4
number of restrained dofs in the truss	1
number of restrained dofs in the truss	2
number of restrained dofs in the truss	3
number of restrained dofs in the truss	4
Forces[6]	-100
Forces[10]	-100

The results are compared with the (Baykasoğlu Adil1 etal) in page. No 969 and changing the number of generation and the population sizes are varied with varying mutation probability. **By varying the Iterations, Population size and the mutation probability:**
Iteration value of 40:

6.2.3 Test Case :a

With a iteration value of 40 and probability of mutation 0.6 and population size of 50 for the truss problem is optimized and results are stored.

6.2.4 Test Case :b

With a iteration value of 40 and probability of mutation 0.7 and population size of 50 for the truss problem is optimized and results are stored.

6.2.5 Test Case :c

With a iteration value of 40 and probability of mutation 0.8 and population size of 50 for the truss problem is optimized and results are stored.

In the below graph, Iteration value of 40, with the Population size 50 and varying the mu- tation probability is shown

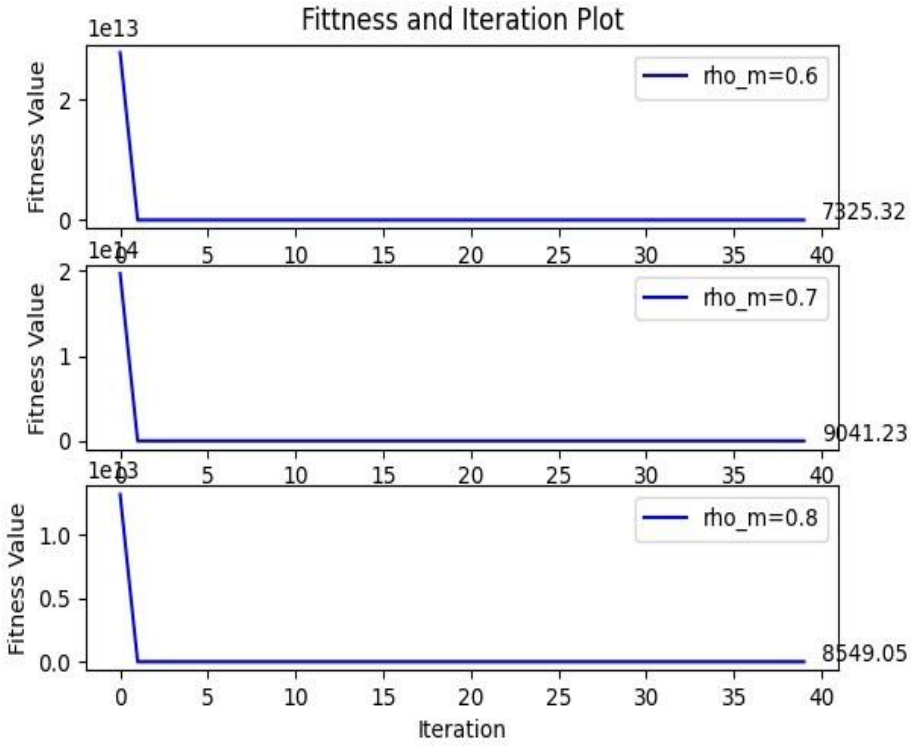

Figure 19: Iteration 40, population size 50 with varying mutation probability

6.2.6 Test Case :d

With a iteration value of 40 and probability of mutation 0.6 and population size of 80 for the truss problem is optimized and results are stored.

6.2.7 Test Case :e

With a iteration value of 40 and probability of mutation 0.7 and population size of 80 for the truss problem is optimized and results are stored.

6.2.8 Test Case :f

With a iteration value of 40 and probability of mutation 0.8 and population size of 80 for the truss problem is optimized and results are stored.

In the below graph, Iteration value of 40, with the Population size 80 and varying the mutation probability is shown

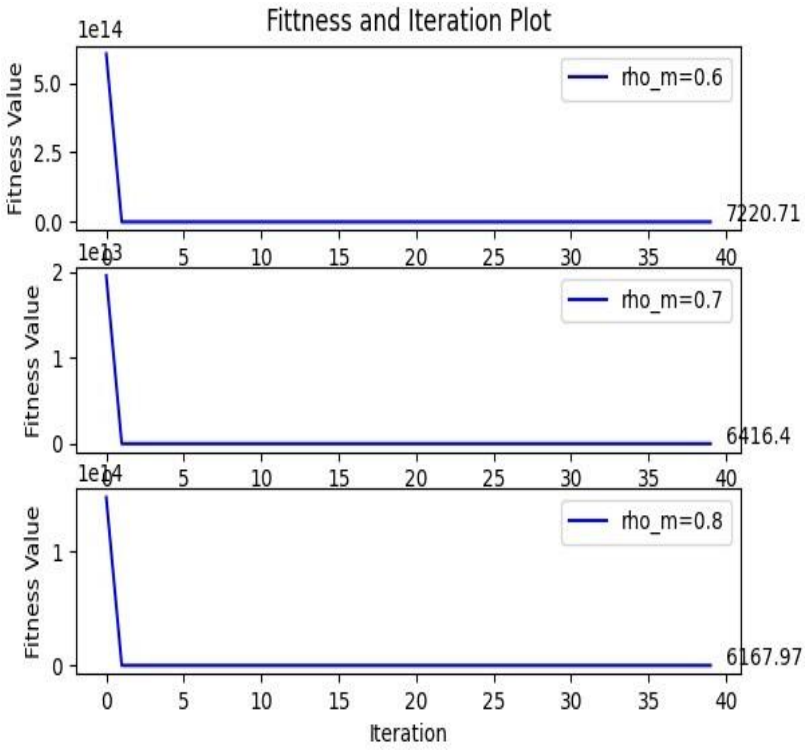

Figure 20: Iteration 40, population size 80 with varying mutation probability

6.2.9 Test Case :g

With a iteration value of 40 and probabilty of mutation 0.6 and population size of 100 for the truss problem is optimized and results are stored.

6.2.10 Test Case :h

With a iteration value of 40 and probabilty of mutation 0.7 and population size of 100 for the truss problem is optimized and results are stored.

6.2.11 Test Case :i

With a iteration value of 40 and probabilty of mutation 0.8 and population size of 100 for the truss problem is optimized and results are stored.

In the below graph, Iteration value of 40, with the Population size 100 and varying the mutation probability is shown

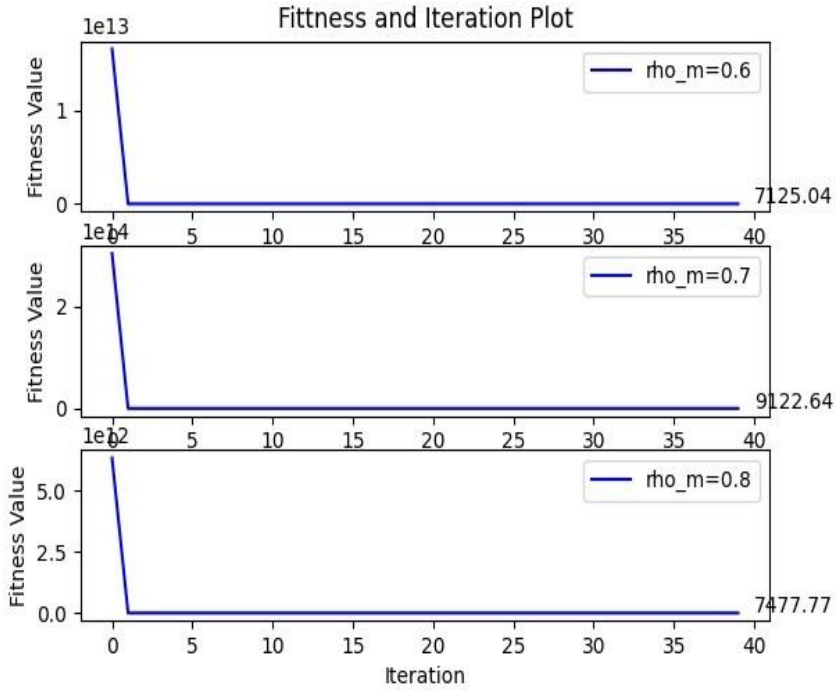

Figure 21: Iteration 40, population size 100 with varying mutation probability

Iteration value of 80:

Iteration value of 80, with varying the Population size and the mutation probability is shown in the below graph:

6.2.12 Test Case :j

With a iteration value of 80 and probabilty of mutation 0.6 and population size of 50 for the truss problem is optimized and results are stored.

6.2.13 Test Case :k

With a iteration value of 80 and probabilty of mutation 0.7 and population size of 50 for the truss problem is optimized and results are stored.

6.2.14 Test Case :l

With a iteration value of 80 and probabilty of mutation 0.8 and population size of 50 for the truss problem is optimized and results are stored.

In the below graph, Iteration value of 80, with the Population size 50 and varying the mu- tation probability is shown

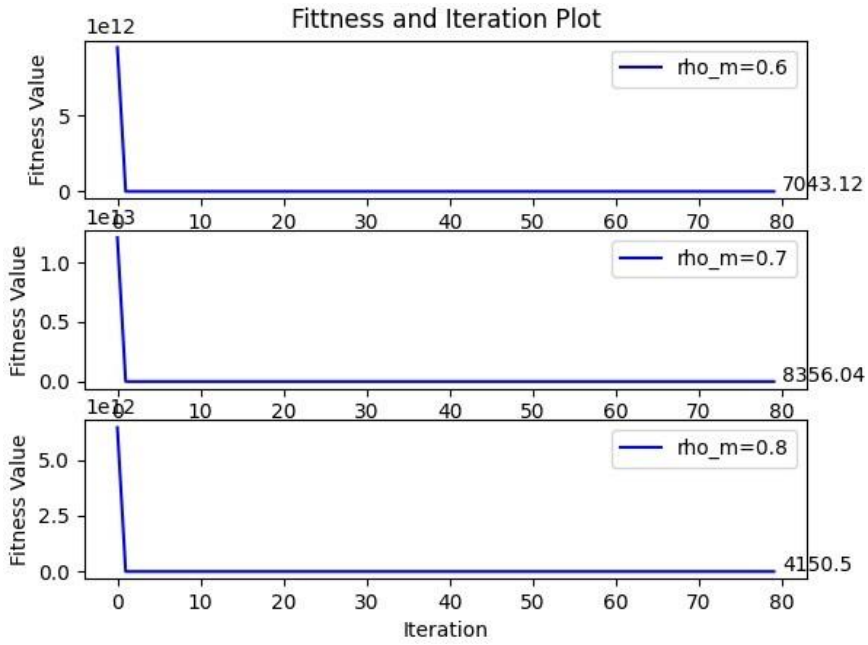

Figure 22: Iteration 80, population size 50 with varying mutation probability

6.2.15 Test Case :m

With a iteration value of 80 and probability of mutation 0.6 and population size of 80 for the truss problem is optimized and results are stored.

6.2.16 Test Case :n

With a iteration value of 80 and probability of mutation 0.7 and population size of 80 for the truss problem is optimized and results are stored.

6.2.17 Test Case :o

With a iteration value of 80 and probability of mutation 0.8 and population size of 80 for the truss problem is optimized and results are stored.

In the below graph, Iteration value of 80, with the Population size 80 and varying the mutation probability is shown

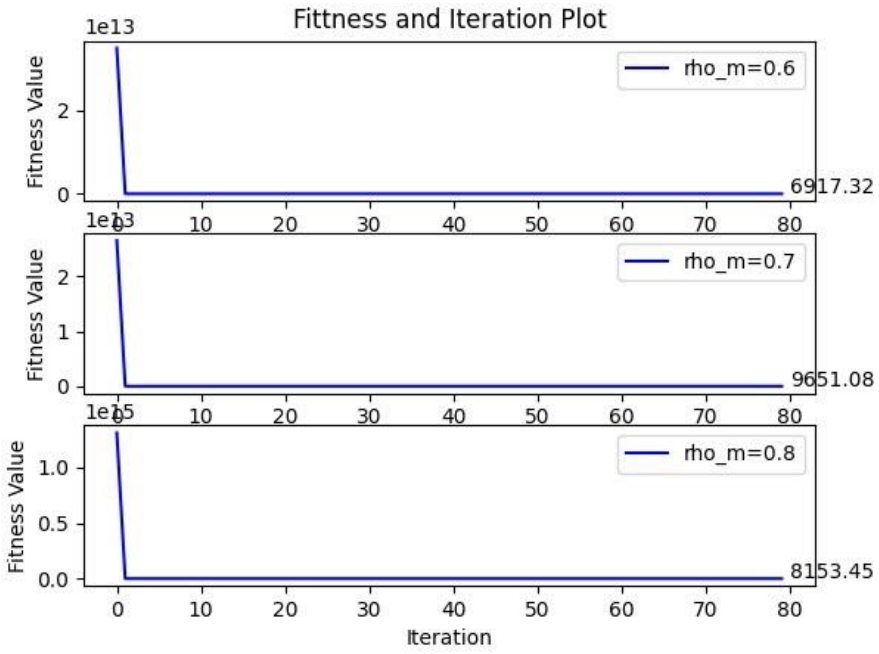

Figure 23: Iteration 80, population size 80 with varying mutation probability

6.2.18 Test Case :p

With a iteration value of 80 and probability of mutation 0.6 and population size of 100 for the truss problem is optimized and results are stored.

6.2.19 Test Case :q

With a iteration value of 80 and probability of mutation 0.7 and population size of 100 for the truss problem is optimized and results are stored.

6.2.20 Test Case :r

With a iteration value of 80 and probability of mutation 0.8 and population size of 100 for the truss problem is optimized and results are stored.

In the below graph, Iteration value of 80, with the Population size 100 and varying the mutation probability is shown

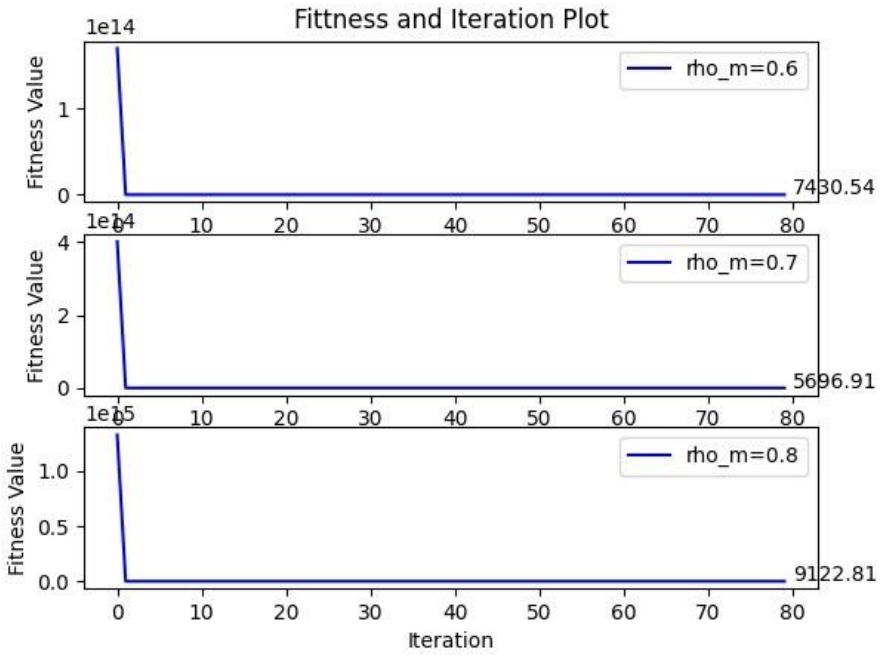

Figure 24: Iteration 80, population size 100 with varying mutation probability

The solutions are compared to the paper's findings, which provide the Penalty function values (IP and P2 values are selected based on trail and error method). There is a percentage of error with the results in the paper due to the Penalty parameter values chosen.

According to the graph above, as population size and mutation probability increase,
A lower value of mutation probability means more mutation, requiring the algorithm to dis- cover new areas of the design space, while a higher value means more crossover, enabling the algorithm to further optimize locally the solutions so far.

A larger population with a higher mutation probability is favored for a faster optimal solution. (For a population size of 100, iteration 40, and a mutation probability of 0.8, the truss weight is optimized with maximum stress within the permissible range.)

6.3 Functionality Tests (Unit test):

Unit testing is a software testing process that involves putting individual units of source code through a series of tests to see whether they are ready for use (Source). It assesses and determines the code's consistency.
For this program, I have conducted unit test for the below mentioned functions

Funciton
get all elements in input array
dir cosine
transformation matrixr
single pt crossover
multi pt crossover
scaled fitness
force vector calc
get restrained dofs

Because of the random value generation in the functions (such as randomly mutate pop and GA algorithm), I was unable to perform the unit test.

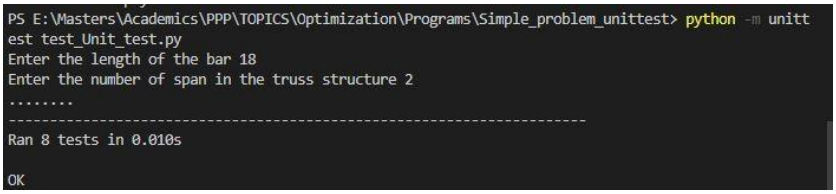

Figure 25: Functionality Test

7 Manual:

In the folder **"Main program"** it has the file named **"Genetic algorithm.py"**

1. Open a terminal window and type the correct path to the file.
2. Type "python" or "python3" based on the python version using, and type **"Genetic algorithm.py"**.

Figure 26: Programming path

3. Press "Enter"
4. The program will ask to enter the parameters, for example see Figure 20

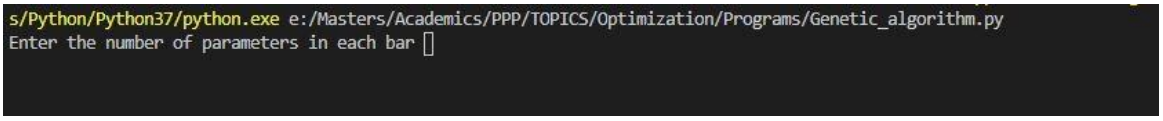

Figure 27: Inputs from user

5. User have to give the material parameters for FEM and the parameters for the Genetic algorithm, for example see Figure 21 and Figure 22.

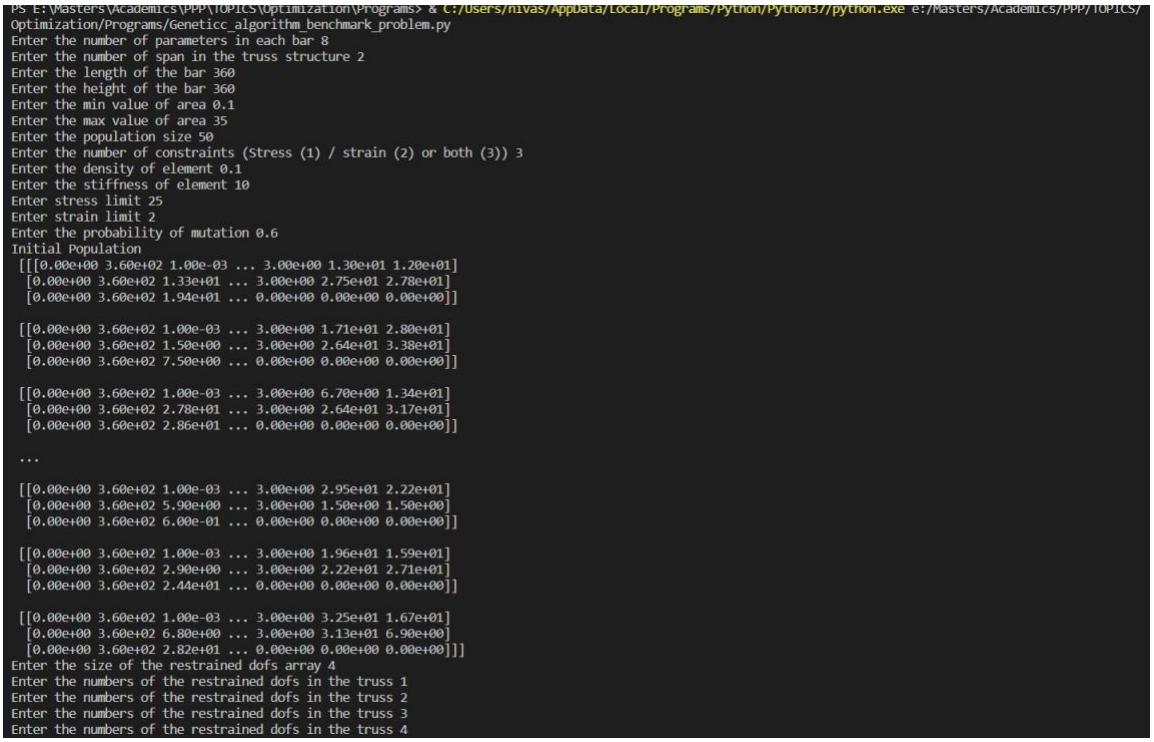

Figure 28: Material and GA parameters

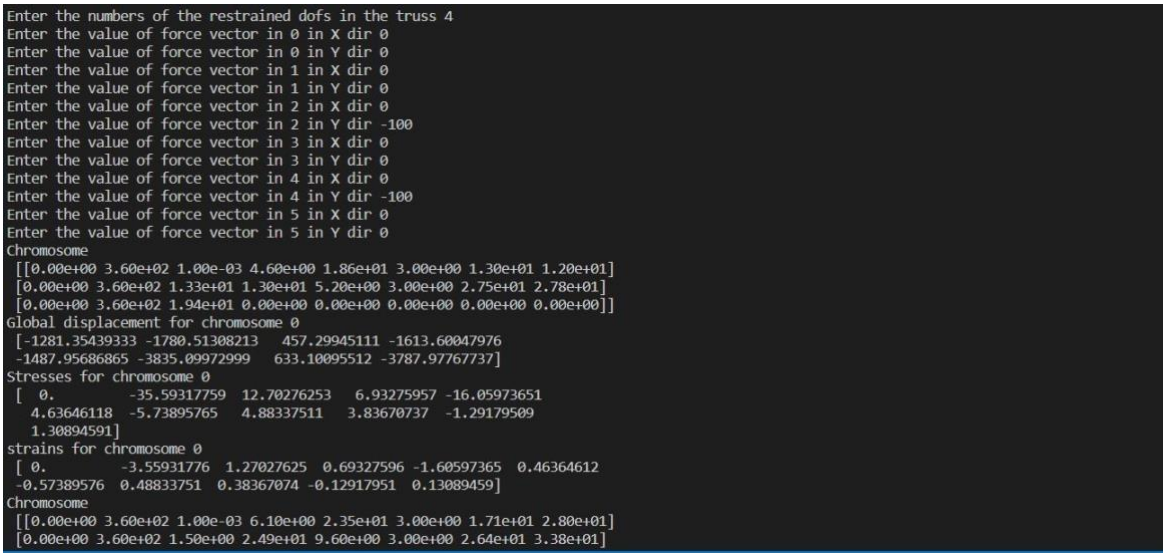

Figure 29: Material and GA parameters

6. The output of the program is saved in the **"Genetic algorithm.txt"** and **"best solution.txt"**. Open the text file for the detailed information.

Remarks:

All the values entered should be numbers; if they aren't, an error message will be shown.

Error:

An error message will appear in the window if the user attempts to enter a value that is not a number and the program will be terminated. See Figure 24

```
PS E:\Masters\Academics\PPP\TOPICS\Optimization\Programs> python .\Genetic_algorithm.py
Enter the number of parameters in each bar a

 Only integer values are valid
PS E:\Masters\Academics\PPP\TOPICS\Optimization\Programs>
```

Figure 30: Error message

Suggestions for GA algorithm parameters due to computational time and other material parameters are dependent on the problem to be solved.

Population size	50-100
Mutation probability	0.6-0.8

8 Literature:

b) Razvan Cazacu*, Lucian Grama Steel truss optimization using genetic algorithms and FEA

c) Baykasoğlu Adil1 · Baykasoğlu Cengiz2 Baykasoğlu Adil1 · Baykasoğlu Cengiz2

d) Matlab Codes for Finite Element Analysis FEM Analysis of 2D trusses

9 Git log

Final commit

commit 24620abd2f8dcf24c574f29a87e99fc5ac3598aa
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Thu Apr 15 16:22:48 2021 +0200

Final commit

commit 0e1d8359aa88bc9b853d3a135dcecf9d73deea3f
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Thu Apr 15 13:58:56 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_100_&_40_iterations_0.8
:...skipping...
commit fad692856dd5fb12e5fa79e627d20465ae801a33 (HEAD -> master, origin/master)
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Thu Apr 15 18:28:10 2021 +0200

Functionality test

commit 3a72af7a83a3865348eafa9c3099d05e7baf34f5
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Thu Apr 15 18:27:40 2021 +0200

Functionality test

commit 8495321928b2d250d54d85d31d3ae0dc995ba84d
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Thu Apr 15 16:25:55 2021 +0200

Final commit

commit 8292de355ad4ce9b544796f5e16c2b7005fd2d6f
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Thu Apr 15 16:25:09 2021 +0200

Final commit

commit 658b9724b6fe9b7fd59801bde652764076dcae39
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Thu Apr 15 16:23:51 2021 +0200

Final commit

commit 24620abd2f8dcf24c574f29a87e99fc5ac3598aa
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Thu Apr 15 16:22:48 2021 +0200

Final commit

commit 0e1d8359aa88bc9b853d3a135dcecf9d73deea3f
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Thu Apr 15 13:58:56 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_100_&_40_iterations_0.8

commit 2b25b90b198dc6dbf416bd4a715d885b39ad1024
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Thu Apr 15 13:03:33 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_100_&_40_iterations_0.6

commit 96efe4fdecc3d89b8babf765577018779d21f932
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Wed Apr 14 21:16:56 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_50_&_100_iterations_0.8

commit 3cdd3145fc2ca1b7a56d6e086da18d7cbde99c7e
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Wed Apr 14 19:28:42 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_50_&_100_iterations_0.7

commit d69a78376b8b6bbff64a70d34310f27fc814961d
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Wed Apr 14 18:40:19 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_50_&_100_iterations_0.6

commit 137cfa73b4e37293292d9d93b9fdc115993505ca
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Wed Apr 14 17:54:33 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_100_&_80_iterations_0.8

commit 736c97cab7b2ab94f59371024c4320125530a721

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Wed Apr 14 16:20:18 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_100_&_80_iterations_0.7

commit f95d1d3958df151b19b5db9ce783cf19291944de

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Wed Apr 14 14:46:16 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_100_&_80_iterations_0.6

commit d37fa55c4336b4936a6a1b308c92c88d4b9a3203

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Wed Apr 14 06:14:30 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_80_&_80_iterations_0.8

commit d5801a258dc07601032bfbda6d733225312168e4

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Wed Apr 14 05:17:10 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_80_&_80_iterations_0.7

commit 888e57f6b3594ead04d2b589253230ce7a0e2233

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 23:56:32 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_80_&_80_iterations_0.6

commit 21502817d007b73bf92f43dd7a22387e2d9e637d

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 22:51:51 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_50_&_80_iterations_0.8

commit 731ae7a904ab489daf62baff5314d60635668f7b

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 21:30:07 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_50_&_80_iterations_0.7

commit 175fe3be29928f28152aa126ee549f76a192ff04

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 21:05:27 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_50_&_80_iterations_0.6

commit ab5fe86d64c28649bf826af02c4ceb1a4cd5f29b

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 20:33:50 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_100_&_40_iterations_0.8

commit b597a4a01f6ead7b630c424526f6bdfe3c847588

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 20:06:47 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_100_&_40_iterations_0.7

commit f3d13e817213cbec36a0779513bfce15d186e4a0

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 19:32:21 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_80_&_40_iterations_0.8

commit 448c0d76a8d4c5985db4bdb58a73e7a6af94707f

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 19:14:37 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_80_&_40_iterations_0.7

commit 43f1b7673f656ce74bc4f91e9a4c8d62042396be

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 18:57:53 2021 +0200

Genetic_algorithm_changes in truss structure creation

commit aeb456bb2e0ebd810fdbe490135cc0a4c5d919c1

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 18:55:53 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_80_&_40_iterations_0.6

commit b0ced6bb45f7995c0826e8329536900c02d5c43c

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 18:36:06 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_50_&_40_iterations_0.8

commit da602c481a4ca0672291798631f30da8ff4b1568

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 18:26:46 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_50_&_40_iterations_0.7

commit ff2c0c3aa906da6ca2cdf3f3ed376a1cdecd96f4

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 06:22:22 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_50_&_40_iterations_0.6

commit e8c3d4794fbafeaa746a2d2aff73a111d69c8841

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 05:26:58 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_50_&_40_iterations_0.8

commit a127346084ce920471c3c8e10ee702b3e5c8ff99

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 05:26:19 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_50_&_40_iterations_0.7

commit a4b7291031d29c0f7f1a55fbab502ab25a9967b4

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 05:25:25 2021 +0200

Genetic_algorithm_benchmark_prblm_with_Population_size_50_&_40_iterations_0.6

commit b728c914d5dba03c7920dcbc166cb7631c584304

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 03:54:14 2021 +0200

Fitness_value_and_iteration_plot_for_Population_size_100_with_varying_mutation_probability

commit 8b316c23dce29f0240a0655c8db68558bd9c90a3

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 03:52:59 2021 +0200

Fitness_value_and_iteration_plot_for_Population_size_80_with_varying_mutation_probability

commit 906ec4ab69e8d007dcd520b728c25077e46e5bac

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 03:51:14 2021 +0200

Fitness_value_and_iteration_plot_for_Population_size_40_with_varying_mutation_probability

commit c65e1cb3cfe456591bd87cce4e08aa048455a1a9

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 03:48:15 2021 +0200

Genetic_algorithm_with_FEM Verification draw line_function changes in paper verification

commit 57895c2f5698e99a5b6eab86b5611e226ab6981f

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 03:47:04 2021 +0200

Genetic_algorithm_with_FEM Verification draw line_function changes for excel

commit 48d328a6d89dc1a7a8630666e370cee72fdb875c

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 03:33:04 2021 +0200

Genetic_algorithm_with_FEM Verification draw linw function changes

commit cffb9224a169ecf83c72b6c1d5cd1fd6981985a1

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 03:26:38 2021 +0200

Genetic_algorithm_with_Population_size_100_0.8

commit 31f48da0572440e4bfe51bd8d8bc2f8a27730cd3

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 03:25:11 2021 +0200

Genetic_algorithm_with_Population_size_100_0.7

commit 31872a574eed10486c569ac814400b4f0b01769f

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 03:23:20 2021 +0200

Genetic_algorithm_with_Population_size_100_0.6

commit 4ccd3cc9b293b5ea4cc4c506f724c4fa50b73bb7

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 03:21:53 2021 +0200

Genetic_algorithm_with_Population_size_80_0.8

commit d9bdb4f5b8f841a95cde703963a88a88c318dc53
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Tue Apr 13 03:20:49 2021 +0200

Genetic_algorithm_with_Population_size_80_0.7

commit 5ba9d89ceb13d57956f799f46f1ce748c4dc58e7
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Tue Apr 13 03:19:59 2021 +0200

Genetic_algorithm_with_Population_size_80_0.6

commit d63b711c4536d8f74e43cf9ef98d026aaa24c82b
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Tue Apr 13 03:18:53 2021 +0200

Genetic_algorithm_with_Population_size_40_0.8

commit 4d69c1b42f28dd62455ccf40b72501a4f33fc77f
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Tue Apr 13 03:10:07 2021 +0200

Genetic_algorithm_with_Population_size_40_0.7

commit 7958bb02d4e2ed7d392198c652cf11071de2ebfe
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Tue Apr 13 03:08:44 2021 +0200

Genetic_algorithm_with_Population_size_40_0.6

commit 0c1a7cdd6cdea64bccfe3b1ce9a034689efe36bc

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 02:33:50 2021 +0200

User value error implemented in FEM verification

commit 7dc44efe0d93ff0d6fa6b5a89a19e585fb3633ea

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 02:32:19 2021 +0200

User value error implemented in mutation

commit 470a8b8ddcf2e904d95ff0266aa13194cc28ae7d

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 02:28:34 2021 +0200

User value error implemented

commit b59318780e89affdd8ef17119a3b7d8e815e0b5d

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 02:20:30 2021 +0200

FEM verification with error handling

commit ed74d78d45431fdbf870fb6697361518c2bf796a

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 02:16:40 2021 +0200

FEM verification for rigid body translation

commit a35c7de249e5d3481ab396f8c403c1e49374d54d

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 01:00:20 2021 +0200

FEM verification with excel sheet_1

commit 02b0b1e046a3714c3e446f79bd3d40df025bfe4a

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 00:45:05 2021 +0200

FEM verification with excel sheet

commit a3751746d978dafdbf35a559629b2540df97e425

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Apr 13 00:20:16 2021 +0200

Code documentation and verification of FEM

commit 540d80030a16cad5b02463a4627331ea6989f71f

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Mon Apr 12 22:52:08 2021 +0200

Code documentation

commit e753a7dfe3c6cc3df23f47fbd6f75a66fc35eb8e

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Mon Apr 12 15:38:27 2021 +0200

Fitness and iteration curve for benchmark problem with varying generation size, number of iteration and mutation probability

commit 5488a61a70e9183c1a493ad85bb5206514de1bf0

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Mon Apr 12 00:48:27 2021 +0200

Checking the weight of the structure with the paper

commit 8bb24948f3a22d2b4e8ec88d292332cb0be92dc1

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Sun Apr 11 21:19:58 2021 +0200

Changes in the function and GA function changes

commit f3f7cd09bbb5770943ece419cc4b310767ba8846

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Sun Apr 11 21:18:19 2021 +0200

Changes in the function and GA

commit bfea7031a80c184237afa21757fcd787a5c53434

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Sun Apr 11 21:16:46 2021 +0200

Changes in the definition of functions

commit 68c22df332a12c4a208dc5fab5066b3ed2d44d8f

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Sun Apr 11 18:39:17 2021 +0200

Fitness and iteration curve for benchmark problem with varying generation size, number of iteration and mutation probability

commit 5488a61a70e9183c1a493ad85bb5206514de1bf0

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Mon Apr 12 00:48:27 2021 +0200

Checking the weight of the structure with the paper

commit 8bb24948f3a22d2b4e8ec88d292332cb0be92dc1

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Sun Apr 11 21:19:58 2021 +0200

Changes in the function and GA function changes

commit f3f7cd09bbb5770943ece419cc4b310767ba8846

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Sun Apr 11 21:18:19 2021 +0200

Changes in the function and GA

commit bfea7031a80c184237afa21757fcd787a5c53434

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Sun Apr 11 21:16:46 2021 +0200

Changes in the definition of functions

commit 68c22df332a12c4a208dc5fab5066b3ed2d44d8f

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Sun Apr 11 18:39:17 2021 +0200

Print function changes

commit 559928a5a0de5284df3e26952fb55be7ad7e4a74

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Sun Apr 11 18:03:54 2021 +0200

Fitness value and Iteration plot for generation size 100 and varying mutation probability

commit 7ff28fe677ba978d41d40988832b24a653d9543a

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Sun Apr 11 18:01:27 2021 +0200

Fitness value and Iteration plot for generation size 80 and varying mutation probability

commit 630275f915a377013d81d4de2429247bd31f1d1c

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Sun Apr 11 17:59:26 2021 +0200

Fitness value and Iteration plot for generation size 40 and varying mutation probability

commit 850680d34d8183eb95383c8802f16f4a30405a81

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Sun Apr 11 17:26:53 2021 +0200

Genetic Algorithm for simple problem solving with varying population size and mutation probability

commit e7eed78ad1d7b9da0da92880bc2304623e2ff883

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Sun Apr 11 17:20:58 2021 +0200

Genetic Algorithm for simple problem solving results in a file

commit 64442d467ed800760d4be46590ccc989d2a8e85e

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Sun Apr 11 04:21:13 2021 +0200

Genetic Algorithm for simple problem solving with graph

commit bf57ca1075f3a0f65303b2562402f75adb0e36a7

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Sat Apr 10 20:48:12 2021 +0200

Genetic Algorithm for rigid body translation with Determinant of K matrix calculation

commit 8d413fa95fdc727e8d9bdd9a219ca784c754c98c

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Sat Apr 10 20:47:41 2021 +0200

Genetic Algorithm FEM verification with Determinant of K

commit 6867dc9725614ee582eb73f6d6206f864a54c020

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Sat Apr 10 20:38:43 2021 +0200

Genetic Algorithm for rigid body translation

commit 9ff2fb8a96afed49c7c3797328de2abc6f6768f4

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Sat Apr 10 20:18:21 2021 +0200

Genetic Algorithm FEM verification

commit eb49078e2cdd93fdd9c8b810e0aa84a29eff7596
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Mon Apr 5 16:58:16 2021 +0200

Genetic Algorithm with Rigid body translation

commit 151c54019ffa756c440b65111af6eb83b85e90d0
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Sun Apr 4 20:45:55 2021 +0200

benchmark problem with given values and weight checked

commit 01c70b1a0e6e6428b9b13160639be62e7ec8cfd0
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Sun Apr 4 00:23:44 2021 +0200

benchmark problem update in the coding

commit efa96252f8336a6fd432c6d7883912c4aa454559
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Fri Apr 2 23:38:26 2021 +0200

benchmark problem program basic run

commit cc6abcd288db620afc8856f27b7b78d916fa0fef
Author: Venkatasubramanian <nivass18.vn@gmail.com>
Date: Fri Apr 2 20:25:08 2021 +0200

Corrections in fitness constraints

commit b41bf14017d4fc070b5966695a46d235c3cc0b1d

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Fri Apr 2 19:27:16 2021 +0200

Corrections in algorithm sequence

commit 4d923e26bec39bffee5991532555de90434b1c6e

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Fri Apr 2 00:55:50 2021 +0200

Genetic Algorithm for simple problem

commit ec7dedd418966c0149a56851778b11f587cc6977

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Fri Apr 2 00:55:04 2021 +0200

Genetic Algorithm test_case_02

commit 4c4efe355cfffd41afd5412bd4c290557f77a1de

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Fri Apr 2 00:52:59 2021 +0200

Genetic Algorithm function

commit d753198ac47f60f8ceb3689a8262e5d92ee17da5

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Mar 23 11:15:03 2021 +0100

Genetic Algorithm FEM part verification with paper

commit f8830cf8155fb021c3ccabcc84320f89b2f27531

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Mar 23 09:17:01 2021 +0100

Genetic Algorithm FEM part verification for test case 02

commit 343d6e9238ffed45e8d858f84396b9a05d5c81b5

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Mon Mar 22 17:13:50 2021 +0100

Fitness Evaluation unittest

commit fecc3d0cc34f8bf51936c4ec5bedb66e37bb7aa1

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Mon Mar 22 01:33:35 2021 +0100

Genetic algorithm for simple problem

commit 51dfc40a5f601cdcd2f2fee1de0b260346f859a7

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Feb 23 18:50:33 2021 +0100

modification of program based on random population of the initial input array

commit 2db98b62ba8c1b2f5c5e8c887f7b22bcac85eeed

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Wed Feb 10 21:56:15 2021 +0100

Fitness function for GA is done and Sigularity of K matrix is done for regular, irregular elements

commit e02b97d19598ee5105d6e4422bac9358161c8e6d

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Feb 9 16:58:11 2021 +0100

Stress and strain plot is done and random initial input array is done and results are checked

commit 8dd992b62ee5627bbb46b5019e2a1a2e9d00a1c8

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Mon Jan 25 23:37:30 2021 +0100

Global displacement vector, global stress and straim vectors are updated and values are verified

commit fd84b993f5ee29712e68a77ea63d218634be30b1

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Fri Jan 22 20:49:38 2021 +0100

Global displacement vector is done and need to work on get the values for each element inorder to find stresses'

commit f93651bbe4a9ac958de1a2509596f056f3746f68

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Thu Jan 21 19:28:48 2021 +0100

K and M matrix values checked and reduced K and M matrix also done

commit 11290aa30445e5ec3527ea54c2e30dff581c3ace

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Sat Jan 16 19:40:26 2021 +0100

K and M matrix arrangement done and need to verify the answers'

commit ff41a8f03b109d7a27033e4ea16d88ba74c738b5

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Mon Jan 4 19:00:42 2021 +0100

K transformation matrix is done, need to work on erros

commit 4f5e002725d986bdc0402144a9135ce557480b0d

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Fri Jan 1 16:44:58 2021 +0100

picturization of the truss

commit a42d1fbaf16d11f8a4a83f3643e447d6dd82cb81

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Dec 29 17:19:35 2020 +0100

areas of the cross_section is retrived, plot to mention the node numbers

commit 3a4cef31f6f4c2db5751980aacd69504dcb53af0

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Mon Dec 28 14:08:21 2020 +0100

nodes and element numbers are updated

commit f1086a06820f7a8e57a6663c7af46ddbf93e1b37

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Mon Nov 30 14:38:07 2020 +0100

X and Y coordinates for the bars

commit 82d88fa125fcecabf908652eb2a815464acd507a

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Nov 10 20:36:54 2020 +0100

The final array of parameter is modified

commit ea826d5223aa12e7558ece21fc9cddaf00940fd2

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Nov 10 20:17:49 2020 +0100

modification of an array

commit c0a5ffdfcc6d05d71a1e2982c985325d390c35a8

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Nov 10 18:58:24 2020 +0100

data from the user and accessing

commit ff3ce01f39f2b1058b02e6140a40729f9a394224

Author: Venkatasubramanian <nivass18.vn@gmail.com>

Date: Tue Nov 10 17:23:23 2020 +0100

Initial commit