UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Superior d'Enginyeries Industrial, Aeroespacial i Audiovisual de Terrassa

Universitat Politècnica de Catalunya

MSc in Aerospace Engineering

—– Computational Engineering —–
Computational Structural Analysis

TASK 03: Structural topology optimization of 2D structures

Authors:
Pérez Ricardo, Carlos
Puigdelloses Vallcorba, Jaume

January 8^{th}, 2021

Contents

1 Introduction **3**

2 Problems solved **4**

2.1 Test example 4
2.2 Problem 1: L-shape structure 6
2.3 Problem 2: MBB beam 8
2.4 Problem 3: Gripper (compliant mechanism) 9
2.5 Problem 4: Structural sizing analysis of a cantilever beam 11
2.6 Conclusions 15

3 Bibliography **16**

List of Figures

2.1 Test domain - Cantilever beam 4
2.2 Test results from [1] 5
2.3 Optimal topology for test example 5
2.4 Cost evolution for test example 5
2.5 Deformed structured for the optimal topology for test example 6
2.6 Problem1 : L-shape structure setting 6
2.7 Optimal topology for L-shape structure 7
2.8 Cost evolution for L-shape structure 7
2.9 Deformed structured for the optimal topology for L-shape structure 7
2.10 Problem 2: MBB beam setting 8
2.11 Optimal topology for MBB Beam 9
2.12 Cost evolution for MBB Beam 9
2.13 Deformed structured for the optimal topology for MBB Beam 9
2.14 Problem 3: Gripper setting 10
2.15 Optimal topology for Gripper (compliant mechanism) . 10
2.16 Cost evolution for Gripper (compliant mechanism) . . . 10
2.17 Deformed structured for the optimal topology for Gripper (compliant mechanism) 11
2.18 Problem 4: Structural sizing analysis of a cantilever beam setting 11
2.19 Maximum displacement versus the number of elements in y-direction 13
2.20 Optimal topology for the optimal cantilever solution . . 13
2.21 Cost evolution for the optimal cantilever solution 13
2.22 Deformed structured for the optimal cantilever solution 14

1 Introduction

In this task, structural topology optimization of some 2D structures will be performed using VARTOP method, a topology optimization technique developed by our research group at the UPC [1]. Topology optimization (TO) is a mathematical method that optimizes material layout within a given design space, for a given set of loads, boundary conditions and constraints with the goal of maximizing the performance of the system. [2]

Topology Optimization has a wide range of applications in aerospace, mechanical, bio-chemical and civil engineering. Currently, engineers mostly use TO at the concept level of a design process. Due to the free forms that naturally occur, the result is often difficult to manufacture. For that reason the result emerging from TO is often fine-tuned for manufacturability.

The source code to find the optimal topology could be obtained from a GitHub repository. All lines of code were developed by the research group at the UPC, and several functions will be used by providing some inputs and storing and working with the provided outputs. No change or minimal changes will be done in the source code (considering functions and source code as a black box).

This assignment will be focused on the topological optimization of a set of 2D examples using the provided mathematical tool, with the aim of showing you the potential of these (topology optimization) techniques in the development and design of components in a wide range of applications. In particular, the results obtained from this tool will be used to perform a structural sizing analysis.

As an end user of this computational tool, you have to define the boundary conditions of the corresponding problem (i.e., the nodes in which the displacements are prescribed as well as the nodes in which the forces are applied), along with the 2D finite element mesh defined with the number of elements in both x and y-axes (unit square finite elements). In addition, other parameters related with the optimization must be defined: initial volume percentage of void material (Vol0), target volume percentage of void material (Vol), number of steps from Vol0 to Vol (nsteps), exponential factor of the volume evolution (k) and regularization parameter (tau). A general Matlab call of the code would be:

$$UNVARTOP\_2D\_compliance(nelx, nely, nsteps, Vol0, Vol, k, tau)$$

with k normally being equal to 0 and tau equal to 0.5.

2 Problems solved

Several problems will be solved in order to obtain the solution for different cases. In some cases the compliance function will be used for others the compliance & mechanism. Before explaining each problem, it is important to mention that some minor changes were carried out in the functions: UNVARTOP_2D_compliance and UNVARTOP_2D_complmechanism saved as UNVARTOP_2D_compliance_modified and UNVARTOP_2D_complmechanism_modified.

A common pattern was observed for the cases, the code used for solving each case was the same except for the definition of the forces F, fixed nodes, active nodes and passive nodes. So it was thought that it would make sense to provide as an input the definition of these, so the same function can be used (black box principle). To evaluate the expression, eval() function from MATLAB is used [3].

Moreover, a function to plot the deformed structure was implemented. As the values for the obtained displacements were much higher than the values of the initial structure, a factor (established by default as 0.1) was introduced to represent to deformed structure. And as in some cases the structure could contain up to 7701 nodes, only some lines of the structure are represented: the outer ones, the nodes in the middle and the ones located at 1/4 and 3/4 in x-direction and y-direction. However defining more lines could be advisable.

2.1 Test example

Consider a cantilever beam with a vertical pointwise load F applied on the bottom right corner of the design domain Ω. The displacements are prescribed on the left side of the domain.

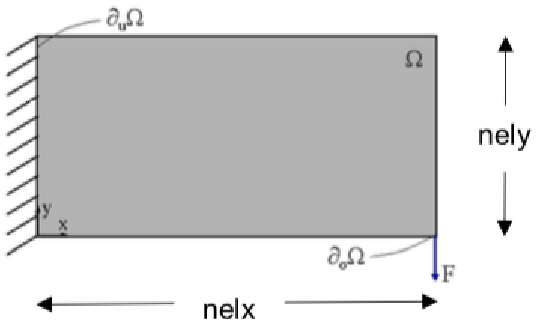

Figure 2.1: Test domain - Cantilever beam

Equivalently, the displacements for both degrees of freedom, the fixed degrees of freedom and the load are prescribed:

- F(2*find(coord(:,2)==0 & coord(:,1)==nelx)) = -0.01*nelx;
- fixed_dofs = 2*find(coord(:,1)==0)+(-1:0);

- active_node = [ ];
- passive_node = [ ];

As a test, this example was run with nelx = 100, nely = 50, nsteps = 50, Vol0 = 0, Vol = 0.5, k=0 and tau = 0.5. The obtained results obtained in [1] were the following:

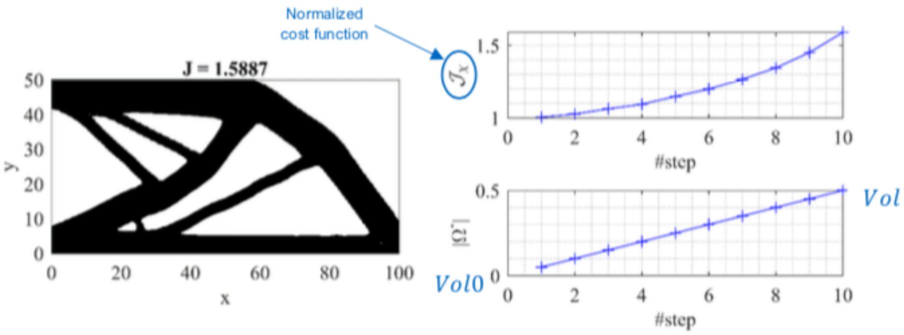

Figure 2.2: Test results from [1]

The left figure illustrates the optimal topology for the target volume (Vol = 0.5), while the right one displays the evolution of the cost function (expression to be minimized) with the steps (i.e. with an increasing void percentage).

Using the function modified version of UNVARTOP_2D_compliance, the following results were obtained:

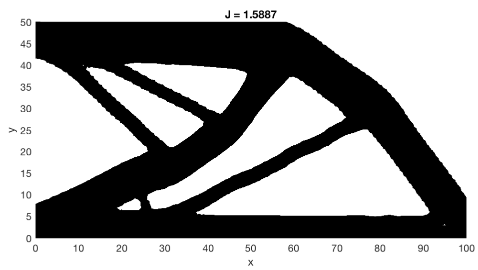

Figure 2.3: Optimal topology for test example

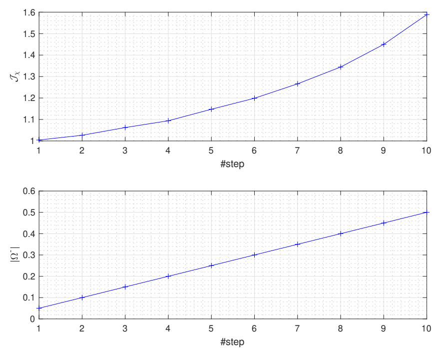

Figure 2.4: Cost evolution for test example

The deformed structure with a factor 0.1 can be shown 2.5. As it can be observed the value and figure for the optimal topology is the same as the one obtained in [1] as expected.

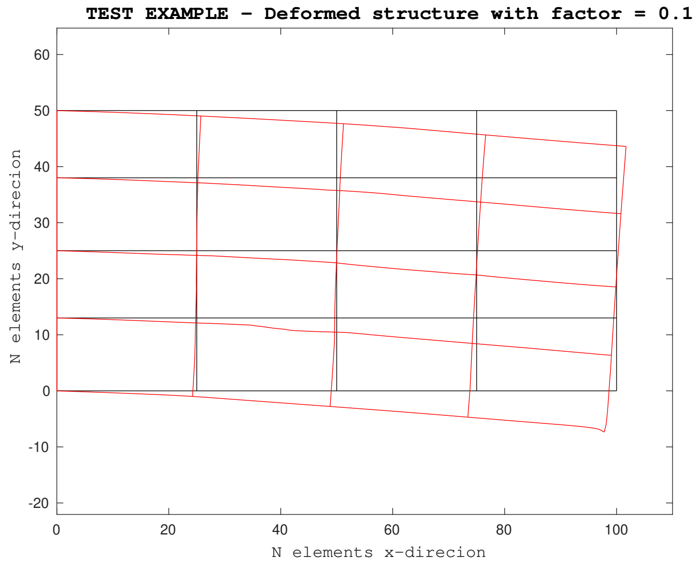

Figure 2.5: Deformed structured for the optimal topology for test example

2.2 Problem 1: L-shape structure

Consider the design of a simplified hook of aspect ratio 1:1 with a single load applied on the right side of the domain at y = 0.2. The nodes on the top-left boundary (y= 1 and x < 0.4) are fixed (the displacement of the corresponding degrees of freedom is prescribed to 0). In addition, the domain has a prescribed void zone in the top right area, defined by xi $\geq$ 0.4 and yi $\geq$ 0.4. All the nodes included in this area are listed in passive_node and the characteristic function of these elements is therefore set to β.

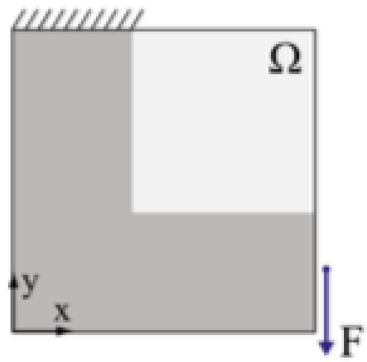

Figure 2.6: Problem1 : L-shape structure setting

According to the definition of the previous problem, perform an optimization with 140 elements in each direction in 10 steps, from Vol0 = 0.36 to Vol = 0.75. Set the exponential parameter k and the regularization parameter tau to 0 and 0.5, respectively.

The the fixed degrees of freedom, loads and active and passive nodes are prescribed:

F(2*find(coord(:,2)==round(0.2*nely) & coord(:,1)==nelx),1)=-0.01*nelx;

fixed_dofs = reshape(2*find(coord(:,1) <= 0.4*nelx&coord(:,2)==nely)+(-1:0),1,[]);

active_node = [];

passive_node = find(coord(:,1)>ceil(nelx*0.4) & coord(:,2)>ceil(nely*0.4));

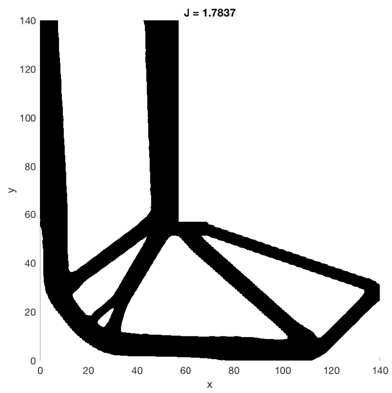

Figure 2.7: Optimal topology for L-shape structure

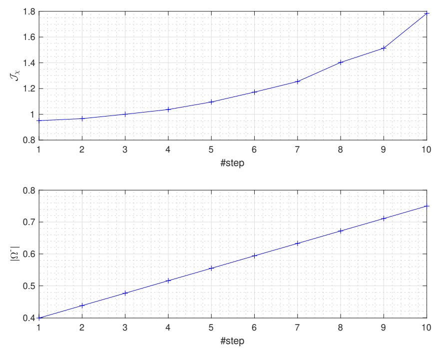

Figure 2.8: Cost evolution for L-shape structure

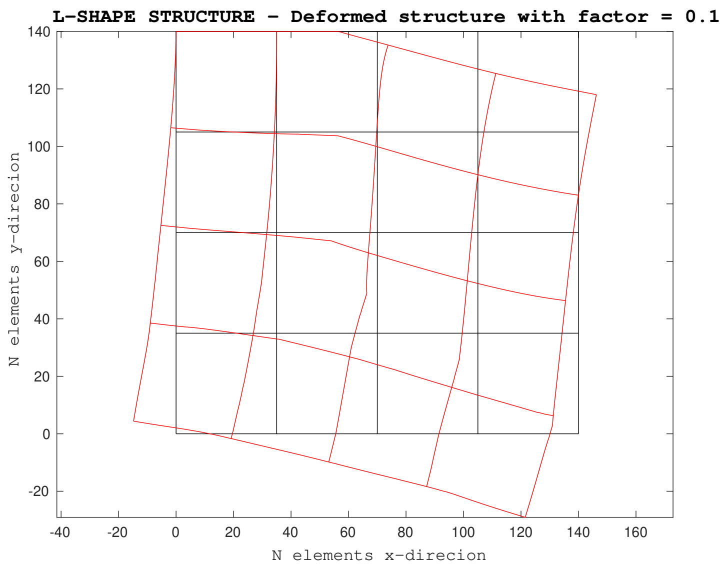

Figure 2.9: Deformed structured for the optimal topology for L-shape structure

The deformed structure with a factor 0.1 can be shown 2.9. The top-left nodes that are fixed in both x and y directions suffer no displacement in the deformed structure as one can expect and the the solution makes sense with the disposition of the force and fixed nodes.

2.3 Problem 2: MBB beam

Optimize half of the MBB beam with an aspect ratio of 2.5 to 1 (x-axis and y-axis, respectively), and a single vertical force F applied at the top-left corner. Symmetry must be assumed on the left side of the domain (horizontal displacement is prescribed to 0) and the vertical displacement at the bottom-right corner is also constrained.

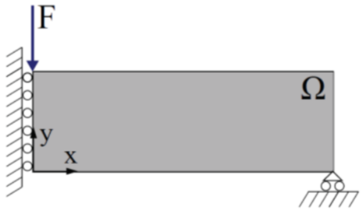

Figure 2.10: Problem 2: MBB beam setting

Discretize the domain with 190 and 76 elements in the x-axis and y-axis, respectively. Perform an optimization from Vol0= 0 to Vol = 0.5 in 10 equally spaced steps (k = 0) and tau = 1.

The the fixed degrees of freedom, loads and active and passive nodes are prescribed:

- F(2*find(coord(:,2)==nely & coord(:,1)==0),1) = -0.01*nelx;
- fixed_dofs = [reshape(2*find(coord(:,1)==0)- 1,1,[]), reshape(2*find(coord(:,1)==nelx & coord(:,2)==0),1,[])];
- active_node = [];
- passive_node = [];

The deformed structure with a factor 0.1 can be shown 2.13. It can be observed that the left part of the structure heads downwards but maintaining the restriction that it can only move in y-direction. Similarly with the restriction in the bottom-right node, so both node restrictions are fulfilled as one would expect.

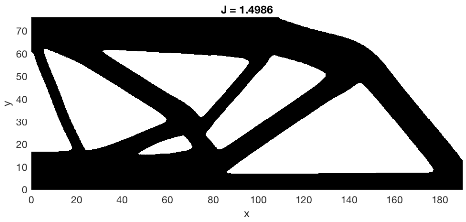

Figure 2.11: Optimal topology for MBB Beam

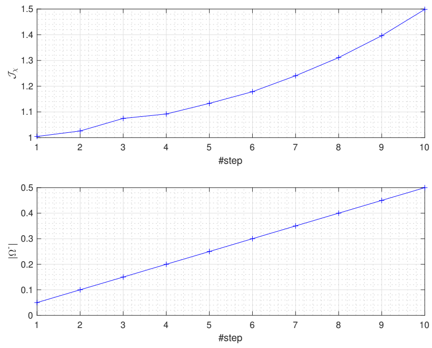

Figure 2.12: Cost evolution for MBB Beam

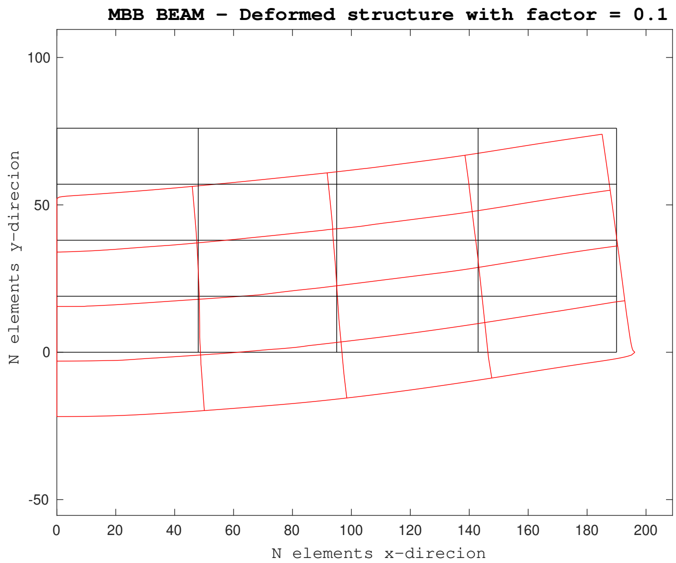

Figure 2.13: Deformed structured for the optimal topology for MBB Beam

2.4 Problem 3: Gripper (compliant mechanism)

Use the numerical tool to optimize the design of a compliant mechanism with an aspect ratio 2.5:1, in which the vertical displacement at the output port (jaws of the gripper) is maximized when a horizontal force is applied at the input port (left side of the domain). The domain is supported by a small area in the bottom-left corner ($y < 0.1$ and $x = 0$) and symmetry is applied on the top side of Ω ($y = 1$). A small area in the output port is set to soft material (i.e. included in passive_node list) in order to represent the gap in the jaws of the gripper. In addition, some stiff material areas are restricted in both ports (active_node list), and an additional spring stiffness must be included in both ports to ensure convergence, the values being equal to 0.05 and 0.005 for the input port and output port, respectively. It can be directly added to the stiffness matrix in the corresponding degrees of freedom.

Note that UNVARTOP_2D_complmechanism(nelx,nely,nsteps,Vol0,Vol,k,tau) must now be used for this example. Using a 200x80 finite element mesh, perform the optimization of this example in 12 steps from Vol0 = 0 to Vol = 0.7, using an exponential evolution with k = 2 and a regularization parameter of tau = 0.5.

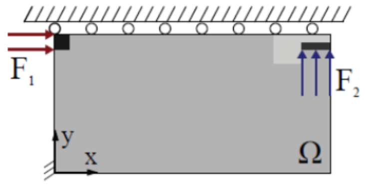

Figure 2.14: Problem 3: Gripper setting

Discretize the domain with 190 and 76 elements in the x-axis and y-axis, respectively. Perform an optimization from Vol0= 0 to Vol = 0.5 in 10 equally spaced steps (k = 0) and tau = 1.

The the fixed degrees of freedom, loads and active and passive nodes are prescribed:

- F(2*find(coord(:,2)>=0.9*nely & coord(:,1)==0) - 1,1) = 0.0001*nelx;
- F(2*find(coord(:,2)==round(0.9*nely) & coord(:,1)>= 0.9*nelx),2)=0.0001*nelx;
- fixed_dofs = [reshape(2*find(coord(:,2)==nely),1,[]), reshape(2*find(coord(:,1)==0 & coord(:,2)<=0.1*nely)+(-1:0),1,[])];
- active_node = [find(coord(:,2)>0.9*nely & coord(:,1)<0.05*nelx); find(coord(:,2)> 0.9*nely & coord(:,2)<=0.95*nely & coord(:,1) >= 0.9*nelx)];
- passive_node = [find(coord(:,1)>0.8*nelx & coord(:,1)<0.9*nelx & coord(:,2)>0.8*nely); find(coord(:,1)>=0.9*nelx & coord(:,2)>0.95*nely)];

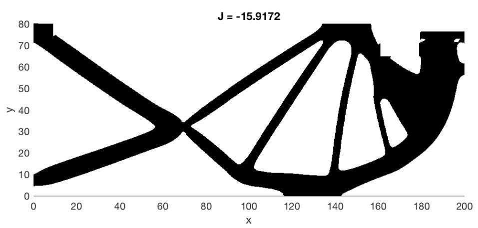

Figure 2.15: Optimal topology for Gripper (compliant mechanism)

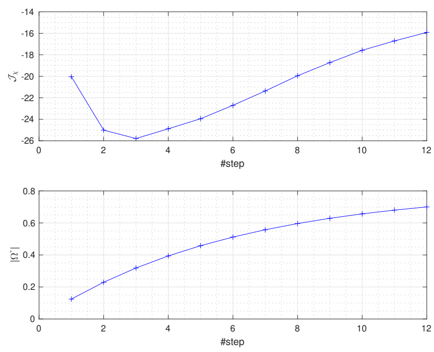

Figure 2.16: Cost evolution for Gripper (compliant mechanism)

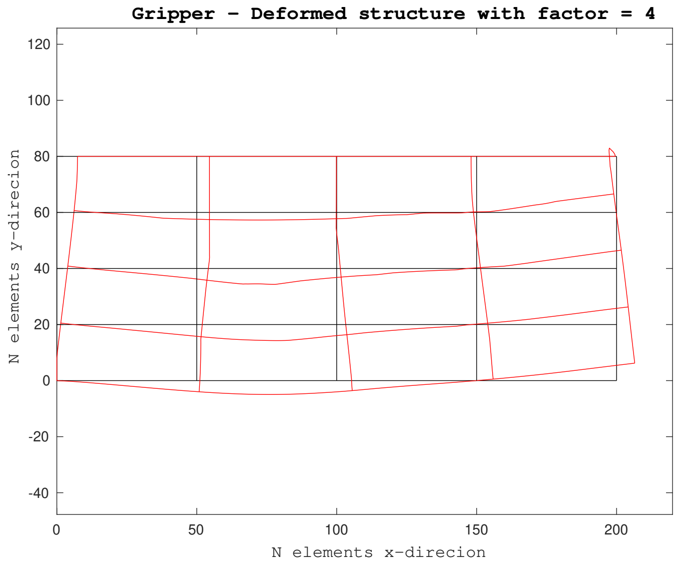

Figure 2.17: Deformed structured for the optimal topology for Gripper (compliant mechanism)

The deformed structure with a factor 4 can be shown 2.17. In this case, two load cases are calculated and the displacements U obtained had 2 values for each component for each node, one referring to the first load case and the other to the second one. So in order to obtain, the total displacements, both loads were added. Moreover, the order of the displacements were much smaller than for the previous cases so in this case it has been used a factor 4 (it is to said X_{def} = X + factor ·U ; where X corresponds to the initial coordinates of the structure and U the values of displacements obtained).

2.5 Problem 4: Structural sizing analysis of a cantilever beam

The topology optimization technique can be used to determine which is the minimum domain size (minimum vertical size when the horizontal size is kept constant) subject to a maximum acceptable vertical displacement for the cantilever beam. A vertical point-wise load is applied on the middle of the right side of the domain while the displacements in the left side are prescribed to 0, as illustrated below:

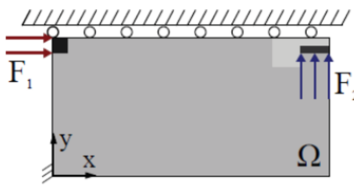

Figure 2.18: Problem 4: Structural sizing analysis of a cantilever beam setting

Then, perform the optimization of the previous problem for a set of nely values (ranging from 20 to 160), keeping constant the value of the load F and the number of elements in the x- direction (e.g. nelx = 100). Note that the actual size in this implementation is measured with the number of elements in each direction, since the finite elements are squares of unit size. Therefore, an increase in the number of elements in a given direction implies an increase of this length and a variation in the aspect ratio.

In addition, the amount of stiff material (in number of elements) must be also kept constant. Therefore, keep in mind that the target void fraction Vol should change with the number of elements nely (or the vertical size) to keep the total number of stiff elements constant. The corresponding target volume percentage Vol is given by:

$$Vol(nely) = 1 - \frac{(1 - \bar{Vol} \cdot \bar{nely})}{nely} \tag{2.1}$$

where $\bar{Vol}$ and $\bar{nely}$ are reference values. Consider the example with nelx = 100, nely = 50, Vol0 = 0 and Vol = 0.7 as the reference case to determine the exact Vol fraction.

As for the number of steps nsteps, it can be set to 14, although it can be increased to 20 in case the optimization does not converge. The other two parameters can be set to 0 and 0.5 for the evolution factor k and the regularization factor tau, respectively.

From each topology optimization problem, you must obtain the maximum displacement value as well as the optimal topology layout so that you can plot the maximum displacement versus the number of elements in the vertical direction with the corresponding topologies. Determine then the optimal topology with the minimum vertical size that does not exceed a maximum displacement of 80 units.

As observed in Figure 2.19, the optimal minimum vertical size so it does not exceed a maximum displacement of 80 units is around 65 elements (the maximum displacement obtained in this case is 80.3156 units so the optimal solution would be 66 elements).

The optimal solution for nely = 66 is presented as follows, where the maximum displacement achieved in a node in the structure is 79.3978 units (that observing the Figure 2.22 one could say that is probably the bottom-right node. The value of J obtained is 3.8514.

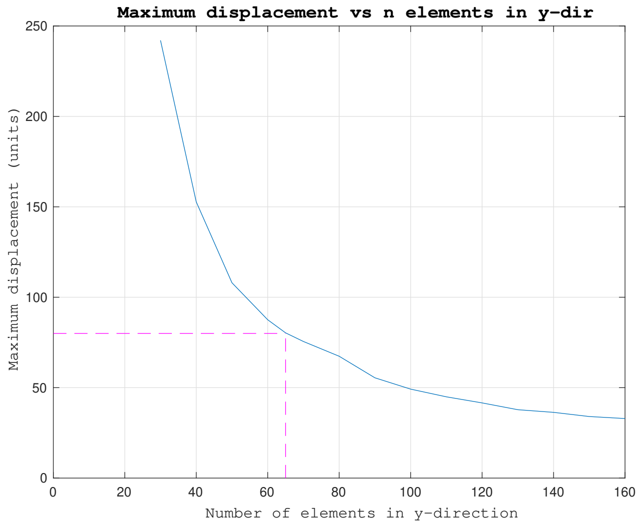

Figure 2.19: Maximum displacement versus the number of elements in y-direction

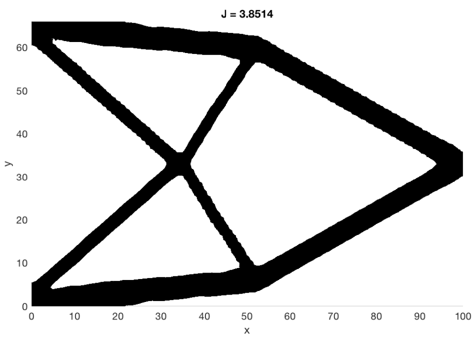

Figure 2.20: Optimal topology for the optimal cantilever solution

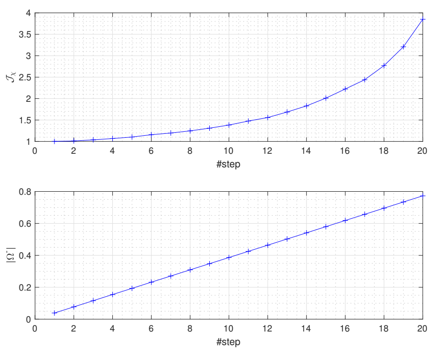

Figure 2.21: Cost evolution for the optimal cantilever solution

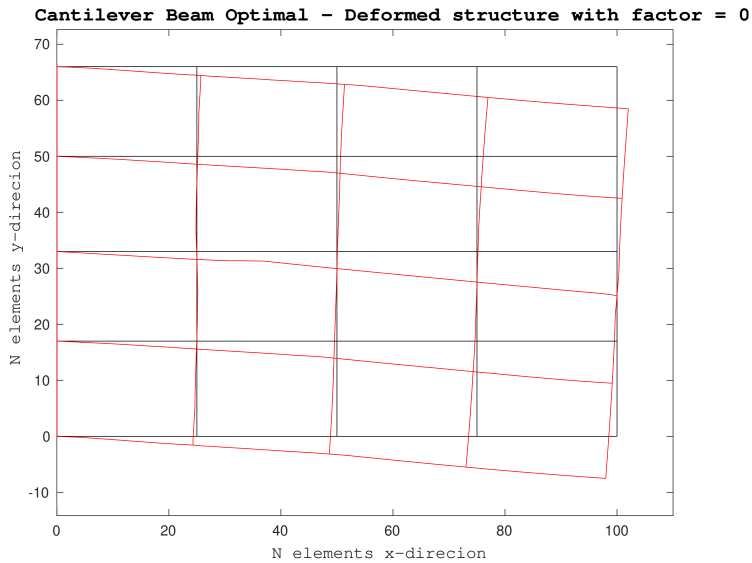

Figure 2.22: Deformed structured for the optimal cantilever solution

2.6 Conclusions

After developing this task, several conclusions were achieved. First of all, it is important to highlight that the source code imported from Git Hub and the article corresponding to the development of the algorithm was complex and the work behind it is recognisable. Reading through the article, several concepts familiar to engineers were introduced (Jacobian, topology, Lagrangian function, loss function...), however they were not observed in the area of Computational Structure Analysis, in subjects studied before. The power they offer enable to obtain for instance the optimal topology for a structure.

Optimal topology is advisable in order to reduce weight, minimise deformations by reinforcing most affected areas wisely; especially in aerospace where weight is so important. This academic article can offer many structural and composites companies a tool to enhance their designs, by reducing the weight, enhancement the robustness and by allocating the material wisely. Applications in which optimal topology can be useful are infinite.

Throughout the development of this task, several problems were solved. The results obtained are successful, as restrictions in the problem were fulfilled and the displacements refactorised achieved were as we would expect. Projects like this offer a numeric approach to determine which areas of a structure will need a reinforcement to make it more robust.

3 Bibliography

References

[1] Daniel Yago, Juan Cante, Oriol Lloberas-Valls, and Javier Oliver. "Topology Optimization using the UNsmooth VARiational Topology OPtimization (UNVARTOP) method: an educational implementation in Matlab EDUCATIONAL ARTICLE". In: *Structural and Multidisciplinary Optimization* (Nov. 2020). DOI: 10.1007/s00158-020-02722-0.

[2] *Topology optimization - Wikipedia*. URL: https://en.wikipedia.org/wiki/Topology{_}optimization (visited on 12/30/2020).

[3] *Evaluate MATLAB expression - MATLAB eval - MathWorks España*. URL: https://es.mathworks.com/help/matlab/ref/eval.html (visited on 12/30/2020).