Práctica 1 Descripción y uso de código de optimización topológica

Uriel G. Ernesto A. Marcela O. Ana L. Diego O.

5 de septiembre de 2022

Resumen

En esta práctica explicaremos lo que es un codigo topológico y que sucede cuando lo optimizamos, así mismo se mostrará un ejemplo de cómo se vería un código de optimización topológica

1. Introducción

La optimización topológica es una herramienta matemática que le perite a un diseñador sintetizar topologías de forma óptima. En la mecánica se entiende como topología optima a una pieza o parte mecánica diseñada especialmente para maximizar o minimizar alguna característica deseada[1].

2. Objetivo

El estudiante conocerá cada una de las secciones que integran el código de optimización topológica, como se debe crear el archivo (.m) en MATLAB y como se ejecuta el análisis.

3. Instrucciones

El estudiante deberá presentar una propuesta de análisis de formas y de la programación, de características de trabajo específicas (programación) que presenta.

4. Estado del arte

La optimización topológica es una técnica englobada dentro del campo de análisis estructural. Se basa en el análisis mecánico de un componente o estructura. Su principal objetivo es el aligeramiento estructural manteniendo las funcionalidades mecánicas del componente objetivo. A diferencia de otros tipos de optimización, la optimización topológica ofrece un nuevo concepto de diseño estructural enfocado a aquellas aplicaciones donde el peso del componente es crucial[2].

Gracias a los nuevos métodos computacionales, es posible llevar la optimización a un nivel más complejo de análisis a nivel estático, dinámico, plástico, modal o de impacto, entre otros, los cuales pueden considerarse durante el proceso de optimización.

La optimización topológica comienza con la creación de un modelo 3D en la fase de borrador, en el que se aplicaran las diferentes cargas o fuerzas para la pieza (una presión sobre las lengüetas de sujeción, por ejemplo). Después, el software se encarga de calcular todas las tensiones aplicadas[3].

En este nivel, se puede realizar un corte de la pieza con el fin de retirar las partes no sometidas a las fuerzas. La geometría final, que cumple con los requisitos mecánicos y de diseño, se puede obtener finalmente después de

alisar la pieza. De esta forma, la optimización topológica responde a la necesidad de reducción de masa además del aumento de la resistencia mecánica de la pieza.

5. Codigo de programación

simulate.m | top99neo.m *

```
%%% A 99 LINE TOPOLOGY OPTIMIZATION CODE BY OLESIGMUND, OCTOBER 1999 %%%
function P1(nelx,nely,volfrac,penal,rmin)
% INITIALIZE
x(1:nely,1:nelx) = volfrac;
loop = 0;
change = 1.;
% START ITERATION
while change > 0.01
    loop = loop + 1;
    xold = x;
% FE-ANALYSIS
    [U]=FE(nelx,nely,x,penal);
% OBJECTIVE FUNCTION AND SENSITIVITY ANALYSIS
     [KE] = lk;
     c = 0.;
     for ely = 1:nely
        for elx = 1:nelx
             n1 = (nely+1)*(elx-1)+ely;
             n2 = (nely+1)* elx +ely;
             Ue = U([2*n1-1;2*n1; 2*n2-1;2*n2; 2*n2+1; 2*n2+2; 2*n1+1;2*n1+2],1);
             c = c + x(ely,elx)^penal*Ue'*KE*Ue;
             dc(ely,elx) = -penal*x(ely,elx)^(penal-1)*Ue'*KE*Ue;
        end
    end
    % FILTERING OF SENSITIVITIES
    [dc] = check(nelx,nely,rmin,x,dc);
    % DESIGN UPDATE BY THE OPTIMALITY CRITERIA METHOD
    [x] = OC(nelx,nely,x,volfrac,dc);
    % PRINT RESULTS
    change = max(max(abs(x-xold)));
    disp(['It.:' sprintf('%4i',loop) 'Obj.:' sprintf('%10.4f',c) ...
    ' Vol.: ' sprintf('%6.3f',sum(sum(x))/(nelx*nely)) ...
    ' ch.: ' sprintf('%6.3f',change )])
    % PLOT DENSITIES
    colormap(gray); imagesc(-x); axis equal; axis tight; axis off;pause(1e-6);
end
end

%%%%%%%%% OPTIMALITY CRITERIA UPDATE %%%%%%%%
function [xnew]=OC(nelx,nely,x,volfrac,dc)
l1 = 0; l2 = 100000; move = 0.2;
while (l2-l1 > 1e-4)
        lmid = 0.5*(l2+l1);
        xnew = max(0.001,max(x-move,min(1.,min(x+move,x.*sqrt(-dc./lmid)))));
        if sum(sum(xnew)) - volfrac*nelx*nely > 0
            l1 = lmid;
        else
            l2 = lmid;
        end
end
end

```

Figura 1: Parte 1 del codigo

simulate.m | top99neo.m

```
                l2 = lmid;
            end
    end
    end

    %%%%%%%%% MESH-INDEPENDENCY FILTER %%%%%%%%%%
    function [dcn]=check(nelx,nely,rmin,x,dc)
    dcn=zeros(nely,nelx);
    for i = 1:nelx
        for j = 1:nely
            sum=0.0;
            for k = max(i-round(rmin),1):min(i+round(rmin),nelx)
                for l = max(j-round(rmin),1):min(j+round(rmin), nely)
                    fac = rmin-sqrt((i-k)^2+(j-l)^2);
                    sum = sum+max(0,fac);
                    dcn(j,i) = dcn(j,i) + max(0,fac)*x(l,k)*dc(l,k);
                end
            end
            dcn(j,i) = dcn(j,i)/(x(j,i)*sum);
        end
    end
    end

    %%%%%%%%% FE-ANALYSIS %%%%%%%%%%
    function [U]=FE(nelx,nely,x,penal)
    [KE] = lk;
    K = sparse(2*(nelx+1)*(nely+1), 2*(nelx+1)*(nely+1));
    F = sparse(2*(nely+1)*(nelx+1),1); U =sparse(2*(nely+1)*(nelx+1),1);
    for ely = 1:nely
        for elx = 1:nelx
            n1 = (nely+1)*(elx-1)+ely;
            n2 = (nely+1)* elx +ely;
            edof = [2*n1-1; 2*n1; 2*n2-1; 2*n2; 2*n2+1;2*n2+2;2*n1+1; 2*n1+2];
            K(edof,edof) = K(edof,edof) + x(ely,elx)^penal*KE;
        end
    end
    % DEFINE LOADSAND SUPPORTS(HALF MBB-BEAM)
    F(2,1) = -1;
    fixeddofs =union([1:2:2*(nely+1)],[2*(nelx+1)*(nely+1)]);
    alldofs = [1:2*(nely+1)*(nelx+1)];
    freedofs = setdiff(alldofs,fixeddofs);
    % SOLVING 127
    U(freedofs,:) = K(freedofs,freedofs) \F(freedofs,:);
    U(fixeddofs,:)= 0;
    end
    %%%%%%%%% ELEMENT STIFFNESS MATRIX %%%%%%%
    function [KE]=lk
    E = 1.;
    nu = 0.3;
    k=[ 1/2-nu/6 1/8+nu/8 -1/4-nu/12 -1/8+3*nu/8 ...
    -1/4+nu/12 -1/8-nu/8 nu/6 1/8-3*nu/8];
    KE = E/(1-nu^2)*[ k(1) k(2) k(3) k(4) k(5) k(6) k(7) k(8)
    k(2) k(1) k(8) k(7) k(6) k(5) k(4) k(3)
    k(3) k(8) k(1) k(6) k(7) k(4) k(5) k(2)
    k(4) k(7) k(6) k(1) k(8) k(3) k(2) k(5)
    k(5) k(6) k(7) k(8) k(1) k(2) k(3) k(4)
    k(6) k(5) k(4) k(3) k(2) k(1) k(8) k(7)
    k(7) k(4) k(5) k(2) k(3) k(8) k(1) k(6)
    k(8) k(3) k(2) k(5) k(4) k(7) k(6) k(1)];
    end
```

Figura 2: Parte 2 del codigo

6. Procedimiento de la programación

Al empezar el programa encontramos que necesitamos colocar ciertas variables como son: •nelx, que indica el número de elementos en las direcciones horizontales.

•nely, que indica ese mismo valor, pero para direcciones verticales.

•volfrac, la cual sirve para identificar la fracción de volumen.

•penal, que hace referencia al poder de penalización.

•rmin, esta variable dicta el tamaño del filtro el cual es dividido por el tamaño del elemento.

```
%%% A 99 LINE TOPOLOGY OPTIMIZATION CODE BY OLESIGMUND, OCTOBER 1999 %%%
function P1(nelx,nely,volfrac,penal,rmin)
% INITIALIZE
x(1:nely,1:nelx) = volfrac;
loop = 0;
change = 1.;
% START ITERATION
while change > 0.01
    loop = loop + 1;
    xold = x;
% FE-ANALYSIS
    [U]=FE(nelx,nely,x,penal);
% OBJECTIVE FUNCTION AND SENSITIVITY ANALYSIS
     [KE] = lk;
     c = 0.;
     for ely = 1:nely
        for elx = 1:nelx
             n1 = (nely+1)*(elx-1)+ely;
             n2 = (nely+1)* elx +ely;
             Ue = U([2*n1-1;2*n1; 2*n2-1;2*n2; 2*n2+1; 2*n2+2; 2*n1+1;2*n1+2],1);
             c = c + x(ely,elx)^penal*Ue'*KE*Ue;
             dc(ely,elx) = -penal*x(ely,elx)^(penal-1)*Ue'*KE*Ue;
        end
    end
    % FILTERING OF SENSITIVITIES
    [dc] = check(nelx,nely,rmin,x,dc);
    % DESIGN UPDATE BY THE OPTIMALITY CRITERIA METHOD
    [x] = OC(nelx,nely,x,volfrac,dc);
    % PRINT RESULTS
    change = max(max(abs(x-xold)));
    disp(['It.:' sprintf('%4i',loop) 'Obj.:' sprintf('%10.4f',c) ...
```

Figura 3: Inicio de codigo

En esta parte se inicia por igual el análisis del material uniformemente en el dominio del diseño. Después, se inicia con la subrutina del análisis de elemento finito. Esta subrutina sirve para regresar o alojar sus resultados en un arreglo o vector de desplazamiento U.

La siguiente parte corresponde al análisis de sensibilidad y la subrutina de rigidez. En esta seccción, se realiza, usando la función “for”, un bucle de todos los elementos. Dentro de este bucle, se consigue extraer un segundo vector de desplazamiento el cual viene siendo el vector “Ue”.

La segunda sección del código es el optimizador basado en criterios de optimalidad. Esta subrutina actualiza las variables de diseño y utiliza la sección “sum(sum(xnew))” la cual es una funcion monótonamente decreciente del multiplicador de Lagrange (lag) que indica el volumen del material.

```
32          ' Vol.: ' sprintf('%6.3f',sum(sum(x))/(nelx*nely)) ...
33          ' ch.: ' sprintf('%6.3f',change )])
34          % PLOT DENSITIES
35          colormap(gray); imagesc(-x); axis equal; axis tight; axis off;pause(1e-6);
36      end
37      end
38
39      %%%%%%%%% OPTIMALITY CRITERIA UPDATE %%%%%%%%
40      function [xnew]=OC(nelx,nely,x,volfrac,dc)
41      l1 = 0; l2 = 100000; move = 0.2;
42      while (l2-l1 > 1e-4)
43              lmid = 0.5*(l2+l1);
44              xnew = max(0.001,max(x-move,min(1.,min(x+move,x.*sqrt(-dc./lmid)))));
45              if sum(sum(xnew)) - volfrac*nelx*nely > 0
46                  l1 = lmid;
47              else
48                  l2 = lmid;
49              end
50      end
51      end
```

Figura 4: Sección 2 del código

La tercera seccion, consiste en un filtrado de malla, aqui se controla las variaciones en las variables en caso de que ocurra algún percance.

```
%%%%%%%%%% MESH-INDEPENDENCY FILTER %%%%%%%%%%
function [dcn]=check(nelx,nely,rmin,x,dc)
dcn=zeros(nely,nelx);
for i = 1:nelx
    for j = 1:nely
        sum=0.0;
        for k = max(i-round(rmin),1):min(i+round(rmin),nelx)
            for l = max(j-round(rmin),1):min(j+round(rmin), nely)
                fac = rmin-sqrt((i-k)^2+(j-l)^2);
                sum = sum+max(0,fac);
                dcn(j,i) = dcn(j,i) + max(0,fac)*x(l,k)*dc(l,k);
            end
        end
        dcn(j,i) = dcn(j,i)/(x(j,i)*sum);
    end
end
end
```

Figura 5: Sección 3 del código

la última parte del código, corresponde al código del elemento finito y a la matriz de rigidez global, la cual está formada por un bucle sobre todos los elementos, algo similar a lo que se vio en una sección previa del código. Acá se vuelven a utilizar las variables n1 y n2 que indican los números de nodos de elementos superior izquierdo y derecho en números de nodos globales. Estos datos se obtienen y se usan para insertar la matriz de rigidez de elementos adecuadamente en la matrizde rigidez global. Finalmente, se realiza un calculo para determinar la matriz de rigidez del elemento. Para esto se utiliza el módulo de Young (E) y la relación de Poisson (nu).

```
%%%%%%%%%% FE-ANALYSIS %%%%%%%%%%%
function [U]=FE(nelx,nely,x,penal)
[KE] = lk;
K = sparse(2*(nelx+1)*(nely+1), 2*(nelx+1)*(nely+1));
F = sparse(2*(nely+1)*(nelx+1),1); U =sparse(2*(nely+1)*(nelx+1),1);
for ely = 1:nely
    for elx = 1:nelx
        n1 = (nely+1)*(elx-1)+ely;
        n2 = (nely+1)* elx +ely;
        edof = [2*n1-1; 2*n1; 2*n2-1; 2*n2; 2*n2+1;2*n2+2;2*n1+1; 2*n1+2];
        K(edof,edof) = K(edof,edof) + x(ely,elx)^penal*KE;
    end
end
% DEFINE LOADSAND SUPPORTS(HALF MBB-BEAM)
F(2,1) = -1;
fixeddofs =union([1:2:2*(nely+1)],[2*(nelx+1)*(nely+1)]);
alldofs = [1:2*(nely+1)*(nelx+1)];
freedofs = setdiff(alldofs,fixeddofs);
% SOLVING 127
U(freedofs,:) = K(freedofs,freedofs) \F(freedofs,:);
U(fixeddofs,:)= 0;
end
%%%%%%%%%% ELEMENT STIFFNESS MATRIX %%%%%%%
function [KE]=lk
E = 1.;
nu = 0.3;
k=[ 1/2-nu/6 1/8+nu/8 -1/4-nu/12 -1/8+3*nu/8 ...
-1/4+nu/12 -1/8-nu/8 nu/6 1/8-3*nu/8];
KE = E/(1-nu^2)*[ k(1) k(2) k(3) k(4) k(5) k(6) k(7) k(8)
k(2) k(1) k(8) k(7) k(6) k(5) k(4) k(3)
```

Figura 6: Parte final del código

```
    k(3) k(8) k(1) k(6) k(7) k(4) k(5) k(2)
    k(4) k(7) k(6) k(1) k(8) k(3) k(2) k(5)
    k(5) k(6) k(7) k(8) k(1) k(2) k(3) k(4)
    k(6) k(5) k(4) k(3) k(2) k(1) k(8) k(7)
    k(7) k(4) k(5) k(2) k(3) k(8) k(1) k(6)
    k(8) k(3) k(2) k(5) k(4) k(7) k(6) k(1)];
end
```

Figura 7: Parte fina del código

7. Implementacion o desarrollo de la programación en sus diferentes vistas

Para la primera prueba se utilizarán los valores P1(30,10,0.5,3,1.5), los resultados fueron los siguientes.

```
K>> P1(30,10,0.5,3.0,1.5)
It.:   1Obj.:  984.5548 Vol.:  0.500 ch.:  0.200
It.:   2Obj.:  580.4116 Vol.:  0.500 ch.:  0.200
It.:   3Obj.:  419.0640 Vol.:  0.500 ch.:  0.200
It.:   4Obj.:  357.2235 Vol.:  0.500 ch.:  0.194
It.:   5Obj.:  337.8750 Vol.:  0.500 ch.:  0.125
It.:   6Obj.:  327.5757 Vol.:  0.500 ch.:  0.141
It.:   7Obj.:  319.9654 Vol.:  0.500 ch.:  0.105
It.:   8Obj.:  312.9614 Vol.:  0.500 ch.:  0.113
It.:   9Obj.:  307.3377 Vol.:  0.500 ch.:  0.090
It.:  10Obj.:  302.7199 Vol.:  0.500 ch.:  0.095
It.:  11Obj.:  298.9022 Vol.:  0.500 ch.:  0.075
It.:  12Obj.:  295.5064 Vol.:  0.500 ch.:  0.080
It.:  13Obj.:  292.2006 Vol.:  0.500 ch.:  0.068
It.:  14Obj.:  288.9708 Vol.:  0.500 ch.:  0.066
It.:  15Obj.:  285.5598 Vol.:  0.500 ch.:  0.062
It.:  16Obj.:  281.9851 Vol.:  0.500 ch.:  0.063
It.:  17Obj.:  277.8991 Vol.:  0.500 ch.:  0.059
It.:  18Obj.:  273.2379 Vol.:  0.500 ch.:  0.066
It.:  19Obj.:  267.7815 Vol.:  0.500 ch.:  0.062
It.:  20Obj.:  261.5262 Vol.:  0.500 ch.:  0.073
It.:  21Obj.:  254.5783 Vol.:  0.500 ch.:  0.064
It.:  22Obj.:  247.6060 Vol.:  0.500 ch.:  0.061
It.:  23Obj.:  241.9718 Vol.:  0.500 ch.:  0.048
It.:  24Obj.:  238.3026 Vol.:  0.500 ch.:  0.031
It.:  25Obj.:  236.0399 Vol.:  0.500 ch.:  0.025
It.:  26Obj.:  234.4791 Vol.:  0.500 ch.:  0.023
It.:  27Obj.:  233.3757 Vol.:  0.500 ch.:  0.020
```

(a) Primeros resultados

```
It.:  28Obj.:  232.5235 Vol.:  0.500 ch.:  0.022
It.:  29Obj.:  231.8796 Vol.:  0.500 ch.:  0.019
It.:  30Obj.:  231.3397 Vol.:  0.500 ch.:  0.021
It.:  31Obj.:  230.9233 Vol.:  0.500 ch.:  0.019
It.:  32Obj.:  230.5497 Vol.:  0.500 ch.:  0.019
It.:  33Obj.:  230.2440 Vol.:  0.500 ch.:  0.018
It.:  34Obj.:  229.9589 Vol.:  0.500 ch.:  0.018
It.:  35Obj.:  229.7230 Vol.:  0.500 ch.:  0.018
It.:  36Obj.:  229.4823 Vol.:  0.500 ch.:  0.017
It.:  37Obj.:  229.2861 Vol.:  0.500 ch.:  0.017
It.:  38Obj.:  229.0731 Vol.:  0.500 ch.:  0.016
It.:  39Obj.:  228.8956 Vol.:  0.500 ch.:  0.016
It.:  40Obj.:  228.7072 Vol.:  0.500 ch.:  0.015
It.:  41Obj.:  228.5415 Vol.:  0.500 ch.:  0.015
It.:  42Obj.:  228.3703 Vol.:  0.500 ch.:  0.014
It.:  43Obj.:  228.2082 Vol.:  0.500 ch.:  0.015
It.:  44Obj.:  228.0390 Vol.:  0.500 ch.:  0.014
It.:  45Obj.:  227.8872 Vol.:  0.500 ch.:  0.014
It.:  46Obj.:  227.7236 Vol.:  0.500 ch.:  0.014
It.:  47Obj.:  227.5839 Vol.:  0.500 ch.:  0.013
It.:  48Obj.:  227.4199 Vol.:  0.500 ch.:  0.014
It.:  49Obj.:  227.2842 Vol.:  0.500 ch.:  0.013
It.:  50Obj.:  227.1235 Vol.:  0.500 ch.:  0.014
It.:  51Obj.:  226.9944 Vol.:  0.500 ch.:  0.013
It.:  52Obj.:  226.8463 Vol.:  0.500 ch.:  0.014
It.:  53Obj.:  226.7106 Vol.:  0.500 ch.:  0.013
It.:  54Obj.:  226.5837 Vol.:  0.500 ch.:  0.014
It.:  55Obj.:  226.4665 Vol.:  0.500 ch.:  0.012
It.:  56Obj.:  226.3483 Vol.:  0.500 ch.:  0.013
```

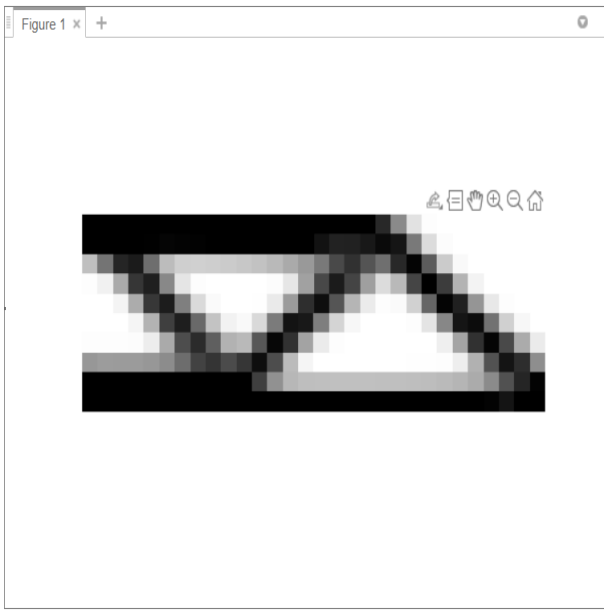
(b) Continuacion de resultados parte 2

Figura 8: Lista de resultados

Continuacion de resultados:

```
It.:  56Obj.:  226.3483 Vol.:  0.500 ch.:  0.013
It.:  57Obj.:  226.2407 Vol.:  0.500 ch.:  0.012
It.:  58Obj.:  226.1522 Vol.:  0.500 ch.:  0.013
It.:  59Obj.:  226.0659 Vol.:  0.500 ch.:  0.012
It.:  60Obj.:  226.0011 Vol.:  0.500 ch.:  0.012
It.:  61Obj.:  225.9469 Vol.:  0.500 ch.:  0.011
It.:  62Obj.:  225.9007 Vol.:  0.500 ch.:  0.012
It.:  63Obj.:  225.8573 Vol.:  0.500 ch.:  0.011
It.:  64Obj.:  225.8175 Vol.:  0.500 ch.:  0.011
It.:  65Obj.:  225.7725 Vol.:  0.500 ch.:  0.011
It.:  66Obj.:  225.7431 Vol.:  0.500 ch.:  0.011
It.:  67Obj.:  225.7040 Vol.:  0.500 ch.:  0.011
It.:  68Obj.:  225.6692 Vol.:  0.500 ch.:  0.010
It.:  69Obj.:  225.6473 Vol.:  0.500 ch.:  0.010
It.:  70Obj.:  225.6202 Vol.:  0.500 ch.:  0.010
It.:  71Obj.:  225.6070 Vol.:  0.500 ch.:  0.010
It.:  72Obj.:  225.5880 Vol.:  0.500 ch.:  0.010
```

(a) Continuacion de resultados parte 3



(b) Figura resultante

Figura 9: Resultados parte 2

8. Conclusiones

- Diego O.

Ya finalizada esta práctica se puede concluir acerca de la importancia que tiene el uso de los diferentes lenguajes de programación como lo es Matlab para poder hacer una representación mas visual de diferentes aplicaciones en la vida cotidiana, tal cual es el caso de la optimización topológica el cual al ser un análisis mecánico de una estructura tiene principal objetivo el aligeramiento estructural manteniendo las funcionalidades mecánicas del componente objetivo que ya en un caso más aterrizado a la unidad de aprendizaje seria a un objeto como una prótesis de mano, dedo entre otras.

- Ana L.

En esta práctica, por medio del software MATLAB, se llevo a cabo un analisis de un codigo de optimizacion topologico, es decir, una herramienta de análisis estructural en la cual su principal objetivo es el de aligerar estructuralmente cierto componente y en donde el peso de este es crucial. En el código en MATLAB se pudieron observar varias funciones fundamentales para realizar este tipo de optimización, al igual que se aprendio a utilizar la interfaz MATLAB para hacer posible la realización de esta práctica.

- Uriel G.

Al termino de realizar de la practica e investigar varios conceptos para la compresión del objetivo de la practica se puede concluir que MATLAB es una gran herramienta que ha ayudado a generar diferentes soluciones en varios campos de aplicación por medio de métodos como la optimización topológica que se basa principalmente en generar y mantener una estructura adecuada para diferentes piezas que son utilizadas en campos por mencionar la automotriz , manteniendo sus funcionalidades mecánicas . Cabe mencionar tambien que la generación del codigo fue complicada ya que algunos comandos no eran comprendidos fácilmente, por eso mismo fue necesario el realizar prueba y error para lograr solucionar el error marcado.

- Marcela O.

En esta práctica utilizamos el software MATLAB enfocado al análisis de un código de optimización topológico. La programación y estudio de este caso nos ayuda en los análisis de estructuras. Para el código tuvimos imprevistos y trabas por el poco uso que hemos tenido con el software, pero pudimos corregirlo con éxito. Así mismo nos dimos cuenta que el uso de programas como MATLAB son herramientas que nos ayudan e impactan de gran manera, no solo en proyectos escolares, también su uso en la industria.

- Ernesto A.

Para esta práctica realizamos una investigación para la comprensión de la optimización topologica, para ello se nos proporcionó un ejemplo el cuál pudimos comprender por medio de un código de MATLAB, el cuál pedía ciertos datos para su funcionamiento adecuado, con ello pudimos ver cómo la optimización tecnológica es usada en el campo laboral como es en el análisis de estructuras o piezas en todo caso.

Referencias

[1] Anonimo. Optimización topológica, 2019.

[2] L. C. La optimización topológica en la impresión 3d, Diciembre 2020.

[3] C. M. Optimización topológica en el diseño de elementos estructurales mecánicos, 2012.