UNIVERSIDAD AUTÓNOMA
DE NUEVO LEÓN
FACULTAD DE INGENIRÍA
MECÁNICA Y ELÉCTRICA

Laboratorio de Biomecánica

Práctica 3: Diseño de la estructura de un panorámico.

Catedrático: Yadira Moreno Vera

Martes V2

Brigada 214

Integrantes del Equipo #1	**Matricula**
Susana Rubio Medina	1798151
Melissa Lizeth Galindo Reyes	1856086
Oziel Alberto Torres Villarreal	1900260
Gabriel Eduardo Vázquez Ortega	1903060
José Manuel Reséndiz García	1907334
Miguel Ángel Martínez Villanueva	1934489

Semestre Agosto – Diciembre 2022

San Nicolás de los Garza, N.L.

18 de Octubre de 2022

Índice

Nombre y definición de la forma geométrica.

Los panorámicos se exponen a altas ráfagas de viento, por lo que su estructura ocupa ser muy rígida para soportar estas fuerzas. El espacio de diseño a evaluar será de 2 dimensiones, las cargas y los apoyos de observan en la figura 3.1.

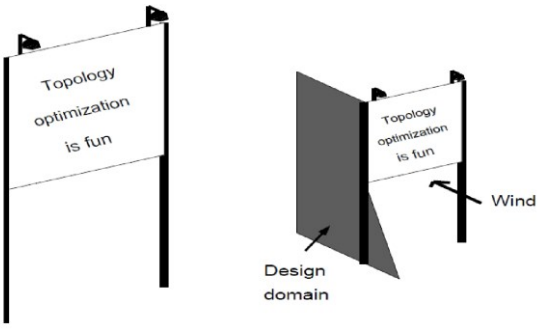


Figura 3.1: Imagen del Panorámico

En la figura 3.2 se puede ver el espacio de diseño para esta práctica. Se espera una fracción volumétrica aproximada de 0.20% del espacio de diseño. Supongamos que el panorámico es muy rígido 1, y sus patas son del mismo material que el marco.

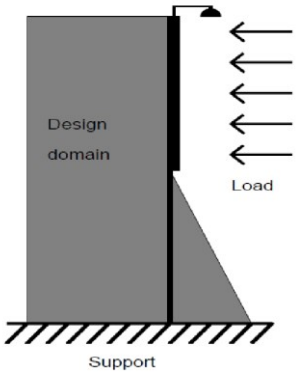


Figura 3.2: Espacio de diseño

Estado del Arte

El diseño óptimo de estructuras ha estado tradicionalmente orientado a la resolución de problemas de optimización de formas y dimensiones. Sin embargo, más recientemente ha surgido otra rama de investigación que propone modelos que proporcionan soluciones estructurales óptimas y que no requieren la definición previa de la tipología estructural: la optimización topológica de estructuras. Estas formulaciones proporcionan tanto la tipología estructural como la forma y dimensiones óptimas. Las formulaciones más habituales de estos planteamientos pretenden obtener una solución que maximice la rigidez de la estructura dadas unas limitaciones en la cantidad de material a utilizar. Estas formulaciones han sido ampliamente analizadas y utilizadas en la práctica pero presentan inconvenientes muy importantes tanto desde un punto de vista numérico como práctico.

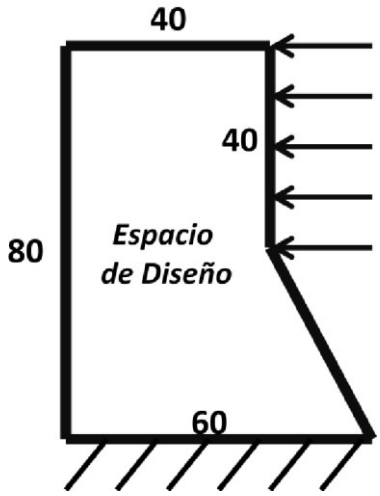
La optimización topológica de estructuras pretende obtener la distribución de una cierta cantidad de material sobre un dominio predefinido que proporciona la estructura resistente más adecuada. De acuerdo con los planteamientos originales del problema de optimización topológica propuestos por Bendsøe, el objetivo principal de estos modelos consiste en obtener distribuciones de material adecuadas indicando las partes del dominio en las que debe o no debe existir material. En consecuencia, el planteamiento original del problema da lugar a modelos de diseño óptimo con variables de diseño binarias (material o vacío). En la práctica, los modelos más habituales que se proponen para abordar este problema plantean formulaciones en las que se asume que las variables de diseño son continuas para evitar el tratamiento de problemas de optimización con variables discretas. Asimismo, se asume que el estado material en cada elemento es uniforme en todos sus puntos. Además, con estas condiciones se evita que el problema esté mal puesto desde un punto de vista teórico, tal y como ocurre si se analiza el estado material en cada punto del dominio. Por lo tanto, en la práctica se establece una variable de diseño por cada elemento: la densidad relativa, que define el estado material del mismo. La densidad relativa de cada elemento puede tomar valores entre 0, que indica elemento vacío, y 1, que indica elemento con material sólido.

Título del documento	A 99 Line Topology Optimization code written in Matlab
Fuente Bibliográfica	O. Sigmund,- 99 Line Topology Optimization Code - Department of Solid Mechanics, Building 404, Technical University of Denmark, DK-2800 Lyngby, Denmark.
Objetivo	Proveer la definición de cada una de las secciones que integran el código de optimización topología de 99 líneas en Matlab, saber ejecutarlo y como analizar los resultados obtenidos. Teniendo de base el código anterior para implementar la parte para forma una forma deseada.
Contenido	En este artículo se encuentra el código de optimización topológica utilizado para la minimización de la conformidad de estructuras cargadas estáticamente. El código esta dividido en 36 líneas para el programa principal, 12 líneas para un optimizador, 16 líneas para establecer el filtro de malla y 35 líneas para el código de elemento finito es posible añadir más líneas dependiendo de lo que se busque y es común cuando se añade más de una carga al elemento. Este gran volumen de líneas se debe a la gran cantidad de comentarios que contiene sin embargo estos comentarios nos ayudan a entender mucho más el código el cual pretende ser utilizado únicamente en un ambiente educativo.
Palabras Clave	Optimización topología, código, Matlab
Conclusión	Es un artículo muy interesante con muchas aplicaciones su función principal es entender lo que hace el código de optimización topológica a los estudiantes y que se debe realizar con los resultados.

Propuesta de diseño

Se tomarán ciertas consideraciones para la solución de esta práctica: 5 cargas, los apoyos tendrán restricciones en “X”,”Y” y el espacio de diseño para esta práctica será de:

Se eligió este diseño porque es bastante diferente a lo que encontramos y sería interesante como se implementaría para la programación. La forma de esta geometría es parecida a unos soportes vistos en algunos ejercicios de otras materias, después de este punto se vera como implementarlo en MATALB



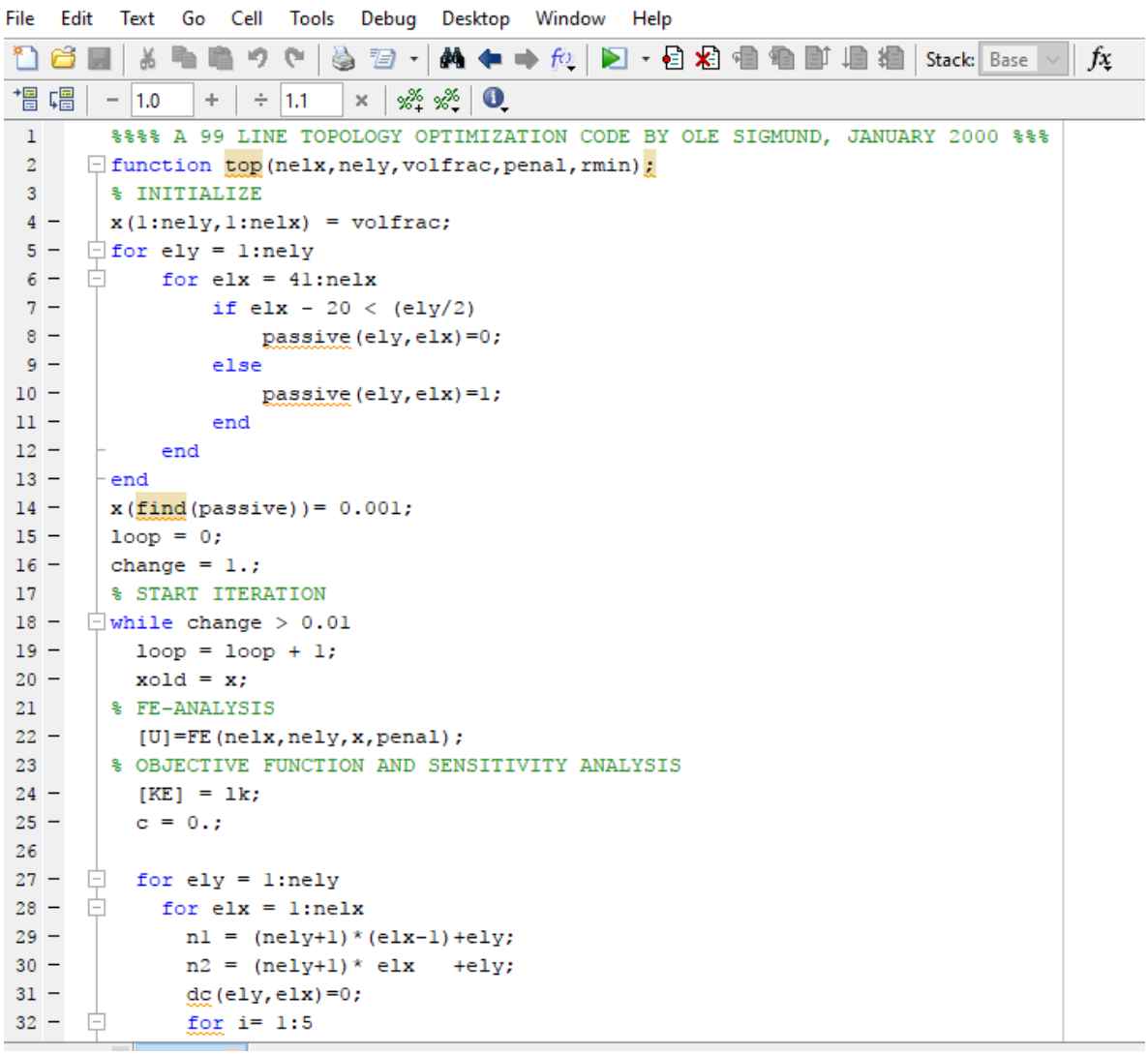
Alcance y Limitaciones.

El alcance de este diseño puede llegar a ser un proyecto grande aplicado, por ejemplo cuando nosotros vemos que la mayoría de los panorámicos tienen ciertas similitudes en cuanto a su estructura, esto es porque es la que mejor se adapta a las diferentes condiciones en las que se llegue a encontrar el panorámico. Entonces con la realización de este programa lograremos la optimización de material para realizar la estructura lo que hará que nuestro precio baje y sigamos teniendo las mismas propiedades que las que teníamos con la estructura sin optimizar.

Una vez que se tengan las medidas deseadas podríamos llegar a optimizar cualquier tipo de panorámico, incluso darle la forma menos común y con la optimización tendremos la seguridad de que el programa de MATLAB se encargue de asignar los apoyos necesarios para que nuestra estructura se encuentre rígida en cualquier momento.

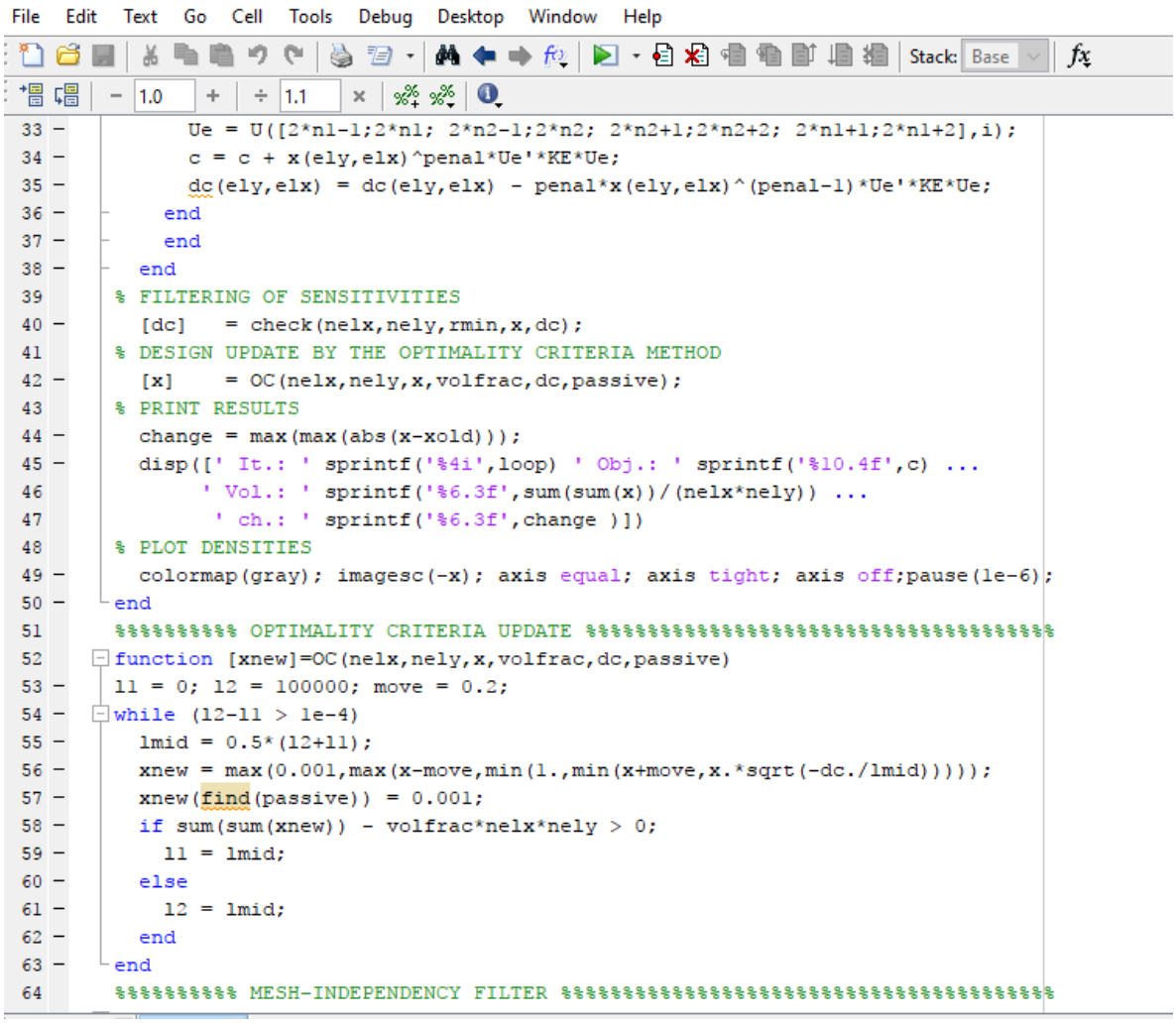
Las limitaciones son que a veces no sabremos a ciencia cierta como se verá el diseño en físico ya que al momento de tener el resultado en la simulación puede llegar a ser algo complejo en cuanto a entender la forma de la estructura.

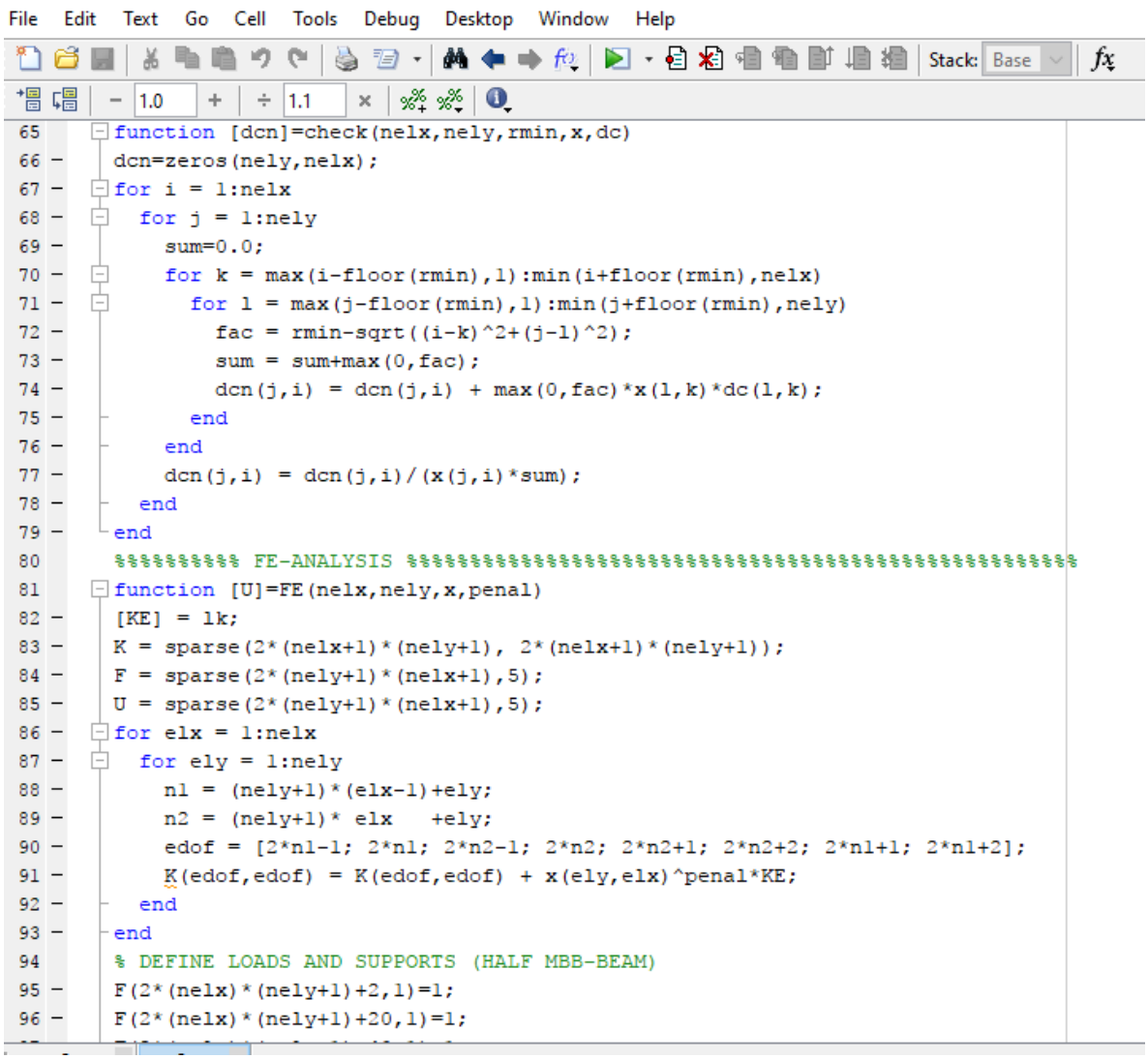
Pasos de desarrollo de la programación en Matlab.




```
%%%% A 99 LINE TOPOLOGY OPTIMIZATION CODE BY OLE SIGMUND, JANUARY 2000 %%%
function top(nelx,nely,volfrac,penal,rmin);
% INITIALIZE
x(1:nely,1:nelx) = volfrac;
for ely = 1:nely
    for elx = 41:nelx
        if elx - 20 < (ely/2)
            passive(ely,elx)=0;
        else
            passive(ely,elx)=1;
        end
    end
end
x(find(passive))= 0.001;
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;
      dc(ely,elx)=0;
      for i= 1:5
```

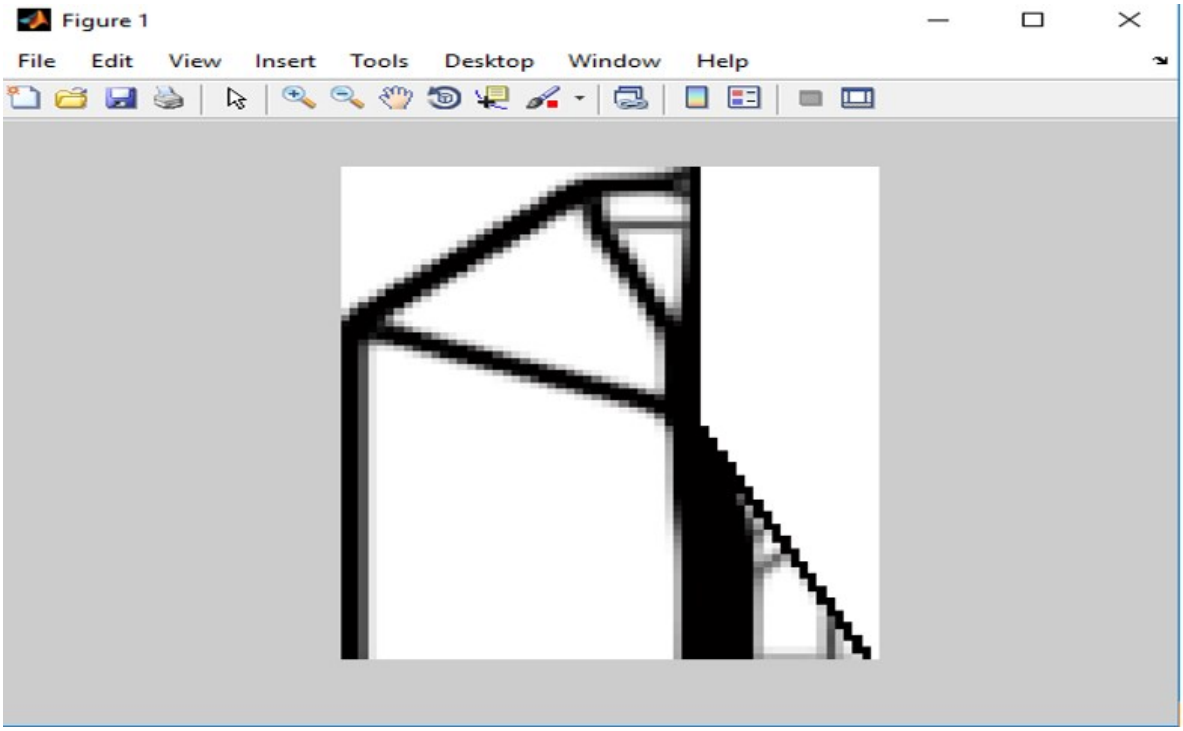

File Edit Text Go Cell Tools Debug Desktop Window Help
Stack: Base
1.0
1.1
Ue = U([2*nl-1;2*nl; 2*n2-1;2*n2; 2*n2+1;2*n2+2; 2*nl+1;2*nl+2],i);
c = c + x(ely,elx)^penal*Ue'*KE*Ue;
dc(ely,elx) = dc(ely,elx) - penal*x(ely,elx)^(penal-1)*Ue'*KE*Ue;
end
end
end
% FILTERING OF SENSITIVITIES
[dc] = check(nelx,nely,rmin,x,dc);
% DESIGN UPDATE BY THE OPTIMALITY CRITERIA METHOD
[x] = OC(nelx,nely,x,volfrac,dc,passive);
% 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
%%%%%%%%%% OPTIMALITY CRITERIA UPDATE %%
function [xnew]=OC(nelx,nely,x,volfrac,dc,passive)
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)))));
xnew(find(passive)) = 0.001;
if sum(sum(xnew)) - volfrac*nelx*nely > 0;
l1 = lmid;
else
l2 = lmid;
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-floor(rmin),1):min(i+floor(rmin),nelx)
      for l = max(j-floor(rmin),1):min(j+floor(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
%%%%%%%%%% 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),5);
U = sparse(2*(nely+1)*(nelx+1),5);
for elx = 1:nelx
  for ely = 1:nely
    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 LOADS AND SUPPORTS (HALF MBB-BEAM)
F(2*(nelx)*(nely+1)+2,1)=1;
F(2*(nelx)*(nely+1)+20,1)=1;
F(2*(nelx)*(nely+1)+20,1)=1;
F(2*(nelx)*(nely+1)+40,1)=1;
F(2*(nelx)*(nely+1)+60,1)=1;
F(2*(nelx)*(nely+1)+80,1)=1;
fixeddofs   = 2*(nely+1):2*(nely+1):2*(nelx+1)*(nely+1);
alldofs     = [1:2*(nely+1)*(nelx+1)];
freedofs    = setdiff(alldofs,fixeddofs);
% SOLVING
U(freedofs,:) = K(freedofs,freedofs) \ F(freedofs,:);
U(fixeddofs,:)= 0;
%%%%%%%%%% 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)];
```

Resultados de la optimización.

Se implementan los cambios a realizar en el código de optimización topológica de estructuras:



Código de la Programación en Matlab:

```
%%%% A 99 LINE TOPOLOGY OPTIMIZATION CODE BY OLESIGMUND, OCTOBER 1999 %%%
function topp(nelx,nely,volfrac,penal,rmin);
% INITIALIZE
x(1:nely,1:nelx) = volfrac;
loop = 0;
%Declarando vacio
for ely = 1:nely
   for elx = 1:nelx
    if (((ely-(nely*0.5)<(2*elx)-(1.36*nelx)) | (ely < (1+nely*0.5))) &
(elx > (1+ nelx)*0.666))
        passive(ely,elx) = 1;
    else
        passive(ely,elx) = 0;
    end
    end
end
x(find(passive))=0.001;
change = 1.;
% START ITERATION
while change > 0.01
loop = loop + 1;
xold = x;
% FE-ANALYSIS
    [U]=FE(nelx,nely,x,penal);
%13 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; %19
        dc(ely,elx) = 0.;
        for i = 1:5
        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) = dc(ely,elx)-penal*x(ely,elx)^(penal-1)*Ue'*KE*Ue;
        end
    end
end
%25 FILTERING OF SENSITIVITIES
[dc] = check(nelx,nely,rmin,x,dc);
%27 DESIGN UPDATE BY THE OPTIMALITY CRITERIA METHOD
[x] = OC(nelx,nely,x,volfrac,dc,passive);
%29 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
%40 %%%%%%%%% OPTIMALITY CRITERIA UPDATE %%%%%%%%%
function [xnew]=OC(nelx,nely,x,volfrac,dc,passive)
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)))));
xnew(find(passive)) = 0.001;
if sum(sum(xnew)) - volfrac*nelx*nely > 0;
l1 = lmid;
else
l2 = lmid;
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
%65 %%%%%%%%% 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),5); U =zeros(2*(nely+1)*(nelx+1),5);
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*nelx*(nely+1)+2,1)          = 1;
F(2*nelx*(nely+1)+(nely/4),2)   = 1;
F(2*nelx*(nely+1)+(nely/2),3)   = 1;
F(2*nelx*(nely+1)+(nely),4)     = 1;
F(2*nelx*(nely+1)+(nely*1.2),5) = 1;

fixeddofs =2*(nely+1):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;
%%%%%%%%%% 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)];
```

Conclusiones del Equipo 1

Susana Rubio Medina. – Matlab puede ser usado, como ya hemos visto con anterioridad para optimizar productos de diversas ramas, sobre todo de la ingeniería. En esta práctica aprendí que también se puede hacer la optimización de un panorámico, lo cual es muy interesante, porque no me había imaginado que Matlab pudiera tener este tipo de usos. Una vez elaborado el código de la práctica 3, se corrió el programa, y en la pantalla principal de MATLAB se realizaron cada uno de los pasos de la práctica (desarrollo y ejercicio propuesto, resultados.). En esta práctica queda demostrado la versatilidad del código para poder representar cualquier tipo de trabajo. Lo que es difícil de representar serían los vacíos con formas de pendientes dado a que estos tienen que ser condiciones lineales que son difíciles en código if-else.

Melissa Lizeth Galindo Reyes. En esta práctica seguimos aplicando la optimización como método de trabajo para resolver nuestro problema que en este caso el objetivo era diseñar una estructura de un panorámico, como parte de esto definimos el nombre y la forma geométrica, estado del arte y diseño como parte fundamental de este trabajo, y así aplicamos el código desarrollado en MATLAB que nos ayudo a comprobar que nuestra propuesta de diseño y comprobar nuestros resultados. No imaginaba que MATLAB tuviera tanto alcance en la parte de una propuesta de diseño por medio de un código. Así mismo me pareció una práctica interesante ya que dentro de lo cotidiano detrás existe todo un método completo para crearlo.

Oziel Alberto Torres Villarreal. – Esta practica me hizo pesar en la estructura que hay detrás de los panorámicos, solía pensar que no eran mas que un letrero y no prestaba demasiada atención en su estructura, tras el análisis me parece que es muy importante que la estructura de un panorámico sea muy resistente por las fuertes corrientes de aire. Me parece impresionante la diversidad de estructuras que se pueden analizar con este código y como una herramienta como Matlab es tan útil, al ver Matlab uno piensa en una calculadora más en la que se puede programar, pero a medida que se indaga en las capacidades que tiene el software puede llegar a ser hasta abrumador la cantidad de aplicaciones que tiene.

Gabriel Eduardo Vázquez Ortega. – Continuando nuevamente con lo que es la optimización tipológica con la ayuda de Matlab, esta vez toco realizar un panorámico, a simple "vista" uno cree que es sencillo, ya que solo piensa que es el letrero y ya, pero hay un factor muy importante el cual es el clima, puede haber mucho viento un día u otro una fuerte lluvia, y estos factores influyen en la creación de este panorámico, como lo fue en el caso de esta práctica que nos decía que este panorámico estaba en constantes ráfagas de viento, por lo cual la estructura de este panorámico tenía que ser muy específica para poder resistir este tipo de clima. Por último, estoy asombrado por la cantidad de usos y aplicaciones que le podemos dar al software "Matlab" en cuestión, nunca me llegue a imaginar que sería capaz de crear una estructura entera con un simple código, dentro de lo que cabe la palabra "simple".

José Manuel Reséndiz García. – Esta práctica fue similar a la anterior con cuestiones de modificar ciertos parámetros para dar la geometría deseada, fue algo difícil esa parte, pero se puedo conseguir sin mucha complicación, sobre la programación no tengo dudas al respecto. De los demás puntos de la práctica es cuestión de investigar acerca de la geometría que se quiere dar , solo lo complicado era la programación, sin más que decir no tengo una inquietud acerca de esta practica.

Miguel Ángel Martínez Villanueva. – Con la práctica finalizada hemos encontrado otra aplicación para la optimización de esta estructura que en esta práctica fue la de un panorámico, se nos planteó que estos están expuestos a ráfagas de viento por ello es importante tener una estructura rígida pero necesitamos realizar esta optimización para ver la forma más correcta de esta estructura, con esto nos damos cuenta que el software de matlab nos puede servir para realizar optimizaciones de todo tipo de piezas/estructuras siempre y cuando tengamos en claro la propuesta de diseño a realizar, la práctica se logró sin algún problema y con la programación tuvimos resultados gratificantes.

Bibliografía

- ✓ De Facultad de Ingeniería, Universidad Autónoma de San Luis Potosí. (Ed.). (2013). Herramienta de diseño bidimensional que integra optimización, topología y forma. MEMORIAS DEL XIX CONGRESO INTERNACIONAL ANUAL DE LA SOMIM. http://somim.org.mx/memorias/memorias2013/pdfs/A1/A1_35.pdf [1]
- ✓ Álvarez, A. (s. f.). Geometría de la bici de MTB: medidas, ángulos y lo que significan. Recuperado 15 de septiembre de 2022, de https://www.mtbpro.es/afondo/geometria-de-la-bici-de-mtb-medidas-angulos-y-lo-que-significan#:%7E:text=La%20geometr%C3%ADa%20de%20una%20bici%20mide%20las%20longitudes,forma%20de%20los%20tubos%20sea%20convencional%20para%20medirlos. [2]
- ✓ Optimización Topológica | Catec. (s. f.). Recuperado 15 de septiembre de 2022, de http://www.catec.aero/es/materiales-y-procesos/l%C3%ADnea-de-investigaci%C3%B3n/optimizaci%C3%B3n-topol%C3%B3gica [3]
- ✓ Kristy Foss. PEDALIA, ¿Qué material es el mejor para el cuadro de mi bicicleta?, Recuperado: 19 de septiembre de 2022, URL: https://pedalia.cc/material-mejor-cuadro-bicicleta/[4]
- ✓ J. París, S. M. (2012 de enero-marzo). *Una formulación de mínimo peso con restricciones en tensión en optimización topológica de estructuras*. Obtenido de Science Direct: https://www.sciencedirect.com/science/article/pii/S0213131511000368
- ✓
- ✓