Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

Laboratorio de biomecánica

“PRÁCTICA 3. DISEÑO DE LA ESTRUCTURA DE UN PANORÁMICO”

Instructor(a): Ing. Isaac Estrada

Brigada: 309

Nombre	Matrícula	Carrera
Edelmiro Eugenio Garcia Sanchez	1640109	IMTC
Jesús Alberto Funes Mendoza	1798459	IMTC
Cristian Arturo Garza Cavazos	1909877	IMTC
Elias Alejandro García Bueno	1676718	IMTC

Semestre Agosto – Diciembre 2022

Día 17 del mes Octubre del año 2022

Ciudad Universitaria, San Nicolás de los Garza, Nuevo León

Objetivo:

El estudiante deberá presentar un reporte con la solución numérica computacional del problema de la simulación del desempeño mecánico de componentes mecánicos por medio del método de MATLAB, esto para desarrollar en el estudiante la capacidad de análisis, implementación y solución de un problema propuesto.

deberá presentar una propuesta de análisis de formas y de la programación para la ejecución de la optimización (descripción funcional) de características de trabajo específicas que presenta la(s) ventaja(s) (mencionar ventajas).

Marco teórico:

Una estructura panorámica es el soporte sobre el cual se posicionará un anuncio publicitario, ya sea de una cara o de tres caras.

¿Cuál es la estructura de un espectacular?

Son sumamente visibles para un gran número de personas, y tal vez esa es su mayor virtud.

Son relativamente económicos en comparación con otros medios de comunicación.

Ofrece a las marcas la oportunidad de llegar tanto a su objetivo como a consumidores potenciales.

¿Donde se encuentran?

Estas estructuras usualmente se encuentran en medio de diversos paisajes urbanos y sostienen diseños publicitarios con el objetivo de promocionar un producto, servicio o transmitir un mensaje.

Cada país tiene ciertas normativas en cuanto a dónde es apropiado o no colocar estos soportes para anuncios publicitarios.

En algunos no está permitido que se construyan estructuras panorámicas a los lados de autopistas porque estos pueden distraer a los conductores.

Los panorámicos se exponen a altas ráfagas de viento, por lo que su estructura ocupa ser muy rígida para soportar estas fuerzas.

En la figura 1 se muestra el panorámico que será el espacio de diseño a evaluar, éste será de 2

dimensiones, con cargas y apoyos como se muestra a continuación:

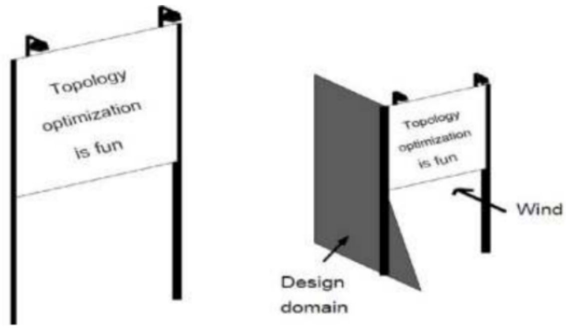


Figura 1. Imagen del panorámico.

Existen diversos materiales que las agencias especializadas usan en la creación y diseño para estas estructuras. Usualmente los postes panorámicos están construidos de metal y acero para que sean lo suficientemente resistentes al clima, lluvias y cualquier otro fenómeno de la naturaleza. Por otra parte, el panorámico en sí mismo son hechos de lona, vallas de PVC, plástico, tela, metal o acrílico. También existen espectaculares digitales o electrónicos que tienen luces, pantallas eléctricas y música.

En la figura 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

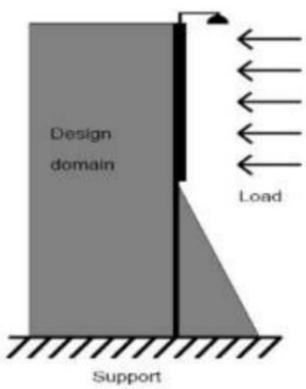
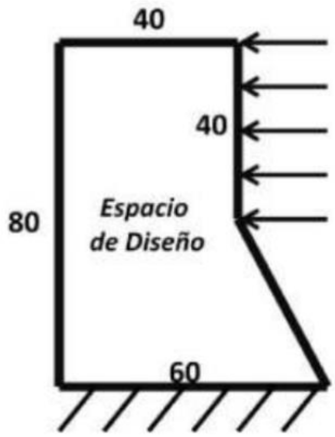


Figura 2. Espacio de diseño

Desarrollo

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:



La optimización topológica suele tener lugar hacia el final del proceso de diseño, cuando la pieza deseada necesita tener un peso menor o usar menos materiales.

El diseñador trabaja para descubrir ciertos parámetros preestablecidos, como las cargas aplicadas, el tipo de material, las limitaciones del modelo y su organización en el espacio. En primer lugar, la optimización topológica estructural determina el espacio de diseñomínimo permisible que es necesario para optimizar la forma del producto.

Después, deforma virtual, el software de optimización topológica aplica presión sobre el diseño desde distintos ángulos, pone a prueba su integridad estructural e identifica el materialinnecesario

Estado e arte	
Titulo del documento	Un código de optimización de topología de 99 líneas escrito en Matlab
Fuente biobibliográfica	Departamento de Mecánica de Sólidos, Edificio 404, Universidad Técnica de Dinamarca, DK-2800 Lyngby, Dinamarca¶e-mail: sigmund@fam.dtu.dk, DK O. Sigmundo) (Department of Solid Mechanics, Building 404, Technical University of Denmark, DK-2800 Lyngby, Denmark¶e-mail: sigmund@fam.dtu.dk, DK O. Sigmund)
objetivo	Dar a conocer cada una de las secciones que integran el código de optimización topológica de 99 lineas en Matlab , saber ejecutar el análisis de 99 lineas en Matlab saber ejecutar el analisis del código y observar los resultados obtenidos
contenido	El documento presenta una implementación compacta de Matlab de un código de optimización de topología para minimizar el cumplimiento de estructuras cargadas estáticamente. El número total de líneas de entrada de Matlab es 99, incluido el optimizador y la subrutina de elementos finitos. Las 99 líneas se dividen en 36 líneas para el programa principal, 12 líneas para el optimizador basado en criterios de optimización, 16 líneas para un filtro de independencia de malla y 35 líneas para el código de elementos finitos. De hecho, excluyendo las líneas de comentarios y las líneas asociadas con la salida y el análisis de elementos finitos, se muestra que solo se requieren 49 líneas de entrada de Matlab para resolver un problema de optimización de topología bien planteado. Al agregar tres líneas adicionales, el programa puede resolver problemas con múltiples casos de carga. El código está destinado a fines educativos
Palabras clave	Optimización topológica, algoritmo.
conclusión	En este articulo investigado sobre la topología y su análisis nos habla como constituye dicho código de 99 líneas y cual es la idea principal que este debe cumplir y que es lo que debe demostrar al ejecutarlo

Procedimientos de la programación

Usaremos el código de la practica 1 dándole unos cambios   en ciertas variables para poder implementarlo para esta tarea   (el panorámico)

```
function toppract(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

%%%%%%%%%% 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

%%%%%%%%%% 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

%%%%%%%%%% 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
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)];
```

Cambio Para Fuerzas Múltiples

Se tiene que editar el Script para poder ingresar las fuerzas necesarias observando vemos que tenemos 5 y para cambiar el anclaje del espacio de diseño a otra posición se necesita modificar o cambiar algunas líneas.

```
%%%%%%%%%%% 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);
```

```
%%%%%%%%%%% 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 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
```

```
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
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
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)];
```

```
% 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);
```

Empotramiento diagonal

Para nosotros poder crear el espacio en blanco en la parte inferior derecha necesitas modificar el código original para poder crear el espacio conocido con los siguientes cambios en la codificación para llegar al resultado

```
function toppract(nelx,nely,volfrac,penal,rmin)
% INITIALIZE
x(1:nely,1:nelx) = volfrac;
loop = 0;
change = 1.;
```

```
function toppract(nelx,nely,volfrac,penal,rmin)
% INITIALIZE
x(1:nely,1:nelx) = volfrac;
loop = 0;
%DECLARACION DE 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)*0.6666))
            passive(ely,elx)=1;
        else
            passive(ely,elx) = 0;
        end
    end
end
x(find(passive))=0.001;
change = 1.;
```

```
% DESIGN UPDATE BY THE OPTIMALITY CRITERIA METHOD
[x] = OC(nelx,nely,x,volfrac,dc);
```

```
% DESIGN UPDATE BY THE OPTIMALITY CRITERIA METHOD
[x] = OC(nelx,nely,x,volfrac,dc,passive);
```

```
%%%%%%%%%% OPTIMALITY CRITERIA UPDATE %%%%%%%%%
function [xnew]=OC(nelx,nely,x,volfrac,dc)
```

```
%%%%%%%%%% OPTIMALITY CRITERIA UPDATE %%%%%%%%%
function [xnew]=OC(nelx,nely,x,volfrac,dc,passive)
```

```
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;
```

```
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;
```

```
%Práctica #3 Laboratorio Biomecánica Equipo 7

function toppract(nelx,nely,volfrac,penal,rmin)
% INITIALIZE
x(1:nely,1:nelx) = volfrac;
loop = 0;
%DECLARACION DE 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)*0.6666))
            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);
% 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
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
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
```

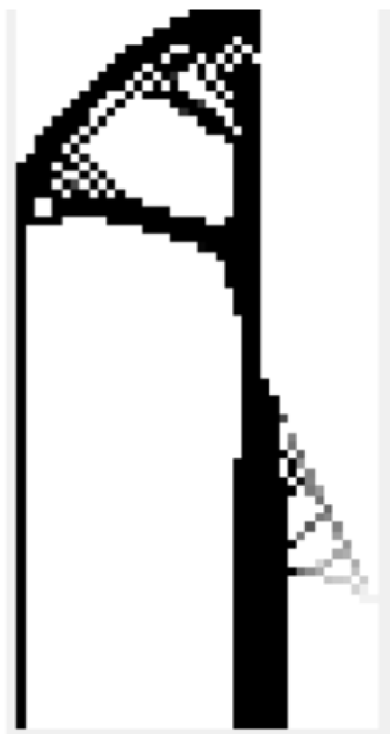
```
%%%%%%%%%% OPTIMALITY CRITERIA UPDATE %%%%%%%%%
function [xnew]=OC(nelx,nely,x,volfrac,dc,passive)
l1 = 0;
l2 = 100000;
move = 0.2;
while (l2-l1 > 1e-4)
lmid = 0.5*(l2+l1);
xnew = max(0.001,max(x-move,min(1.,min(x+move,x.*sqrt(-dc./lmid)))));
xnew(find(passive)) = 0.001;
if sum(sum(xnew)) - volfrac*nelx*nely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

%%%%%%%%%% 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 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
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)];
```



Conclusión

Edelmiro Eugenio Garcia Sanchez 1640109

En esta practica encontramos una cierta similitude con la primera practica ya que tuvimos que modificar la codificacion para poder implementarlo en esta y ver el analisis del comportammiento estatico de una pieza en este caso el panoramico y asi poder plasmarlo en matlab para nuestra codificacion, y gracias a eso aprendimos cual es el comportamiento de un analisis estatico de una pieza por si sola (intemperie).

Jesús Alberto Funes Mendoza 1798459

En esta práctica se siguen reforzando los conocimientos adquiridos en las prácticas anteriores, realmente el reto que vimos en esta práctica fue el ver cómo queríamos que fuera el panorámico y sus estándares ya que en cuestión de código y matlab fue algo muy similar a lo visto ya anteriormente.

Cristian Arturo Garza Cavazos 1909877

Al terminar esta práctica Podemos decir que nos quedamos con un grato aprendizaje, en donde vemos el comportamineto del sistema, primero tuvimos que modificar el codigo de cierta manera, en donde gracias a ellos vimos el analisis estatico de la pieza. Esto nos sirvio para poder mejorar nuestra interpretación de los resultados.

Elias Alejandro García Bueno 1676718

En la practica, observemos el funcionamiento de las iteraciones en Matlab, en la práctica podemos observar como Matlab nos da información de las iteraciones que realiza el código utilizando un código que tarda más de una hora en realizar todas las iteraciones, lo que lo convierte en una especie de ejercicio Eso lleva mucho tiempo, el problema que surge es ajustar el código, pero estos son solo algunos cambios en sí mismos.