

# Laboratorio de Biomecánica

## Práctica 1

### EQUIPO 6

Christopher Alejandro Monsivais Ramos 1899218

Alessandra González Torres 1895846

Angela Itzel Rodríguez Torres 1925284

Abdiel Isaac Rodríguez Pinales 1842044

Martin Alejandro Villanueva Cortes 1870820

Alejandro Sebastian Marin Cota 1814201

6 de septiembre de 2022

## 1. Introducción

### Programación

La programación es el proceso de creación de programas informáticos. Esta definición se puede interpretar de la siguiente manera. La programación no es más que una explicación a la computadora de qué, en qué forma y cómo llegar al usuario. En otras palabras, es una especie de arte de traducir los deseos de una persona al lenguaje de la máquina. La idea principal es crear un algoritmo y traducirlo a un lenguaje de programación. De hecho, un lenguaje de programación es un conjunto de reglas para describir comandos y funciones predefinidas. Cada lenguaje de programación limita al desarrollador a un conjunto estrictamente predeterminado de palabras clave y comandos que pueden ser utilizados en el desarrollo. El nombre de programación geométrica (P.G.) se debe a que se utiliza la generalización de la desigualdad media aritmética geométrica para resolver algunos problemas de optimización. El nombre de Programación Geométrica es debido a que esta técnica utiliza variantes M de la desigualdad media aritmética geométrica.[3]

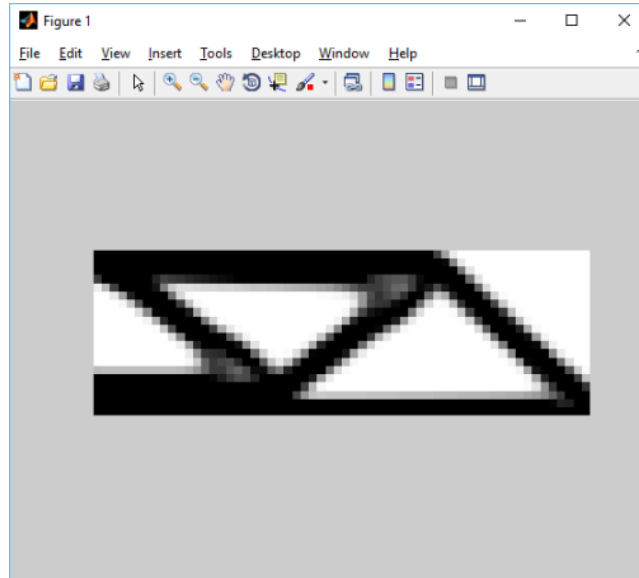


Figura 1: Ejemplo de la forma geométrica

## 2. Estado del arte

Objetivo: Dar a conocer cada una de las secciones que integran el código de optimización topológica de 99 líneas en Matlab, saber ejecutar el análisis del código y observar los resultados obtenidos.

Contenido: El artículo presenta una implementación compacta en Matlab de un código de optimización topológica para la minimización de la conformidad de estructuras cargadas estáticamente. El número total de líneas de entrada de Matlab es de 99 incluyendo el optimizador y la subrutina de elementos finitos. Las 99 líneas están divididas en 36 líneas para el programa principal, 12 líneas para el optimizador basado en criterios de optimización, 16 líneas para un filtro de dependencia de malla y 35 líneas para el código de elementos finitos. De hecho, excluyendo las líneas de comentario y las asociadas a la salida y a análisis de elementos finitos se ansera que solo se necesitan 4u inear de entrada, de Matlab para resolver un problema de optimización topológica bien planteado. Añadiendo tres líneas adicionales, el programa puede resolver problemas con múltiples casos de carga. El código está pensado para fines educativos.[2][1]

Palabras claves: Optimización Topológica, criterios de optimización, web mundial, código Matlab

## 3. Procedimiento de la programación

Codigo que se utiliza:

```
%%% A 99 LINE TOPOLOGY OPTIMIZATION CODE BY OLESIGMUND, OCTOBER 1999 %%%
function topp1(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
end

```

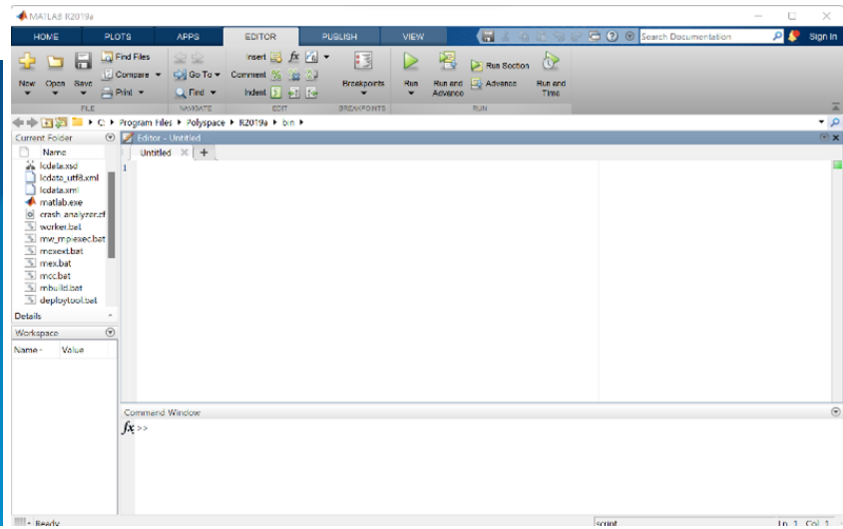
```

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

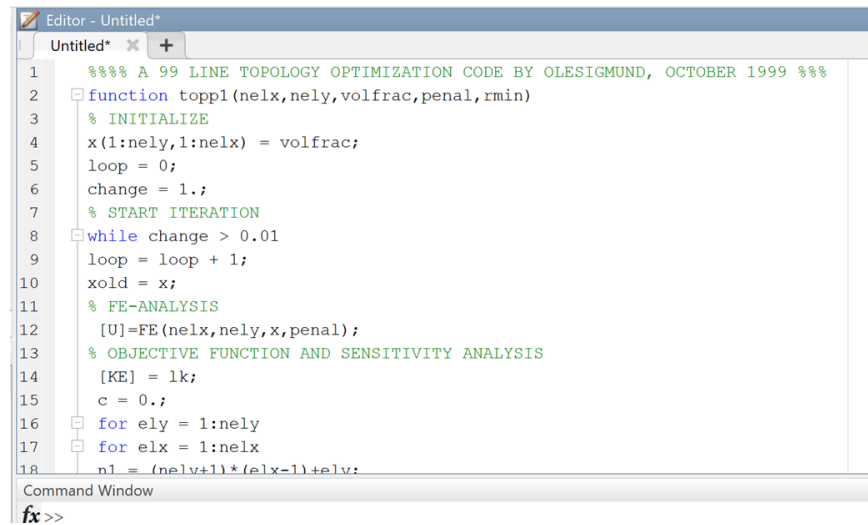
```

## 4. Pasos que seguir para la programación en MATLAB

1) Abrir MATLAB y esperar a que éste se inicialice, y muestre su pantalla principal.



2) Una vez en la pantalla de inicio de MATLAB es necesario abrir un archivo nuevo en el editor de texto, dentro del cual será necesario escribir el código proporcionado.

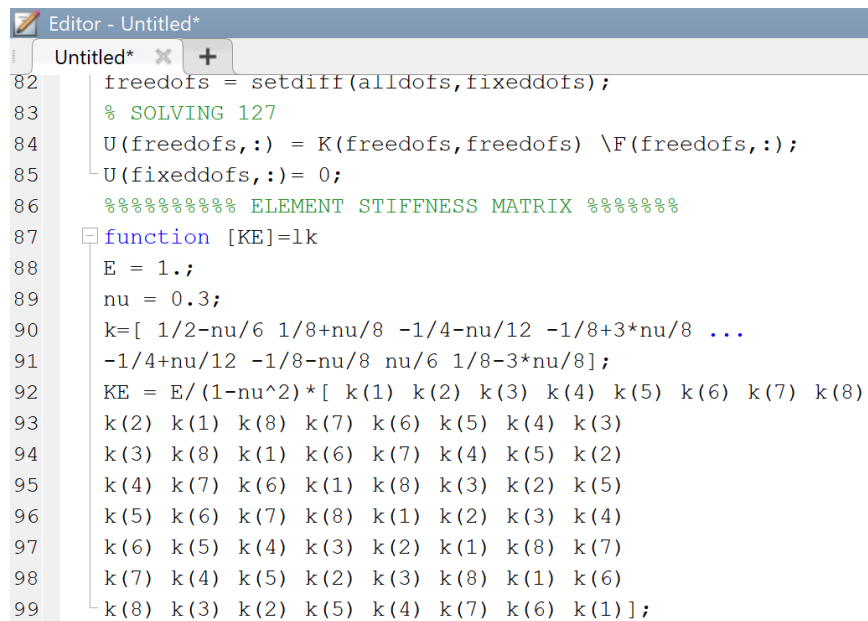


```

Editor - Untitled*
Untitled* x +
1  %%% A 99 LINE TOPOLOGY OPTIMIZATION CODE BY OLESIGMUND, OCTOBER 1999 %%%
2  function topp1(nelx,nely,volfrac,penal,rmin)
3  % INITIALIZE
4  x(1:nely,1:nelx) = volfrac;
5  loop = 0;
6  change = 1.;
7  % START ITERATION
8  while change > 0.01
9  loop = loop + 1;
10 xold = x;
11 % FE-ANALYSIS
12 [U]=FE(nelx,nely,x,penal);
13 % OBJECTIVE FUNCTION AND SENSITIVITY ANALYSIS
14 [KE] = lk;
15 c = 0.;
16 for ely = 1:nely
17 for elx = 1:nelx
18 nl = (nely+1)*(elx-1)+ely;
Command Window
fx>>

```

Figura 2: Código de Matlab P1



```

Editor - Untitled*
Untitled* x +
82 freedofs = setdiff(alldofs,fixeddofs);
83 % SOLVING 127
84 U(freedofs,:) = K(freedofs,freedofs) \F(freedofs,:);
85 U(fixeddofs,:)= 0;
86 %%%%%%%%% ELEMENT STIFFNESS MATRIX %%%%%%%%%
87 function [KE]=lk
88 E = 1.;
89 nu = 0.3;
90 k=[ 1/2-nu/6 1/8+nu/8 -1/4-nu/12 -1/8+3*nu/8 ...
91 -1/4+nu/12 -1/8-nu/8 nu/6 1/8-3*nu/8];
92 KE = E/(1-nu^2)*[ k(1) k(2) k(3) k(4) k(5) k(6) k(7) k(8)
93 k(2) k(1) k(8) k(7) k(6) k(5) k(4) k(3)
94 k(3) k(8) k(1) k(6) k(7) k(4) k(5) k(2)
95 k(4) k(7) k(6) k(1) k(8) k(3) k(2) k(5)
96 k(5) k(6) k(7) k(8) k(1) k(2) k(3) k(4)
97 k(6) k(5) k(4) k(3) k(2) k(1) k(8) k(7)
98 k(7) k(4) k(5) k(2) k(3) k(8) k(1) k(6)
99 k(8) k(3) k(2) k(5) k(4) k(7) k(6) k(1)];

```

Figura 3: Código de Matlab P2

3) El siguiente paso consiste en salvar el archivo. Se recomienda que el archivo se guarde en el directorio raíz de MATLAB que por default muestra. En caso de no ser así, debemos de navegar a “Mis Documentos” y guardar ahí el script recién creado.

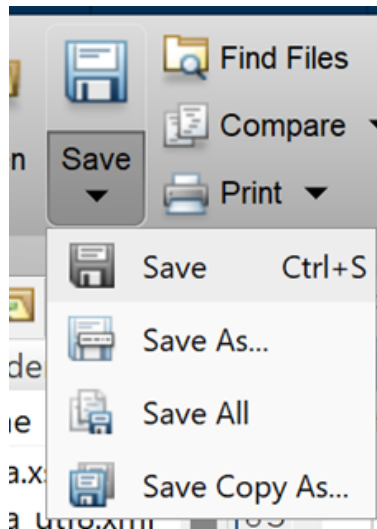


Figura 4: Como salvar un archivo

4) Una vez guardado el script en el directorio correcto, solo hace falta corroborar que el intérprete de MATLAB se encuentre en el mismo directorio. Esto se hace desde la pantalla principal de MATLAB. Para la versión R2019a del software, el directorio actual del intérprete se encuentra en la barra de herramientas superior.

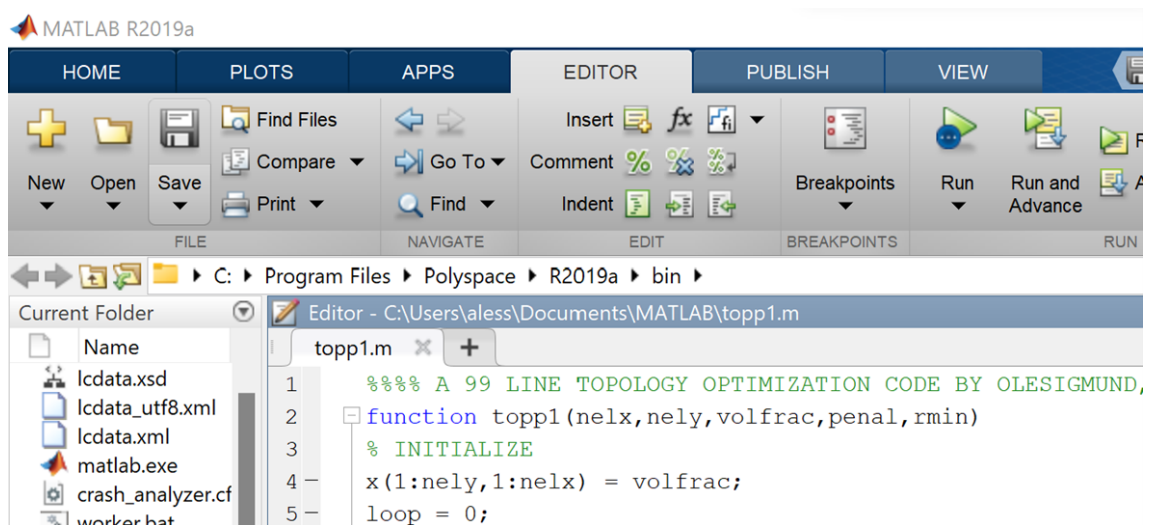


Figura 5: Barra de herramientas superioro

5) Por último, hay que ejecutar el programa desde la ventana de comando de MATLAB. El código que se proporcionó viene preparado para optimizar un dominio de diseño con cargas y restricciones, este caso en particular es evaluado y simulado cuando escribimos desde la línea de comando de MATLAB “topp1(60,20,0.5,3,1.5)”.

- nelx y nely: Son el número de elementos en las direcciones horizontales (x) y verticales (y).
- volfrac: Es la fracción de volumen.
- penal: Es el poder de penalización.
- rmi: Es el tamaño del filtro (dividido por el tamaño del elemento).

## 5. Implementación de la programación

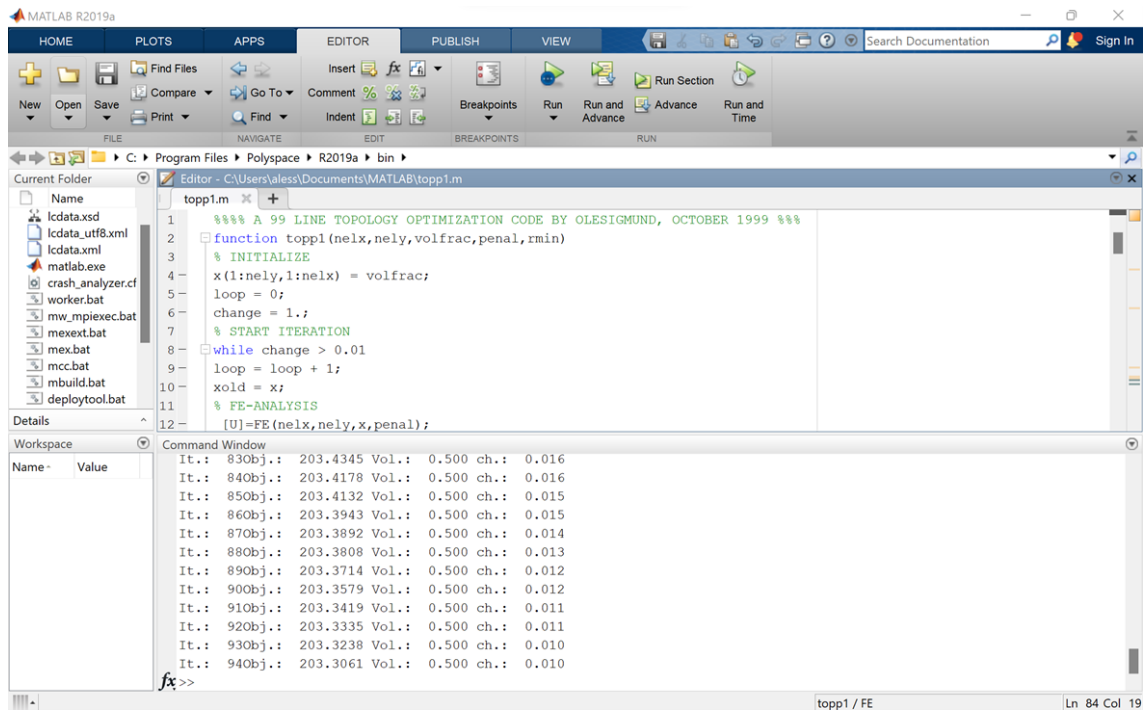


Figura 6: Vista MatLab



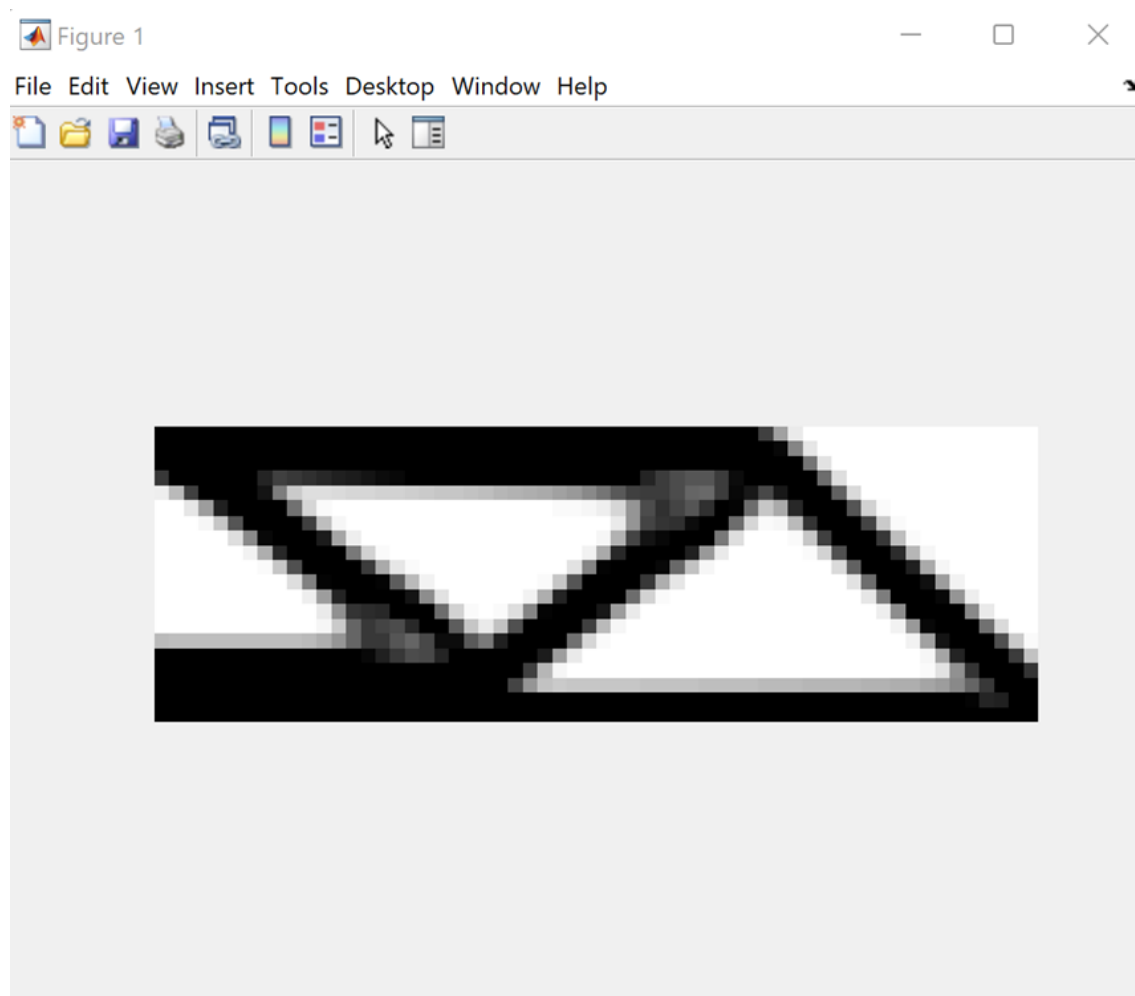


Figura 7: Vista de la figura que se realiza

## 6. Conclusiones

Christopher Alejandro Monsivais Ramos:

Como conclusión a esta práctica puedo decir que nos sirvió mucho para aprender a simular una forma geométrica y a ver un poco de lo que es la optimización topológica, además de que al trabajar con Matlab nos da más experiencia utilizando el software y ahora sabemos otra aplicación que se le puede dar al programa.

Alessandra Gonzalez Torres:

En esta práctica empezamos a investigar sobre el tema de la geometría y de que trata, sin embargo lo que más me sorprendió fue que se pudiera usar el software de Matlab para llevar a cabo la simulación de una forma geométrica. Con eso me pude dar cuenta de el gran alcance que se ha llegado a tener con la tecnología y en todo lo que nos puede ayudar.

Angela Itzel Rodriguez Torres:

En la realización de esta práctica adquirí conocimientos acerca de cómo la optimización topológica nos ayuda en la creación de un modelo en el que se aplican diferentes cargas para una pieza y usamos el software Matlab para codificar y ejecutarlo, fue algo sencillo ya que algunos del equipo incluyéndome ya habíamos usado Matlab antes y con los pequeños conocimientos que cada quien tenia pudimos complementarlo, errores que se corrigieron.

Abdiel Isaac Rodriguez Piñales:

Podemos concluir que los avances de tecnología han sido tales que se ha conseguido la optimización de los nuevos diseños. Como en el método de optimización topológica la cual sirve para reducir los materiales a utilizar los nuevos diseños. Esto ha sido útil en áreas como la ingeniería aeroespacial donde es crucial el volumen y materia de los componentes a utilizar. Todo es gracias al avance en el desarrollo de software.

Alejandro Sebastian Marin Cota:

En esta práctica se aprendió la programación en matlab con la optimización topológica observando el comportamiento del material, el cual fue el esperado por nosotros gracias a la investigación hecha antes de la optimización topológica. Matlab es un programa muy útil para múltiples ámbitos de la ingeniería y en este caso fue la mejor opción la implementación de la practica en este software.

Martín Alejandro Villanueva Cortes:

Con esta primer practica pude aprender sobre como utilizar el MatLab para poder crear una forma geométrica lo cual me llama mucho la atención ya que anteriormente llegue a utilizar el programa de MatLab para resolver matrices, no me imaginaba que también lo podría llegar a utilizar para realizar dicha forma geométrica. Por ultimo la manera en la que realizamos el archivo por medio de la aplicación de overleaf me pareció muy interesante para realizar los reportes sobre este laboratorio.

## Referencias

- [1] 99 line topology optimization code - o. sigmund, department of solid mechanics, building 404, technical university of denmark, dk-2800 lyngby, denmark.
- [2] Optimización topológica — catec., Agosto 2017.
- [3] ¿qué es la programación? ceupe, Julio 2020.