UANL
UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FIME
FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA

UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA

Laboratorio de Biomecánica

Práctica 1. Optimización Topológica y Análisis

Nombre:	Matricula:	Carrera:
Campos Miranda Julio Emilio	1917048	IMTC
Cruz Puente Sergio Eduardo	1903115	IMTC
Cynthia Belen Guerrero Pardo	1026215	IMTC
Jorge Luis Salinas Garza	1916367	IMTC
Acuña Rodríguez Christopher Russ	1894698	IMTC

Maestro: Ing Isaac Estrada

Grupo: 309

Hora: N5

Cd. Universitaria, San Nicolás de los Garza.

21 de Septiembre de 2022

Práctica 1. Optimización Topológica y Análisis

Objetivo

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

Introducción

Para esta práctica, se utiliza un código de optimización topológica, del cual se definen las condiciones iniciales al inicio de la práctica para después poder realizar la simulación del programa y observar el funcionamiento de esta optimización en un análisis.
El código tendrá aproximadamente 99 líneas, de las cuales se dividirán en secciones y nos llevarán al mismo resultado.

Nombre y Definición de la Programación

Programación es la acción de programar que implica ordenar, estructurar o componer una serie de acciones cronológicas para cumplir un objetivo. La programación puede ser aplicada para eventos sociales, a medios de comunicación y al mundo informático de las computadoras.

En informática, la programación es el uso de lenguajes informáticos para imprimir en un sistema computacional funciones y procesos deseados. La programación de una computadora es la forma de indicar a la computadora qué es lo que tiene que hacer.
Un lenguaje de programación es un lenguaje artificial, diseñado para representar algoritmos de forma inteligible para las computadoras.
Existen muchos lenguajes de programación, pero el único que entiende el ordenador directamente, como se ha comentado anteriormente, es el lenguaje máquina, cuyas instrucciones están codificadas en forma de secuencias de ceros y unos (bits). El resto de lenguajes necesitan traducir o interpretar las instrucciones al lenguaje máquina.

La geometría computacional es una rama de las ciencias de la computación dedicada al estudio de algoritmos que pueden ser expresados en términos de la geometría. Algunos de los problemas puramente geométricos surgen del propio estudio de dichos algoritmos, y este tipo de problemas también se considera parte de la geometría computacional. También se considera una rama gráfica del ordenador.

Hasta hace poco, Geometría Computacional se refería al diseño y análisis de algoritmos geométricos, pero en los últimos años ha ampliado su campo, y ahora también incluye el estudio de problemas geométricos desde un punto de vista computacional, incluyendo también convexidad computacional, topología computacional y complejidad combinatorial de disposiciones de poliedros.

En los últimos años ha aumentado el número de áreas en las que se aplican los resultados de esta disciplina. Entre las mismas se incluyen la ingeniería, cristalografía, diseño asistido por computador, sistemas de posicionamiento global, robótica, sistemas de detección de errores, modelado geométrico, gráficos por ordenador, optimización combinatorial, visión por ordenador, reconocimiento de patrones y modelado sólido.

Un ejemplo de esto sería el de encontrar el par de puntos más cercanos puede ser implementado de forma sencilla mediante una búsqueda por fuerza bruta, calculando la distancia entre las n(n-1)/2 combinaciones posibles de pares de puntos y eligiendo la menor, pero esta solución no es aplicable para conjuntos con un elevado número de puntos.

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 (por ejemplo, la industria aeroespacial).

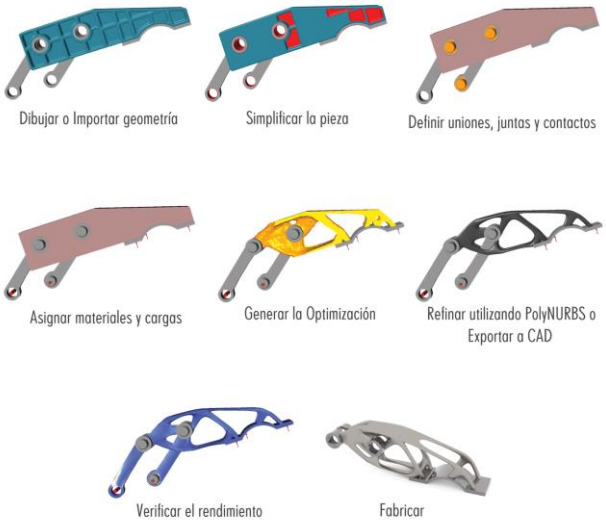
Gracias a los nuevos métodos computacionales, es posible llevar la optimización a un nivel más complejo de análisis a nivel estático, dinámico, plástico, modal o de impacto, entre otros, los cuales pueden considerarse durante el proceso de optimización.

El desarrollo de esta metodología tiene un amplio campo de aplicación para las tecnologías de fabricación aditiva, como por ejemplo la fabricación SLM (Selective Laser Melting), debido a las grandes posibilidades en términos de diseño (geometrías muy complejas).

En el proceso de optimización topológica, se deben de tener en cuenta varios aspectos; el espacio de diseño, el o los casos de carga que va a sufrir la pieza en cuestión, el material y la tecnología con que se va a realizar su fabricación, la reducción de costes mediante la minimización de soportes y aprovechamiento de la cuba de impresión, en caso de utilizar tecnologías aditivas, y muchos más.

Pasos Optimización Topológica:

1. Dibujar o Importar geometría
2. Simplificar la pieza y definir el espacio de diseño
3. Establecer uniones, juntas y contactos
4. Asignar materiales
5. Definir los casos de carga
6. Generar la optimización
7. Refinar la geometría
8. Exportar a CAD o generar STL
9. Verificar el rendimiento
10. Fabricar



Beneficios de la Optimización Topológica

Reducción de peso y volumen

Gracias a la optimización topológica se podrán obtener piezas mucho más ligeras y menos voluminosas manteniendo las prestaciones originales e incluso mejorándolas.

Maximizar la resistencia

Otra manera de encarar la optimización topológica es maximizar la rigidez, lo cual nos permite realizar nuestra pieza lo más resistente posible dentro del espacio de diseño definido y del material que le hayamos asignado. Sin duda es una gran ventaja, ya que sólo aportamos material donde se refuerza la pieza.

Aprovechamiento de la cuba de fabricación

Dentro de los objetivos que buscamos al realizar la optimización topológica es que el diseño permita introducir el mayor número de unidades posibles, reduciendo el volumen y favoreciendo diseños que sean apilables.

Reducción de costes

Los costes se reducen notablemente tanto por minimizar el material necesario para la fabricación como por la reducción de los tiempos de fabricación.

Minimiza los tiempos de fabricación

Al tener que depositar o sintetizar menos material, en el caso de usar fabricación aditiva la cual potencia las bondades de la optimización topológica, el tiempo de fabricación se reduce sensiblemente. Además nuestro diseños se realizan pensando en minimizar los soportes los cual ahorra todavía más tiempo.

Diseños orgánicos y más atractivos

La optimización topológica se basa en el mismo principio en el que actúa la naturaleza, por lo que los diseños presentan formas orgánicas, las cuales no son sólo óptimas para el funcionamiento sino visualmente mucho más atractivas e innovadoras.

Procedimiento de Programación

1. Crear un nuevo script en Matlab

Para empezar con nuestra práctica, lo primero que se tiene que hacer es abrir MATLAB y crear un nuevo script, en donde codificaremos nuestro código y lo ejecutaremos para obtener el análisis.

2. Codificación

```
function P1E1(nelx,nely,volfrac,penal,rmin);
x(1:20,1:60) = volfrac;
loop = 0;
change = 1.;
while change ¿0.01
loop = loop + 1;
xold = x;
[U]=FE(nelx,nely,x,penal);
```

```
[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)p
enal ∗ Ue′ ∗ KE ∗ Ue;
dc(ely,elx) = −penal*x(ely,elx)(penal − l) ∗ Ue′ ∗ KE ∗ Ue;
end
end
[dc] = check(nelx,nely,rmin,x,dc);
[x] = OC(nelx,nely,x,volfrac,dc);
change = max(max(abs(x-xold)));
disp([' It.: ' sprintf('
' Vol.: ' sprintf('
' ch.: ' sprintf('
colormap(gray); imagesc(-x); axis equal; axis tight; axis
off;pause(1e-6);
end
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;
5
else
```

```
l2 = lmid;
end
end
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
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 = zeros(2*(nely+1)*(nelx+1),1);
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)p
enal * KE;
```

```
end
end
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);
U(freedofs,:) = K(freedofs,freedofs) F(freedofs,:);
U(fixeddofs,:)= 0;
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 ...
6
-1/4+nu/12 -1/8-nu/8 nu/6 1/8-3*nu/8];
KE = E/(1-nu2
) * [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)];
```

3. Guardar el código

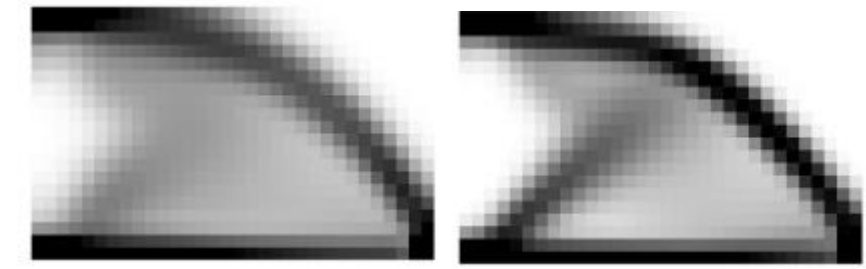
Ahora es necesario guardar nuestro programa con un nombre en específico, asegurándonos que el nombre con el que vayamos a guardar el script y el nombre de la primera función de nuestro código sean iguales.

4. Visualizar el resultado

En este último paso es donde el código es ejecutado y obtenemos los resultados de la práctica, una vez que tenemos las diferentes vistas del análisis podemos concluir con nuestro reporte.

Implementación del Programa

Para realizar la implementación del código en Matlab, usaremos los siguientes valores para los parametros de la funcion Practica1, que son los siguientes. ‘Top(32,20,0.4,3.0,1.2) Ahora escribiremos estos datos en una lınea de comando que se encuentra abajo del código. Ahora solo tendremos que presionar la tecla Enter, y observar el procedimiento. En las siguientes imagenes vemos el inicio y el resultado final.



Se puede apreciar como conforme avancan las iteraciones la pieza se vuelve cada vez más definida.

Conclusiones

Jorge Luis Salinas Garza 1916367

Gracias a la elaboración de esta práctica y la implementación del código de MATLAB pudimos observar las variables que se consideran en este programa, las cuales gracias a ellas es posible la implementación de un código de optimización. Gracias a esta simulación del programa

podemos comprobar los resultados esperados de alguna simulación que estemos realizando y querramos corroborar antes de fabricar la pieza.

Sergio Eduardo Cruz Puente 1903115

Para esta práctica del laboratorio de Biomecánica se tenía como objetivo analizar las distintas secciones que componen un código de optimización topológica, para esto fue necesario implementar el código asignado por el docente en el software MATLAB, en el cual se insertó el código y así poder observar de mejor manera su estructura. El programa consiste en 99 líneas, las primeras 36 líneas son correspondientes al programa principal que se encarga de mandar a llamar las secciones para el correcto funcionamiento del análisis, las líneas 37-48 corresponden a la optimización, las líneas 49-64 es la sección del filtro de malla independiente y las líneas 65-99 corresponden al código de elemento finito.

Julio Emilio Campos Miranda 1917048

En la elaboración de esta práctica, fue interesante como para analizar la pieza se utilizó MATLAB, al ser un código largo, era de suma importancia verificar que todo estuviera bien programado para un buen resultado. El objetivo de esta práctica se cumplió, pues gracias a este reporte considero que pude conocer y entender las secciones que integran el código de optimización topológica, así como la ejecución del análisis.

Christopher Russ Acuña Rdz. 1894698

Durante la realizacion de esta práctica, pude ver como un simple código pudo optimizar de manera topologica una pieza preestablecida, esto lo vuelve muy util al momento de diseñar piezas para soportes de cargas usando el menor material posible.

Cynthia Belen Guerrero Pardo 1926215

Antes de entrar de lleno con la práctica pudimos apreciar el hecho de que la programación es muy importante ya que no es un concepto en específico, ya que se divide en muchas cosas y a partir del marco teórico que nosotros hicimos pudimos ser capaz de hacer el código en MATLAB donde fue que se nos pidió en específico la actividad.

Referencias

- Optimización Topológica | Catec. (s. f.). CATEC. Recuperado 4 de septiembre de 2022, de http://www.catec.aero/es/materiales-y-procesos/l%C3%ADnea-deinvestigaci%C3%B3n/optimizaci%C3%B3ntopol%C3%B3gica#:%7E:text=La%20optimizaci%C3%B3n%20topol%C3%B3gica%20es%20una,funcionalidades%20mec%C3%A1nicas%20del%20componente%20objetivo.
- *Optimización Topológica*. Estudio de Ingeniería y Tecnología Avanzada S.L. (n.d.). Retrieved September 21, 2022, from https://eitaingenieros.com/optimizacion/