ANALISIS COMPARATIVO DE LOS RESULTADOS DE ESFUERZOS SOBRE UN DOMINIO DE SUELO UTILIZANDO EL METODO DE LOS ELEMENTOS FINITOS

1. INTRODUCCION

En este proyecto se abordará un problema 2D de un dominio de suelo elástico, homogéneo e isotrópico, sometido a cargas axiales puntuales y distribuidas en la superficie libre en el cual se quiere construir un túnel y, para esto, se analizará el problema con diferentes tipos de geometría para saber cuál es la más conveniente, es decir, la que menos perturbación genera.

Este problema se abordará mediante el método de elementos finitos utilizando el programa SOLIDS_ISO_GUI.py implementado en la universidad EAFIT por Juan David Gómez, docente de la Escuela de Ingeniería, con el fin de obtener los desplazamientos y deformaciones en cada punto y a partir de ellos realizar el análisis de esfuerzos.

Se dará una breve explicación de los pasos a seguir para la utilización de este programa, y de algunas implementaciones que se le hicieron para optimizar su utilidad. Se incluirán las variaciones del problema presentado con sus respectivos análisis buscando así la geometría óptima.

Además, se mostrarán programas adicionales creados para hacer más eficiente el trabajo.

2. METODOLOGÍA

En nuestro proyecto hicimos uso del programa SOLIDS_ISO_GUI.py que fue implementado en el lenguaje Python el cual funciona a partir de archivos de textos como son:

2.1. *Nodes.txt:* contiene información de los nodos del dominio y sus coordenadas. En la primera columna, el número del nodo, en la segunda y la tercera las coordenadas en X y Y respectivamente, en la cuarta y quinta las reacciones en X y Y respectivamente que se representan con "-1" y con "0" cuando no existen. Anteriormente para ubicar las reacciones se tenía que hacer manualmente, pero nosotros creamos un programa que nos permite de manera más rápida y cómoda ubicar estas reacciones.

2.2. *Eles.txt:* contiene información de los elementos del dominio. En la primera columna se muestra el número de elementos, en la segunda el tipo de elemento (cuadrada, triangular, triangular-6nodos), en la tercera

columna se presenta el tipo de material y en las demás columnas los nodos que conforman el elemento.

2.3. *Loads.txt*: en este archivo se da la información de las cargas que se le aplicaran al dominio. En la primera columna se muestra el numero del nodo donde va aplicada la carga, en la segunda y tercera columna se muestra la dirección y magnitud de la carga en X y Y respectivamente (si es una sola carga es necesario crear dos filas con otro nodo con cargas nulas). Para este se implementó un programa que sirve para crear el archivo loads.txt directamente.

2.4. *Mater.txt:* este documento contiene información relacionada con las propiedades del material del dominio. En la primera columna se muestra el módulo de Young, en la segunda el coeficiente de Poisson. Este archivo se crea usando un programa desarrollado en Python que sirve para uno o dos materiales los cuales en nuestro proyecto serán el suelo y el túnel.

Es necesario aclarar que el archivo de nodes.txt y eles.txt son creados a partir de la malla hecha en el programa de mallado GMSH y posteriormente modificados por *mesher.for* (archivo fortran) para hacerlos compatibles con Python.

3. EJECUCION DEL PROGRAMA

Primero se realizó la malla del dominio discretizado en GMSH con los diferentes casos (sin túnel, túnel cuadrado, túnel circular, túnel semi-circular, túnel doble calzada circular).

Como se mencionó anteriormente, los documentos .msh creados por GMSH son compilados por el archivo mesher.for y se obtienen los archivos nodes.txt y eles.txt, el primero, mediante el programa (…), se modifica como se explicó en el numeral 2.1, este programa además genera los archivos mater.txt y loads.txt indispensables para la ejecución del programa principal.

A continuación, se muestran los programas creados para dicha función.

```
 7
 8 import numpy as np
 9 H=int(input('ingrese longitud del suelo: '))
10 a=np.loadtxt("n.txt")
11 Nelem=len(a[:,1])
12 Cx= np.zeros(Nelem)
13 Cy= np.zeros(Nelem)
14
15 for i in range(Nelem):
16     Cx[i]=a[i,1]
17     Cy[i]= a[i,2]
18
19 print "1: Carga Distribuida"
20 print "2: Carga Puntual"
21 d= int (input('Tipo de Carga:'))
22 w= int(input ('Ingrese magnitud de la carga: '))
23
24
25 ## REACCIONES
26 Rx=np.zeros(Nelem)
27 Ry=np.zeros(Nelem)
28 m=0
29 M=0
30
31 #print 'Reacciones x'
32 while M+1 <=Nelem:
33     if Cx[M]==H or Cx[M]==0:
34         i=-1
35
36     else:
37         i=0
38
39     Rx[m]=i
40     m=m+1
41     M=M+1
42 #print Rx
43
44
45 #print 'Reacciones y'
46 M=0
47 m=0
48 while M+1<=Nelem:
49     if Cy[M]==0:
50         i=-1
51     else:
52         i=0
53
54     Ry[m]=i
55     m=m+1
56     M=M+1
57 #print Ry
58
59 #CARGAS
60 CargaX=np.zeros(Nelem)
61 CargaY=np.zeros (Nelem)
62 Nodo= np.zeros(Nelem)
63
64
65 if d == 2:
66     for i in range (Nelem):
67
68         if Cx[i]==H/2 and Cy[i]==H/2:
69             CargaY[i]=-w
70         else:
71             CargaY [i]=0
72             #print CargaY
73
74         Nodo[i]=i
75         #print Nodo
```

```
76
77
78 if d==1:
79     i=0
80     for i in range (Nelem):
81
82         if Cy[i]==H/2:
83             CargaY[i]= -w
84
85         else:
86             CargaY[i]=0
87             #print CargaY
88
89         Nodo[i]= i
90         #print Nodo
91
92 #Matriz nodes
93 i=0
94 j=0
95 M= np.zeros([Nelem,5])
96 V= np.zeros((5*Nelem))
113
114 C= np.zeros([Nelem,3])
115 V1= np.zeros((3*Nelem))
116 V1[0:Nelem]=Nodo[0:Nelem]
117 V1[Nelem:(2*Nelem)]= CargaX[0:Nelem]
118 V1[(2*Nelem):(3*Nelem)]=CargaY[0:Nelem]
119 m=0
120 for j in range (3):
121     for i in range(Nelem):
122         C[i,j]= V1[m]
123         m=m+1
124 np.savetxt("Cargas",C,'%3i %3i %3i')
125
126
127
```

Imagen 01: Programa que modifica el archivo nodes.txt y crea el archivo loads.txt

```
5 import numpy as np
6
7 N=int(input('Numero de Materiales:'))
8 if N==2:
9     #print ('TENGA EN CUENTA --> Solido:1 ; Suelo:2')
10    E1=int(input('Modulo de Young Material 1:'))
11    print('TENGA EN CUENTA --> Coeficiente de Poisson entre 0.0 y 0.5')
12    v1=float(input('Coeficiente de Poisson Material 1:'))
13    if v1>=0 and v1<=1:
14       E2=int(input('Modulo de Young Material 2:'))
15       v2=float(input('Coeficiente de Poisson Material 2:'))
16       if v2>=0 and v2<=1:
17         EV1=(E1/(1-(v1)**2))
18         EV2=(E2/(1-(v2)**2))
19         vv1=(v1/(1-v1))
20         vv2=(v2/(1-v2))
21         print (EV1,vv1)
22         print (EV2,vv2)
23         M=np.zeros([2,2])
24         A=[EV1,EV2,vv1,vv2]
25         m=0
26         for j in range (2):
27           for i in range (2):
28               M[i,j]=A[m]
29               m=m+1
30         np.savetxt('mater',M,'%8.3f %8.3f')
31       else:
32           print('Coeficiente de Poisson debe estar entre 0 y 0.5')
33    else:
34        print('Coeficiente de Poisson debe estar entre 0 y 0.5')
35
```

```
else:
    #print ('TENGA EN CUENTA --> Solido:1 ; Suelo:2')
    E1=int(input('Modulo de Young Material:'))
    print('TENGA EN CUENTA --> Coeficiente de Poisson entre 0.0 y 0.5')
    v1=float(input('Coeficiente de Poisson Material:'))
    if v1>=0 and v1<=1:
        EV1=(E1/(1-(v1)**2))
        vv1=(v1/(1-v1))
        print (EV1,vv1)
        print (EV1,vv1)
        N=np.zeros([2,2])
        B=[EV1,EV1,vv1,vv1]
        n=0
        for j in range (2):
            for i in range (2):
                N[i,j]=B[n]
                n=n+1
        np.savetxt('mater',N,'%8.3f %8.3f')
    else:
        print('Coeficiente de Poisson debe estar entre 0 y 0.5')

```

Imagen 02: Programa que crea el archivo mater.txt

El programa definitivo.py debe encontrarse en la misma carpeta donde este el archivo node.txt *sin modificar*, este programa también creará los archivos mater.txt y loads.txt en la misma carpeta.

En el momento de ejecutar el programa SOLIDS_ISO_GUI.py, se abre una nueva ventana para seleccionar la carpeta que contenga los cuatro archivos de texto.

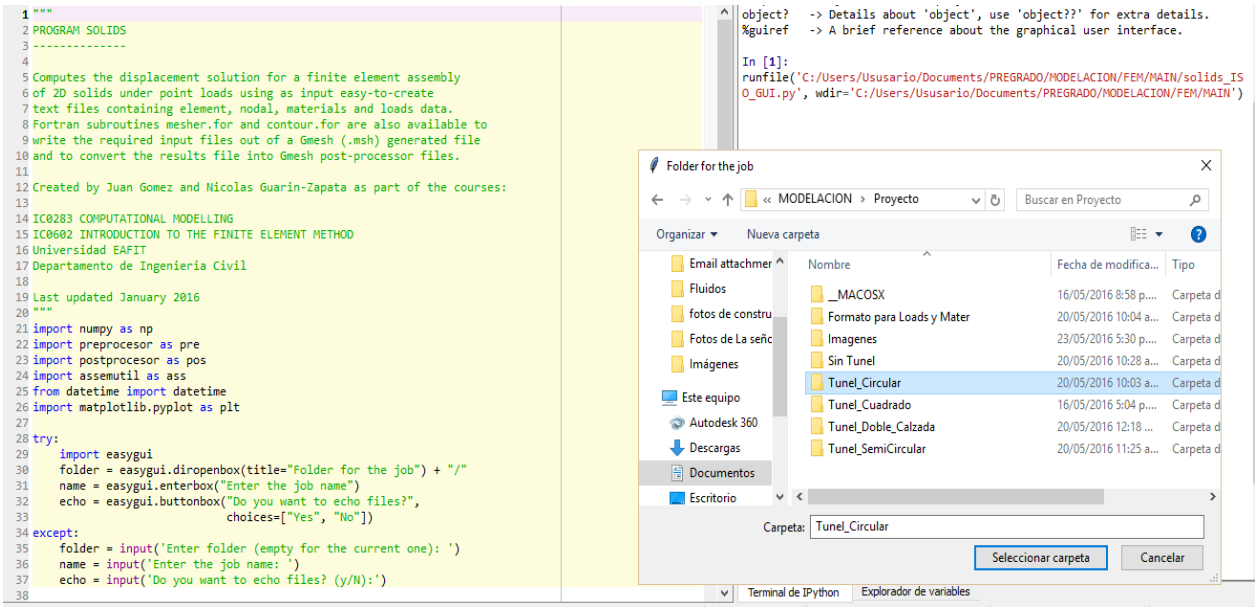


Imagen 03: Ejecución del programa principal SOLIDS_ISO_GUI.py y selección de la carpeta con los cuatro archivos .txt

Posteriormente se debe ingresar un nombre cualquiera y ejecutar el programa. Dependiendo del tamaño del problema es posible que el programa tarde en mostrar los resultados.

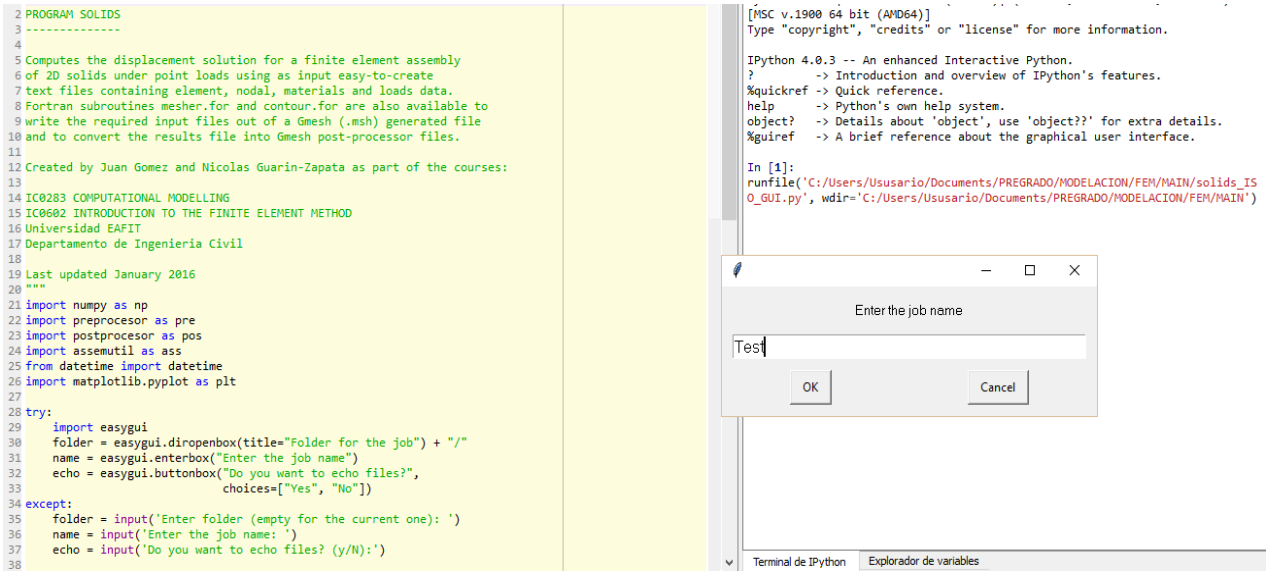
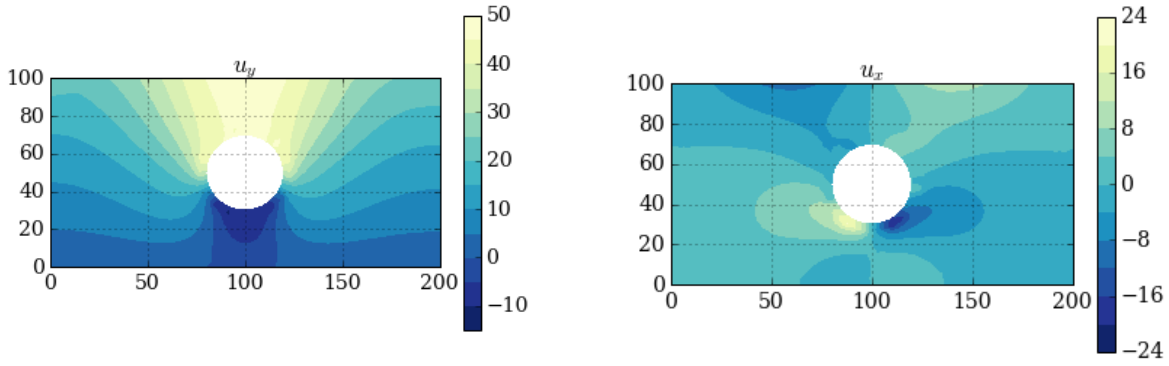


Imagen 04: Ingreso del nombre del trabajo

4. RESULTADOS

A continuación, se muestran los resultados obtenidos con el uso del programa principal y los programas creados por nosotros. En este caso se muestra un ejemplo de un dominio de suelo que contiene un túnel circular y está sometido a una carga distribuida en las coordenadas $(x, 100)$ $con\ x \in (0\ , 200)$



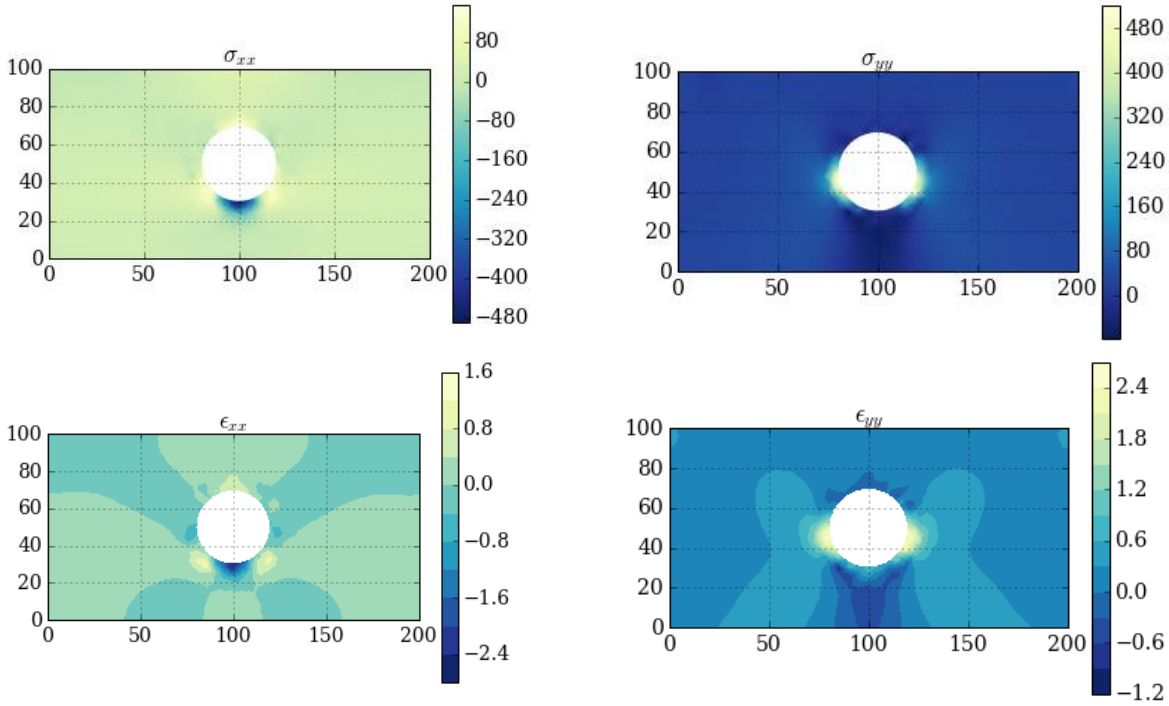


Imagen 05: resultados para un dominio de suelo con un túnel circular y carga distribuida sometida en coordenadas $(x, 100)$ *para* $x \in (0\,, 200)$. *Se muestran los desplazamientos, deformaciones y esfuerzos.*

Los resultados muestran cómo se comporta el suelo frente a una carga distribuida constante de 100 N/m.

5. ANALISIS DE RESULTADOS

Con estos resultados obtenidos podemos ser capaces de analizar los esfuerzos en ciertos puntos del dominio. Pero preferiblemente nos concentraremos en los puntos localizados en las fronteras del suelo con las paredes del túnel.

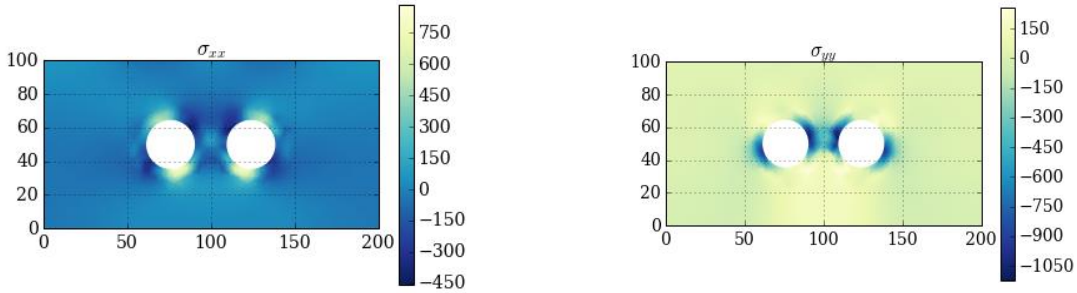


Imagen 06: representación gráfica de los esfuerzos sobre un dominio con dos túneles circulares. Se pueden observar las perturbaciones en las paredes de estos.

Nosotros escogimos las paredes del túnel debido a que ahí se presentan las mayores perturbaciones de esfuerzos en el suelo y, por lo tanto, donde se encuentran los esfuerzos más grandes como se puede observar en la *Imagen 06.*

Es intuitivo mencionar que estas perturbaciones pueden variar en magnitud si se escoge una geometría para el túnel distinta. Un túnel cuadrado puede presentar perturbaciones mayores que las que puede presentar un túnel de geometría menos "fuerte" como un circulo. En la siguiente imagen se ve esta situación si comparamos un suelo con túnel cuadrado, con un suelo sin perturbar.

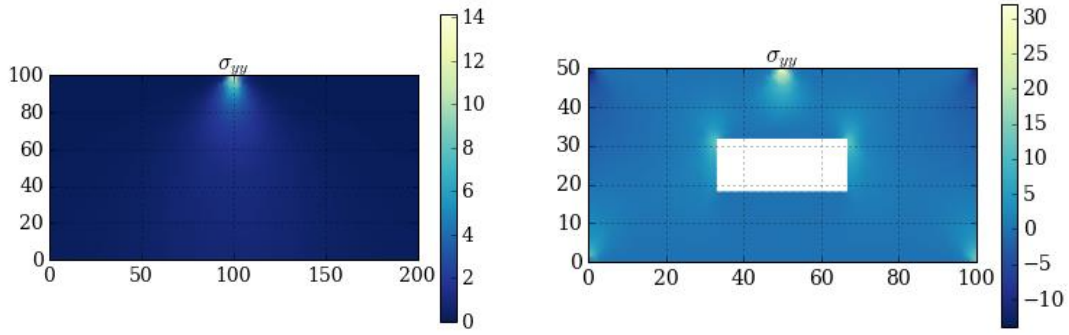


Imagen 07: esfuerzos en Y (σ_{yy}) sobre un suelo sin túnel (izq.) y un suelo con túnel cuadrado (der.). Se pueden observar las perturbaciones ocasionadas una vez puesto un túnel en el dominio. La carga es puntual en dirección –y ubicada en (100,100)

Se asume una resistencia a la compresión del material del túnel a partir del cual nos basaremos para hallar cuáles son las geometrías de túneles que menos afectan el suelo.

6. REFERENCIAS

- Azizi, F. (1999). *Applied Analyses in Geotechnics.* CRC Press.
- Budhu, M. (s.f.). *Soils Mechanics and Foundations 3ra Edicion.*
- Gomez, J. D. (2016). *solids ISO.py: Programa para Analisis de Sólidos Elásticos en Python por el Método de los Elementos Finitos.* Medellin.
- Gomez, J., & Guarin, N. (2016). *Finite Element Analysis Code solids ISO.py.* Medellin.
- MANSFIELD, E. H., & M.A. (1955). *Neutral Holes in Plane Sheet: Reinforced Holes which are Elastically Equivalent to the Uncut Sheet.* Londres.