

solids_ISO.py: Programa para Análisis de Sólidos Elásticos en Python por el Método de los Elementos Finitos

Juan Gomez

21 de enero de 2017

Resumen

Se presenta el programa por elementos finitos **solids_ISO.py**, implementado en el lenguaje de alto nivel Python a manera de ejemplo del reporte del proyecto semestral de la materia IC0285 Modelación Computacional. El proyecto busca incentivar en el estudiante de ingeniería civil de la Universidad EAFIT el uso racional de herramientas computacionales propias o existentes en la solución de problemas complejos durante el ejercicio de la carrera.

Introducción

El método de los elementos finitos (MEF) es una técnica numérica para la solución de problemas de valores en la frontera (PVF) propuesta inicialmente a finales de los 50 y principios de los 60 (Clough & Tocher, 1965; Turner, 2012). Desde entonces, ha experimentado un fuerte desarrollo tecnológico y académico, plasmado en una gran cantidad de paquetes comerciales y de aplicaciones en diferentes áreas de ciencia e ingeniería. Paralelamente, se han escrito una gran cantidad de textos cubriendo aspectos teóricos relativos a la formulación y a la implementación del método (Zienkiewicz & Taylor, 2005; Bathe, 1995a; Hughes, 2012; Reddy, 1993). En la actualidad el método de los elementos finitos puede considerarse como la herramienta de simulación numérica más poderosa y versátil para el estudio de problemas de ingeniería.

Desde el punto de vista teórico el método se basa en la solución de la forma débil del PVF, sobre una discretización o partición del dominio en sub-dominios o elementos finitos en cada uno de los cuales se expresa la solución vía interpolación a partir de valores determinados solo para unos cuantos puntos o nudos. La contribución de cada elemento finito del dominio es posteriormente sumada usando leyes gobernantes entre las diferentes variables del problema. En el caso mecánico la forma débil del PVF corresponde al principio de los desplazamientos virtuales y el ensamblaje o consideración de todos los elementos se construye a partir de condiciones de equilibrio y compatibilidad de desplazamientos.

En ingeniería civil el MEF es aplicable en la simulación de problemas relativos a áreas como mecánica de fluidos, análisis dinámico de estructuras, mecánica de suelos, análisis de presas, o estructuras de contención. Algunos programas comerciales, específicamente desarrollados para uso en ingeniería civil son: Plaxis¹, SAP2000², OpenSees³.

A pesar del gran desarrollo del método y de su amplio uso en oficinas de ingeniería, especialmente en países desarrollados, este no es impartido en los currículos de ingeniería civil en la mayoría de universidades colombianas. En este reporte se presenta un programa por elementos finitos desarrollado en

¹<http://www.plaxis.nl/plaxis2d/>

²<https://www.csiamerica.com/products/sap2000>

³http://opensees.berkeley.edu/wiki/index.php/Main_Page

el lenguaje de alto nivel Python y dirigido al análisis de sólidos elásticos en condiciones de tensión plana y/o deformación plana. El programa ha sido desarrollado principalmente con fines educativos en los cursos IC0285 Modelación Computacional y IC0602 Introducción al Método de los Elementos Finitos de la Universidad EAFIT. El código es de libre distribución⁴. Paquetes similares, desarrollados en Python y dirigidos a uso educativo pueden identificarse en las herramientas SfePy (Cimrman, 2014), PyFEM (De Borst et al., 2012) and FEniCS (Logg et al., 2012).

En el reporte se describe brevemente la estructura del programa, sin ahondar en aspectos teóricos del método como tal. Posteriormente se presentan las soluciones de problemas simples, comunes en ingeniería civil, y obtenidas a través del código. En la última parte del reporte se presentan las conclusiones y se indican algunos puntos por mejorar en futuras versiones de la herramienta de simulación.

Metodología

El programa para análisis de sólidos elásticos en 2-dimensiones `solids_ISO.py` esta implementado en el lenguaje de alto nivel Python y en particular en los recursos computacionales disponibles en los módulos `numpy`, `sympy`, `matplotlib` y `easygui` para determinar la distribución de tensiones, desplazamientos y deformaciones (infinitesimales) de la versión discretizada de un sólido elástico plano sometido a tracciones externas. Por ejemplo, la figura 1 muestra un dominio correspondiente a 1/4 de una sección transversal circular de una tubería sometida a una presión interna. En este caso el dominio de solución ha sido dividido en elementos finitos de geometría triangular.

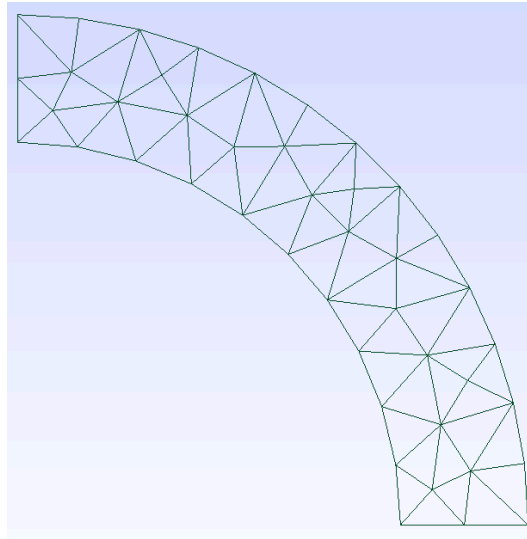


Figura 1: 1/4 de sección transversal de una tubería bajo presión interna discretizada en elementos finitos triangulares.

En el caso de problemas de determinación de la respuesta mecánica de sólidos elásticos, el MEF plantea ecuaciones de equilibrio de la forma:

$$F = Ku$$

para cada elemento finito y posteriormente acopla las ecuaciones resultantes usando condiciones de compatibilidad de desplazamientos y equilibrio de fuerzas entre las caras de los elementos adyacentes para

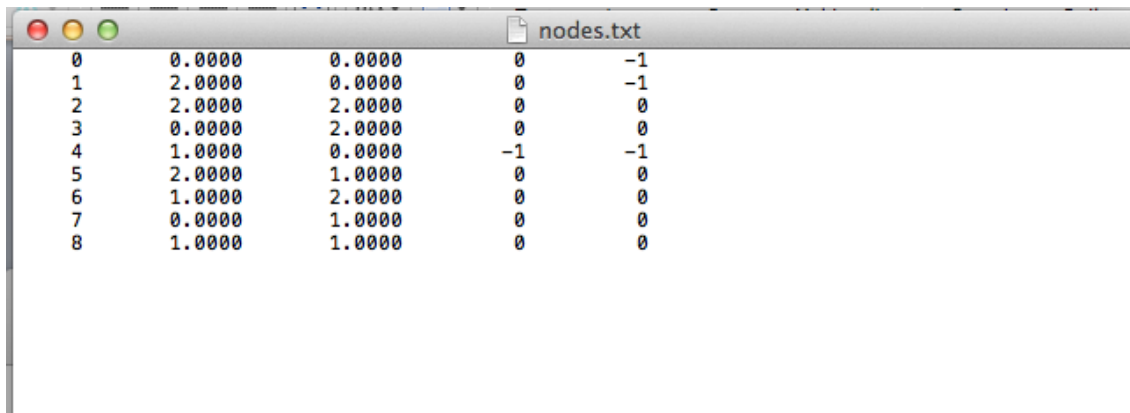
⁴https://github.com/jgomezc1/FEM_PYTHON

formar un sistema global de ecuaciones de la forma:

$$\{F\} = [K] \{u\} \quad (1)$$

en la cual $\{F\}$ es el vector de fuerzas externas aplicadas a la estructura, $\{u\}$ es el vector de desplazamientos (o grados de libertad) de los puntos que conforman cada elemento y $[K]$ es una matriz que contiene constantes de resorte resultante del ensamblaje o consideración de todos los elementos del modelo. Una vez se determinan los desplazamientos o grados de libertad desconocidos tras resolver el sistema de la eq. (1) estos pueden ser usados para determinar las tensiones y deformaciones al interior de cada uno de los elementos del modelo.

En el programa `solids.ISO.py` el modelo se define por medio de archivos de texto que contienen toda la información relativa a la definición de los elementos, materiales, cargas y condiciones de frontera del modelo. En el sitio de descarga del programa⁵ se encuentra un manual del usuario ilustrando de manera detallada el proceso de definición de un modelo. Por ejemplo, la figura 2 muestra una captura de pantalla del archivo con la información de los nodos de una discretización en el cual se indican las coordenadas de cada uno de los puntos del dominio y las condiciones de frontera asociadas a cada nudo. La malla o discretización (ver figura 1) puede crearse de manera alternativa con el paquete externo `gmsh`⁶.



Node	x	y	ux	uy
0	0.0000	0.0000	0	-1
1	2.0000	0.0000	0	-1
2	2.0000	2.0000	0	0
3	0.0000	2.0000	0	0
4	1.0000	0.0000	-1	-1
5	2.0000	1.0000	0	0
6	1.0000	2.0000	0	0
7	0.0000	1.0000	0	0
8	1.0000	1.0000	0	0

Figura 2: Archivo de datos `nodes.txt` almacenando la información correspondiente a la localización de los nudos y sus condiciones de frontera.

La figura 3 muestra los tipos de elementos disponibles en el programa. El primer caso corresponde a un cuadrilátero de 4 nudos (puntos negros en las esquinas), mientras que los 2 restantes corresponden a elementos triangulares de 3 y 6 nudos respectivamente. El número de nudos en un elemento define el orden de los polinomios de interpolación a usar en el elemento. Por ejemplo en el caso del cuadrado de 4 nudos y el triángulo de 3 nudos el esquema de interpolación es lineal, mientras que en el triángulo de 6 nudos el esquema es cuadrático.

⁵https://github.com/jgomezc1/FEM_PYTHON

⁶<http://gmsh.info/>

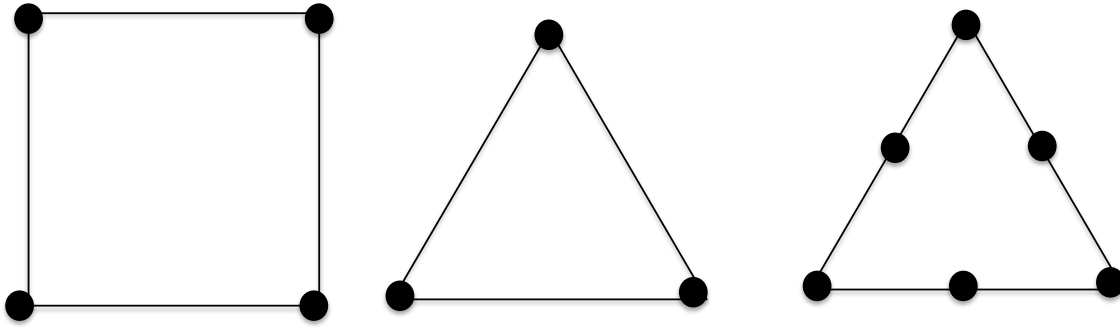


Figura 3: Elementos disponibles en el programa.

Un análisis completo de un sólido por el MEF comprende 3 pasos básicos: (i) creación y lectura del modelo o pre-procesamiento (ii) solución del sistema de ecuaciones y (iii) cálculo y visualización de resultados o post-procesamiento. En el caso de sólidos.ISO.py los pasos (i) y (iii) pueden desarrollarse con la ayuda del programa externo **gmsh** y rutinas adicionales escritas en el lenguaje de bajo nivel Fortran, las cuales hacen operaciones de lecto-escritura entre los archivos de texto generados por **gmsh** y los requeridos por **sólidos.ISO.py**. Estas rutinas se pueden obtener del sitio de descarga del programa. El paso correspondiente a la solución del sistema de ecuaciones se realiza usando la librería **linalg** disponible en el modulo **numpy**. En particular el software utiliza un algoritmo de factorización de la matriz de rigidez.

Ejecución del programa

Una vez se han creado los archivos de texto almacenando el modelo y estos han sido cargados en la carpeta de instalación del programa el usuario se encuentra listo para ejecutar el mismo. Procediendo desde la ventana principal mostrada en la figura 4, será necesario indicar el nombre del problema a analizar y la localización de los archivos de datos.

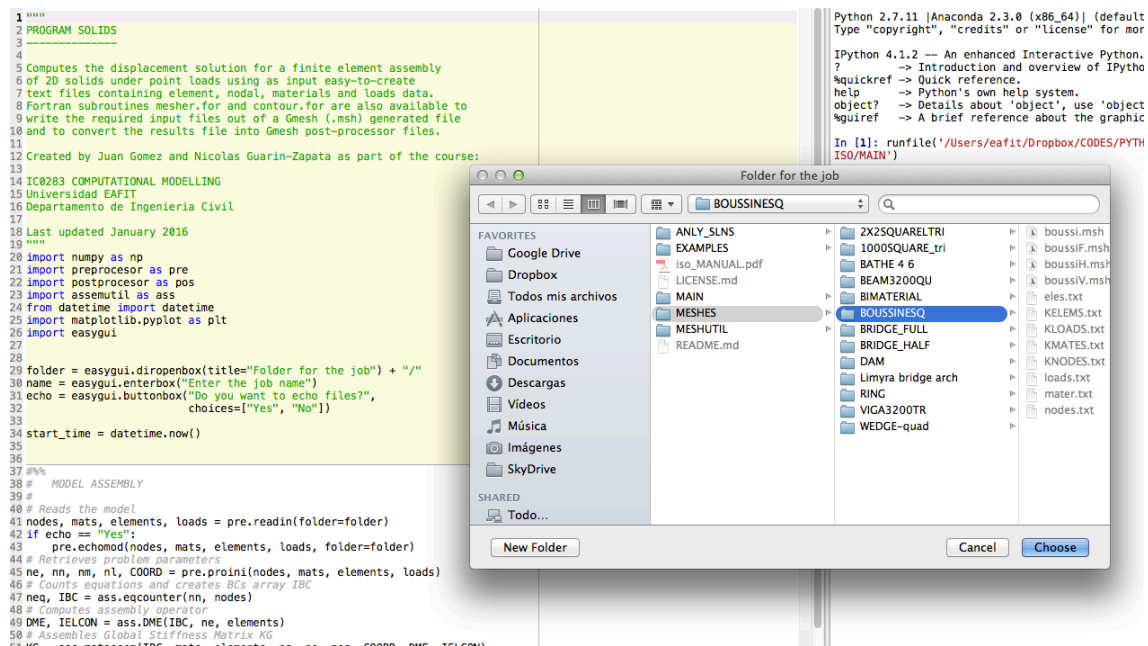


Figura 4: Ejecución inicial del código.

Una vez el código es ejecutado este entrega visualizaciones de los resultados (componentes de los desplazamientos, tensiones, deformaciones y valores principales) en términos de mapas de calor como los que se presentan en la figura 5. Adicionalmente, los resultados también se encuentran disponibles en memoria en diversos arreglos de manera que el usuario puede realizar procesos adicionales usando las diferentes funciones disponibles en Python como se muestra en la figura 6. Paralelamente el programa también genera archivos de texto con datos para realizar la visualización de resultados con `gms`.

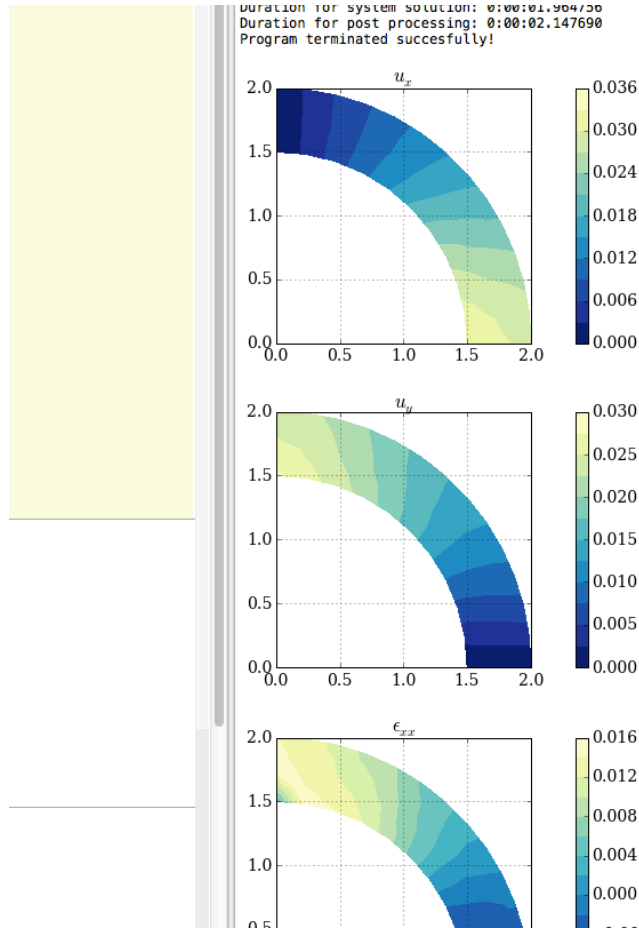


Figura 5: Salida de resultados.

```

In [4]: print UG
[ 4.26710943e-01 -4.26710943e-01 -7.71556614e-01 3.68175442e+00
 7.71556614e-01 3.68175442e+00 -3.00866221e-01 1.55446372e+00
-1.21257058e-15 2.31824558e+00 3.00866221e-01 1.55446372e+00
-1.51402545e-16 1.39380943e+00]

In [5]: print EG
[[-0.32746034 1.52051347 -0.22595465]
 [-0.32746034 1.42775968 -0.15329816]
 [-0.40011682 1.52051347 -0.13320085]
 [-0.40011682 1.42775968 -0.06054437]
 [-0.32746034 1.42775968 0.15329816]
 [-0.32746034 1.52051347 0.22595465]
 [-0.40011682 1.42775968 0.06054437]
 [-0.40011682 1.52051347 0.13320085]
 [-0.67208803 1.87309762 -0.73809395]
 [-0.67208803 1.17862923 -1.00984718]
 [-0.40033481 1.87309762 -0.04362556]
 [-0.40033481 1.17862923 -0.31537878]
 [-0.67208803 1.17862923 1.00984718]
 [-0.67208803 1.87309762 0.73809395]
 [-0.40033481 1.17862923 0.31537878]
 [-0.40033481 1.87309762 0.04362556]]

In [6]: |
```

Figura 6: Resultados disponibles en memoria para post-procesamiento adicional.

Resultados

Para verificar la implementación se compararon los resultados con soluciones analíticas disponibles en la literatura y correspondientes a los problemas de (i) una viga en voladizo ([Timoshenko & Goodier, 1970](#)), (ii) cuña sometida a cortante ([Gomez & Sierra, 2015](#)) y (iii) placa bajo tensiones en la frontera ([Bathe, 1995b](#)). Las soluciones a estos problemas no se incluyen en este reporte. Para mostrar la capacidad del código con problemas mas exigentes desde el punto de vista computacional, se determinó (i) la respuesta de una presa de concreto soportada sobre un semi-espacio y sometida a la presión hidro-estática de un fluido y (ii) la respuesta de un semi-espacio a una carga puntual en la superficie libre. Este último problema corresponde a la denominada solución de Boussinesq ([Timoshenko & Goodier, 1970](#)).

Presa de concreto sometida a presión hidro-estatica

Considere la presa de concreto mostrada en la figura 7. Esta se encuentra soportada sobre un deposito de suelo homogéneo, elástico e isotrópico. La presa se encuentra sometida a la presión ejercida por un fluido de peso específico γ sobre la pared vertical. La presa tiene una altura desde la superficie libre del depósito de suelo de 150 m y el ancho de la pata es de 50 m. El depósito de suelo se considera de 100×100 m. Este último se encuentra restringido en la dirección horizontal sobre las paredes laterales, y en la dirección vertical sobre la superficie del fondo. Para la presa se asumió un modulo de elasticidad $E = 2,0$ y una relación de Poisson $\nu = 0,3$, mientras que para el suelo se tomaron los valores de $E = 1,0$ y $\nu = 0,3$.

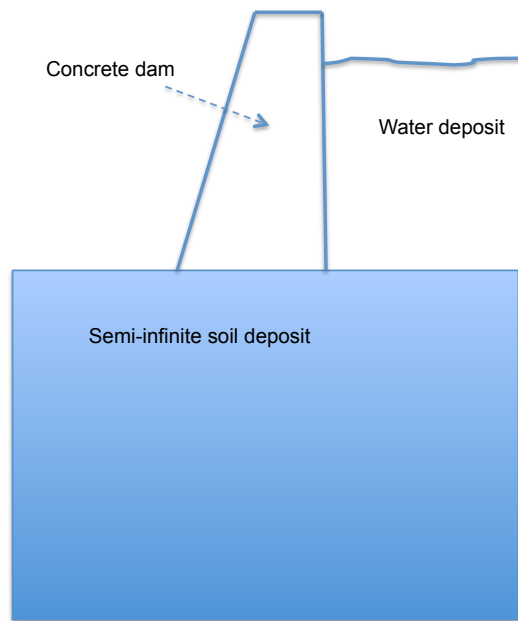


Figura 7: Presa de concreto bajo presión hidro-estática.

El modelo se discretizó en 2735 elementos triangulares lineales conformados por 1446 puntos nodales, generando un sistema de aproximadamente 2800 ecuaciones. La malla se creó con el programa externo **gmsh** (ver figura 8).

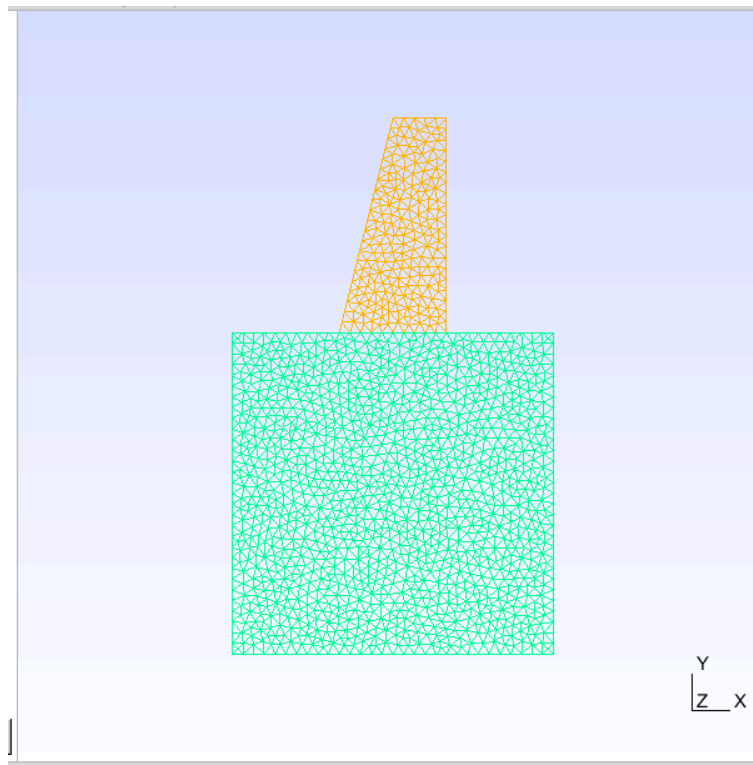


Figura 8: Malla por elementos finitos creada con el paquete externo **gmsh**.

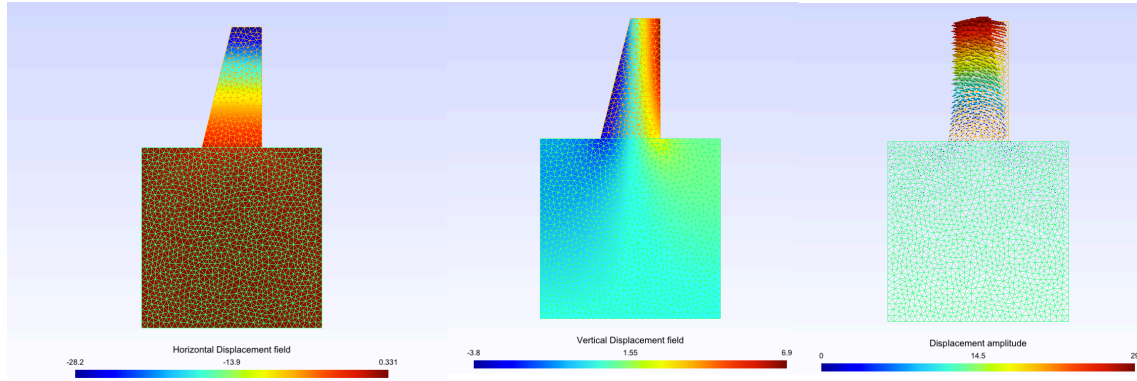


Figura 9: Campo de desplazamientos visualizado con **gms**.

La figura 9 muestra contornos (mapa de color) para el campo de desplazamientos en la presa y en el depósito de suelo. Como es natural esperar, el desplazamiento horizontal (parte a) en la presa disminuye de manera uniforme desde un valor máximo en la cresta hasta un valor mínimo en la interfaz con el depósito de suelo. En cuanto a la componente vertical del desplazamiento (parte b) esta muestra una distribución relativamente simétrica con respecto a una superficie vertical pasando por la cresta. Claramente, se observa un desplazamiento negativo (con esfuerzos de compresión) en la parte de la presa a la izquierda de esta superficie, mientras que la parte derecha de la presa experimenta levantamientos indicados por valores positivos de los desplazamientos horizontales. La configuración deformada de la presa se muestra en la parte c de la figura. La figura 10 muestra las distribuciones de tensiones a cortante, además de las tensiones principales sobre la presa y el suelo. Claramente, las tensiones cortantes experimentan concentraciones sobre la pata de la presa y en las esquinas sobre ambas paredes laterales. De otro lado las tensiones principales experimentan un cambio de signo, como podía anticiparse de la solución por desplazamientos, sobre una superficie vertical que cruza aproximadamente la mitad del depósito de suelo. Estas zonas de cambio de signo, marcando regiones sometidas a tracción y a compresión deben considerarse durante el diseño de la presa.

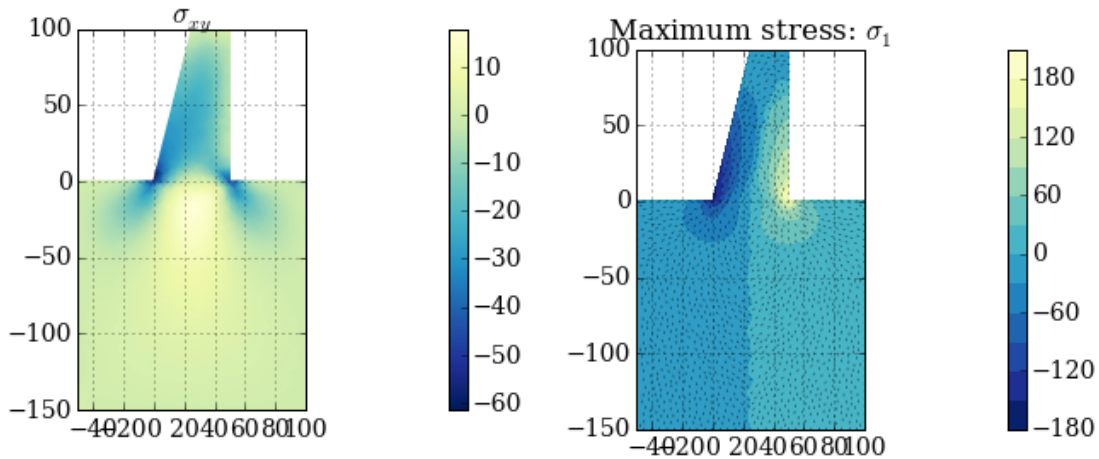


Figura 10: Contornos de tensiones cortantes y tensiones principales.

Problema de Boussinesq

Se determinó la respuesta de un semi-espacio ante una distribución de cargas uniforme de resultante total 50 unidades de fuerza. Para esto se creó un modelo del semiespacio conformado por una malla de 3200 elementos triangulares cuadráticos conformados por 6500 nudos para un sistema de aproximadamente 13000 ecuaciones. El semi-espacio se supuso de $E = 1,0$ y $\nu = 0,3$. La figura 11 muestra el campo vectorial de desplazamientos , así como la configuración deformada del semi-espacio.

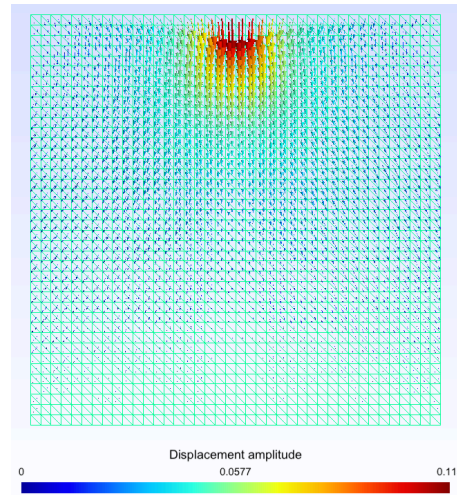


Figura 11: Campo vectorial de desplazamientos para el problema de Boussinesq.

Similarmente, la figura 12 muestra las componentes horizontal (izquierda) y vertical (derecha) para el depósito de suelo. La componente horizontal es simétrica con respecto a la línea de aplicación de la carga experimentando desplazamientos negativos en la parte izquierda del dominio y positivos sobre la parte derecha, mientras que los desplazamientos verticales son negativos (compresión) con valores nulos en las fronteras del semi-espacio. Considerando que el modelo usa un semi-espacio truncado por razones computacionales, la solución debe estudiarse con cuidado pues los resultados aún pueden estar afectados por las condiciones de frontera asumidas.

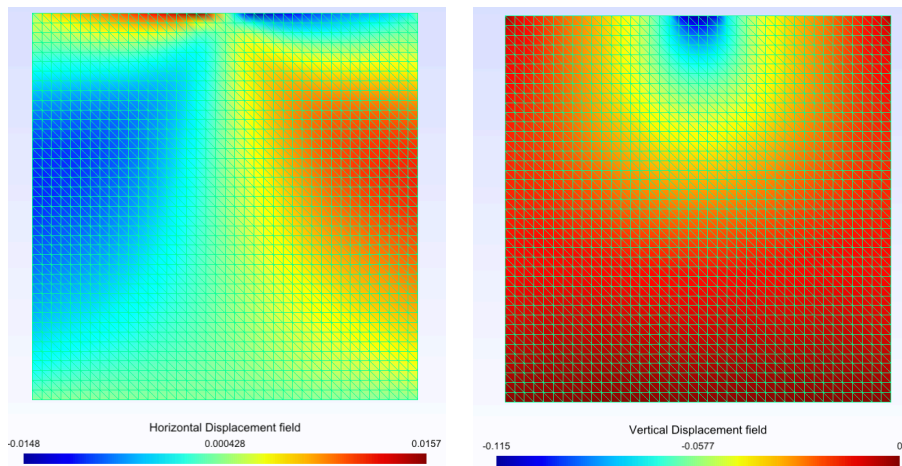


Figura 12: Componentes horizontal y vertical del campo de desplazamientos para el problema de Boussinesq.

Conclusiones

A manera de ejemplo de reporte del proyecto semestral de la materia Modelación Computacional ofrecida en el programa de Ingeniería Civil de la Universidad EAFIT, se reporta sobre la implementación del programa por elementos finitos para elasticidad plana en el lenguaje de alto nivel Python **solids_ISO.py**. El programa, desarrollado con fines educativos, determina la distribución de desplazamientos, tensiones y deformaciones en un sólido plano de geometría arbitraria y sometido a tracciones. La discretización o partición del dominio del problema permite usar elementos finitos cuadriláteros lineales, triángulos lineales y triángulos cuadráticos. El pre-procesado y post-procesado puede hacerse con el programa externo **gmsh**. Para mostrar la capacidad del programa se determinó la respuesta de una presa de concreto soportado sobre un depósito de suelo y la respuesta de un depósito de suelo a una distribución de cargas en la superficie libre (problema de Boussinesq). Para ambos problemas se obtuvieron resultados consistentes con la teoría.

Para versiones futuras del programa se planea introducir la posibilidad de determinar la respuesta ante excitaciones dinámicas, cargas gravitacionales además de expandir la biblioteca de elementos.

Finalmente se concluye que el lenguaje de programación de alto nivel Python, puede usarse de manera eficiente para abordar una amplia gama de problemas de ingeniería.

Referencias

- Bathe, K.-J., 1995a. *Finite Element Procedures*, Prentice Hall, 2nd edn.
- Bathe, K.-J., 1995b. *Finite Element Procedures*, Prentice Hall, 2nd edn.
- Cimrman, R., 2014. Sfepy-write your own fe application, *arXiv preprint arXiv:1404.6391*.
- Clough, R. W. & Tocher, J. L., 1965. Finite element stiffness matrices for analysis of plates in bending, in *Proceedings of conference on matrix methods in structural analysis*, pp. 515–545.
- De Borst, R., Crisfield, M. A., Remmers, J. J., & Verhoosel, C. V., 2012. *Nonlinear finite element analysis of solids and structures*, John Wiley & Sons.
- Gomez, J. & Sierra, C., 2015. Notas de clase: Mecánica de los medios continuos. universidad eafit.
- Hughes, T. J., 2012. *The finite element method: linear static and dynamic finite element analysis*, Courier Corporation.
- Logg, A., Mardal, K.-A., & Wells, G., 2012. *Automated solution of differential equations by the finite element method: The FEniCS book*, vol. 84, Springer Science & Business Media.
- Reddy, J. N., 1993. *An introduction to the finite element method*, vol. 2, McGraw-Hill New York.
- Timoshenko, S. & Goodier, J., 1970. *Theory of Elasticity*, McGraw-Hill, 3rd edn.
- Turner, M. J., 2012. Stiffness and deflection analysis of complex structures, *journal of the Aeronautical Sciences*.
- Zienkiewicz, O. & Taylor, R., 2005. *The Finite Element Method for Solid and Structural Mechanics*, vol. 2, Butterworth-Heinemann, 6th edn.