

Aplicación del Método de Elementos Finitos en Mecánica de Sólidos para Problemas Unidimensionales y Bidimensionales

Martín Achondo Mercado, Catalina Santibañez Mercado

Universidad Técnica Federico Santa María, Departamento de Ingeniería Mecánica, Valparaíso, Chile, martin.achondo@sansano.usm.cl, catalina.santibanez@sansano.usm.cl

Abstract

En el presente documento se abordan tres problemas de mecánica de sólidos en 1D y 2D mediante el método de elementos finitos (MEF), utilizando MATLAB. Los problemas incluyen el análisis de una armadura bidimensional sometida a carga, un sólido de revolución bajo carga compresiva, y una viga circular en tensión plana. El objetivo principal es programar códigos de MEF para obtener desplazamientos, esfuerzos y deformaciones, comparándolos con soluciones teóricas. Respecto a los resultados obtenidos, se destaca la eficacia del MEF para simplificar cálculos y mejorar la precisión. Para la armadura, se logró obtener que el desplazamiento del nodo libre llega a 0.194 [mm]. Para el caso del sólido en compresión, se demostró la independencia de la malla, alcanzando una estabilidad a los 50 nodos. Para este mismo problema se logró alcanzar errores de 2E-12 y 3E-11 para el desplazamiento y esfuerzo respectivamente y un desplazamiento máximo de 0.016 [mm] aproximadamente. Por último, para la viga circular en tensión plana, se obtuvo un desplazamiento máximo de 14.43 con un esfuerzo de Von Mises máximo de 7.02. Pese a que en este último caso no se obtuvo una completa independencia de la malla, los resultados coincidieron con las predicciones teóricas. En general se concluye que el trabajo demuestra la aplicabilidad del MEF a problemas de mecánica de sólidos, resaltando consideraciones sobre la elección de la malla y la eficiencia computacional.

Palabras claves

Métodos de elementos finitos, Formulación débil de Galerkin, Mecánica de sólidos.

Índice

1. Introducción	4
1.1. Objetivo General	4
1.2. Objetivos Específicos	4
1.3. Problema N°1	5
1.4. Problema N°2	5
1.5. Problema N°3	6
1.6. Problema N°4	7
2. Metodología	10
2.1. Problema Elástico	10
2.1.1. Función de Esfuerzos de Airy	12
2.1.2. Formualción Débil de Galerkin	12
2.2. Método de Elementos Finitos	13
2.2.1. Discretización en 1D	14
2.2.2. Discretización 2D	17
2.3. Metodología aplicado a cada problema	20
2.4. Problema N°1 y N°2	20
2.5. Problema N°3	25
2.5.1. Desarrollo teórico	25
2.5.2. Formulación de MEF	27
2.6. Problema N°4	29
2.6.1. Desarrollo teórico	29
2.6.2. Formulación de MEF	29
3. Resultados	32
3.1. Problema N°1 y N°2	32
3.2. Problema N°3	35
3.3. Problema N°4	40

4. Análisis	49
4.1. Problema N°1 y N°2	49
4.2. Problema N°3	49
4.3. Problema N°4	51
5. Conclusiones	53
6. Anexos	55
6.1. Problema N°1 y N°2	55
6.2. Problema N°3	59
6.3. Problema N°4	63

1. Introducción

En el presente trabajo se exponen tres problemas de mecánica de sólidos en 1D y 2D, que serán abordados a través del método de elementos finitos. Para el primer y segundo problema, se trabaja con una armadura bidimensional. El objetivo se centra en la determinación de la matriz de rigidez global de los elementos, la cual guarda relación con las coordenadas locales mediante la matriz de transformación a obtener. Estos conceptos se aplican en una armadura que sustenta una carga en su parte inferior, con el objetivo de calcular desplazamientos, deformaciones, esfuerzos y fuerzas. El tercer problema, se trata de un sólido de revolución con sección transversal variable en función de la altura, sometido a una carga compresiva P . Aquí, se busca obtener expresiones teóricas y llevar a cabo la codificación del problema mediante MEF¹ unidimensionales. Por último, el cuarto problema aborda una viga circular en estado de tensión plana, donde se busca formular un código de MEF en 2D para analizar el problema, esfuerzos y desplazamientos, estudiar su convergencia respecto a la malla y realizar comparaciones respecto a la solución teórica. Todos los problemas detallados anteriormente serán programados utilizando el software comercial de MATLAB. Para el completo entendimiento de los problemas, se detallará un marco teórico aplicable a cada situación, contemplando la mecánica de sólidos y el método de elementos finitos.

1.1. Objetivo General

Programar códigos de MEF para resolver los problemas de mecánica de sólidos planteados en el documento, con el fin de obtener las variables de interés, como: desplazamientos, esfuerzos, etc.

1.2. Objetivos Específicos

- Detallar un marco teórico de mecánica de sólidos y del método de elementos finitos que sea aplicable a los problemas planteados.
- Encontrar expresiones teóricas (en caso que se requiera) respecto a la solución de cada problema.
- Programar el método de elementos finitos para resolver cada problema.

¹Método de Elementos Finitos

- Obtener las variables de interés (desplazamientos, esfuerzos, etc.) y compararlos con los de referencia para generar una discusión y evaluación de lo realizado y obtenido.

Dicho lo anterior, el documento se organizará de la siguiente manera: Presentación de los problemas a desarrollar, metodología para detallar el marco teórico y su aplicación a cada problema, sección de resultados para visualizar lo obtenido en cada situación, una sección de análisis para generar una discusión de lo obtenido, y una conclusión del documento.

1.3. Problema N°1

La matriz de rigidez en coordenadas locales \mathbf{k}'^k para un elemento unidimensional lineal sometido a una carga axial está dada por,

$$\mathbf{k}'^k = \frac{E^k A^k}{h^k} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

donde E^k , A^k y h^k corresponden al módulo de Young, la sección transversal y el largo del elemento respectivamente. Determine la matriz de transformación \mathbf{L} tal que, $\mathbf{k}^k = \mathbf{L}^T \mathbf{k}'^k \mathbf{L}$ donde \mathbf{k}^k es la matriz de rigidez en coordenadas globales del elemento de acuerdo a la Fig. 1.

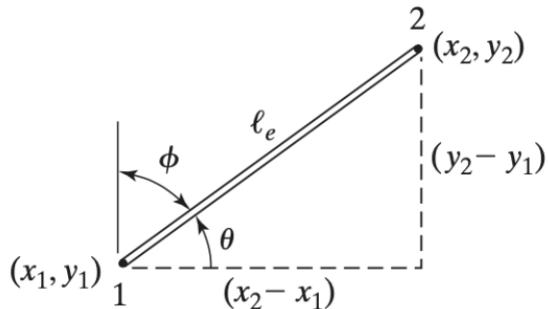


Figura 1: Esquema para la obtención de la matriz L.

1.4. Problema N°2

Utilizando el resultado anterior (matriz \mathbf{L}), determine los desplazamientos, deformaciones, esfuerzos y fuerzas en cada uno de los elementos que componen la armadura mostrada en la Fig. 2. Considere $A^k = A = 250 \text{ mm}^2$ y $E^k = E = 200 \text{ GPa}$ constantes para todas las barras.

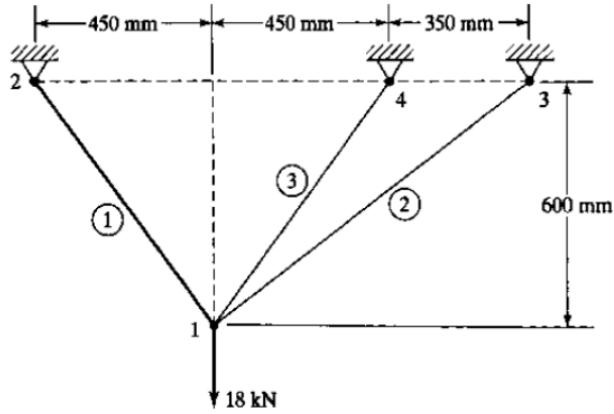


Figura 2: Esquema de la armadura.

1.5. Problema N°3

Un sólido de revolución fabricado en acero soporta una carga P tal como se muestra en la figura 3. El radio varía en función de la profundidad y , de tal forma que se genera un esfuerzo de compresión constante en cualquier sección, siendo r_0 el radio de la parte superior del sólido. Si la altura del sólido es $L = 500$ mm, $r_0 = 50$ mm, $P = 50$ kN y se debe considerar el peso propio,

- Determine la función analítica que describe el radio, en función de la profundidad y .
- Obtenga las expresiones teóricas para los desplazamientos, deformaciones y esfuerzos.
- Desarrolle un código para resolver el problema mediante el Método de Elementos Finitos. Utilice el lenguaje de programación que Ud. estime adecuado considerando funciones de forma lineales.
- Estudie el dominio del sólido de revolución empleando un número adecuado de discretizaciones en función del enfoque de su análisis.
- Contraste los resultados numéricos obtenidos en el ítem c) con los teóricos del ítem b).
- Analice el efecto del cambio de numeración de los nodos en el comportamiento de la matriz de rigidez global del sistema (Sugerencia: Revise la función “spy” de MatLab u otra rutina de similares características).

- g) Estudie el efecto del algoritmo “*symmetric reverse Cuthill-McKee ordering*” en la resolución del sistema de ecuaciones ensambladas (Sugerencia: Revise la función “*symrcm*” de MatLab u otra rutina de similares características).

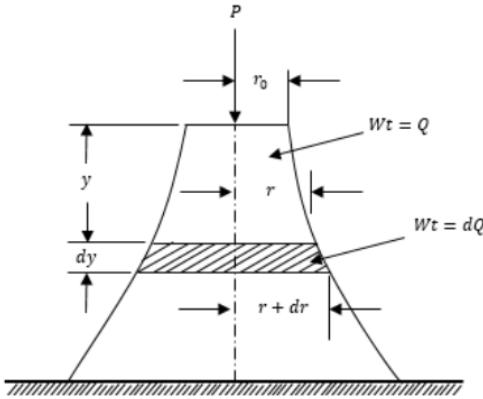


Figura 3: Esquema del sólido de revolución de acero.

1.6. Problema N°4

Considere la viga circular en un estado de tensión plana mostrada en la Fig. 4. La solución analítica de este problema se obtiene en Timoshenko y Goodier [1] usando funciones de esfuerzo de Airy y está dada por,

$$\sigma_r = \frac{P}{N} \left[r + \frac{a^2 b^2}{r^3} - \frac{a^2 + b^2}{r} \right] \sin \theta \quad (1)$$

$$\sigma_\theta = \frac{P}{N} \left[3r - \frac{a^2 b^2}{r^3} - \frac{a^2 + b^2}{r} \right] \sin \theta \quad (2)$$

$$\sigma_r = -\frac{P}{N} \left[r + \frac{a^2 b^2}{r^3} - \frac{a^2 + b^2}{r} \right] \cos \theta \quad (3)$$

donde $N = a^2 - b^2 + (a^2 + b^2) \ln b/a$. Por otro lado, la solución referida a los desplazamientos está dada por,

$$u_r = \frac{P}{NE} \left(\left[\frac{1}{2}(1 - 3\nu)r^2 - \frac{a^2b^2(1 + \nu)}{2r^2} - (a^2 + b^2)(1 - \nu) \right] \sin \theta + (a^2 + b^2)(2\theta - \pi) \cos \theta \right) - K \sin \theta \quad (4)$$

$$u_\theta = \frac{P}{NE} \left(\left[\frac{1}{2}(5 - \nu)r^2 - \frac{a^2b^2(1 + \nu)}{2r^2} - (a^2 + b^2)(1 - \nu) \ln r + (1 - \nu) \right] \sin \theta + (a^2 + b^2)(2\theta - \pi) \cos \theta \right) - K \cos \theta \quad (5)$$

donde, la variable k se obtiene considerando la condición $u_r(a, \pi/2) = 0$, esto es,

$$K = \frac{P}{NE} \left[\frac{1}{2}(1 - 3\nu)a^2 - \frac{b^2(1 + \nu)}{2} - (a^2 + b^2)(1 - \nu) \ln a \right], \quad (6)$$

En las anteriores ecuaciones, E y ν son el módulo de elasticidad y la relación de Poisson, a y b corresponden a los radios interior y exterior, mientras que w es el espesor de la viga. Para esta solución, el desplazamiento u_r en $\theta = 0$ es constante, es decir,

$$u_r(r, 0) = -\frac{\pi P}{2NE}(a^2 + b^2) = u_0. \quad (7)$$

Por lo tanto, en lugar de calcular las fuerzas nodales para la tracción en este límite, simplemente se establecen todos los desplazamientos nodales en la dirección x (dirección r para $\theta = 0$) a un valor constante.

Para resolver numéricamente este problema considere los siguientes parámetros,

$$a = 5, b = 10, w = 1, u_0 = 0, 01, E = 10, 000 \text{ y } \nu = 0, 25 \quad (8)$$

Además, se prescriben las condiciones de contorno para los desplazamiento en coordenadas cartesianas.

$$u_r(x, 0) = u_0 \text{ y } u(0, y) = v(0, a) = 0. \quad (9)$$

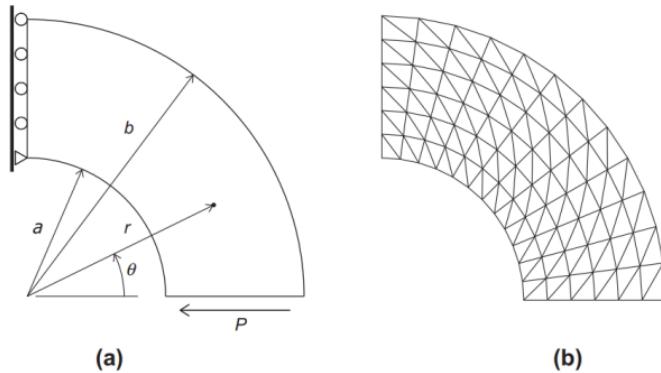


Figura 4: Esquema de una viga circular cargada. (a) Geometría del problema, (b) Ejemplo de una malla triangular.

En función de lo descrito:

- Utilizando los scripts proporcionados en las referencias [2, 3, 4], desarrolle un código para analizar este problema mediante el Método de Elementos Finitos (MEF) [5]. Considere funciones de forma lineales y utilice el lenguaje de programación que Ud. estime adecuado.
- Realice un estudio de convergencia de malla en función de los resultados obtenidos mediante el código desarrollado y su enfoque de análisis. (Sugerencia: Utilice la herramienta “*pdetool*” de MatLab u otra de similares características para realizar las mallas).
- Analice sus resultados numéricos, compárelos con valores de referencia y desarrolle estudios considerando su enfoque de análisis
- Analice el efecto del cambio de numeración de los nodos de la malla en el comportamiento de la matriz de rigidez global del sistema (Sugerencia: Revise la función “*spy*” de MatLab u otra rutina de similares características).
- Estudie el efecto del algoritmo “symmetric reverse Cuthill-McKee ordering” en la resolución del sistema de ecuaciones ensambladas (Sugerencia: Revise la función “*symrcm*” de MatLab u otra rutina de similares características).

2. Metodología

2.1. Problema Elástico

Las ecuaciones de equilibrio para un medio continuo sólido estacionario en forma diferencial, son las siguientes [6]:

$$\begin{aligned}\nabla \cdot \boldsymbol{\sigma} + \mathbf{f} &= 0 \\ \boldsymbol{\sigma} &= \boldsymbol{\sigma}^T\end{aligned}\tag{10}$$

En donde $\boldsymbol{\sigma}$ es el tensor de esfuerzos de Cauchy, y \mathbf{f} las fuerzas de cuerpo.

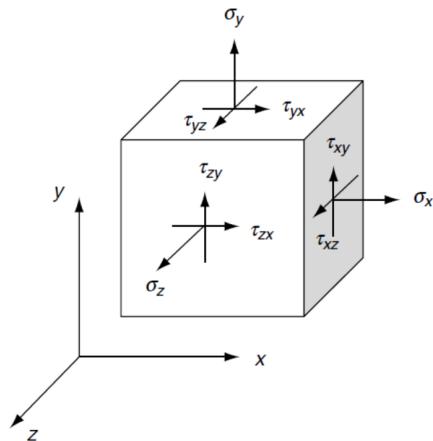


Figura 5: Elemento diferencial

Para este trabajo, se utilizará el modelo de elasticidad lineal, lo que conlleva a las siguientes suposiciones:

- El sólido es continuo.
- El sólido es isotrópico.
- El sólido es homogéneo.
- Existe una relación lineal entre los esfuerzos y las deformaciones.

A partir de las suposiciones presentadas, la relación entre los esfuerzos y las deformaciones será lineal (Ley de Hooke):

$$\begin{aligned}\boldsymbol{\sigma} &= \mathbf{C} : \boldsymbol{\epsilon} \\ \boldsymbol{\sigma} &= \lambda \text{tr}(\boldsymbol{\epsilon}) + 2\mu \boldsymbol{\epsilon}\end{aligned}\tag{11}$$

En donde \mathbf{C} es el tensor de constantes elásticas, λ y μ las constantes de Lamé, y $\boldsymbol{\epsilon}$ el tensor de deformaciones.

$$C_{ijkl} = \lambda \delta_{ij} \delta_{kl} + 2\mu (\delta_{ik} \delta_{jl} + \delta_{il} \delta_{jk})\tag{12}$$

Las constantes de Lamé, λ y μ se relacionan con el módulo de Young E y el coeficiente de Poisson ν como:

$$\mu = G = \frac{E}{2(1+\nu)} ; \quad \lambda = \frac{\nu E}{(1+\nu)(1-2\nu)}\tag{13}$$

Junto a las ecuaciones presentadas anteriormente, se debe considerar la relación entre deformaciones y desplazamientos. Estas tienen la siguiente forma:

$$\boldsymbol{\epsilon} = \frac{1}{2} (\nabla \mathbf{u} + \nabla \mathbf{u}^T)\tag{14}$$

En donde \mathbf{u} es el vector de desplazamientos.

Todas las ecuaciones detalladas anteriormente forman un sistema de 15 ecuaciones y 15 incógnitas (6 componentes de esfuerzo, 6 componentes de deformación y 3 desplazamientos), las cuales bajo condiciones de borde adecuadas pueden ser resueltas.

Existen formulaciones, como las ecuaciones de Lamé-Navier para elasticidad, que simplifican este sistema de 15 ecuaciones en 3. Estas 3 ecuaciones pueden ser resueltas para los desplazamientos, y de esa forma, obtener las deformaciones o esfuerzos si se requiere.

$$\mu \nabla^2 \mathbf{u} + (\mu + \lambda) \nabla (\nabla \cdot \mathbf{u}) + \mathbf{f} = 0\tag{15}$$

2.1.1. Función de Esfuerzos de Airy

De la misma forma, para problemas 2D, se pueden escribir las ecuaciones en términos de un potencial Φ denominado: función de esfuerzos de Airy. La función de esfuerzos de Airy cumple las siguientes relaciones, en coordenadas cartesianas:

$$\sigma_{xx} = \frac{\partial^2 \Phi}{\partial y^2} + V_f ; \quad \sigma_{yy} = \frac{\partial^2 \Phi}{\partial x^2} + V_f ; \quad \sigma_{xy} = -\frac{\partial^2 \Phi}{\partial x \partial y} \quad (16)$$

Para utilizar esta formulación, las fuerzas de cuerpo deben ser conservativas, siendo derivables del potencial V_f . Reemplazando la definición de la función de Airy en las ecuaciones de equilibrio dadas en la ecuación 10, se obtiene:

$$\nabla^4 \Phi + (2 - s) \nabla^2 V_f = 0 \quad (17)$$

En donde s es un parámetro utilizado para diferenciar condición de esfuerzo plano o deformación plana.

2.1.2. Formulación Débil de Galerkin

El problema elástico puede ser escrito en forma variacional utilizando el método de Galerkin. Para esto, se deben integrar los residuales de la ecuación 10 multiplicados por ϕ , correspondiente a un desplazamiento virtual arbitrario consistente con las condiciones de borde y cinemáticamente admisible. De esta manera, se obtiene la siguiente ecuación:

$$\int_{\Omega} \boldsymbol{\sigma} : \boldsymbol{\varepsilon}(\phi) \, dV - \int_{\Omega} \mathbf{f} \cdot \phi \, dV - \int_{\partial\Omega} \mathbf{T}^n \cdot \phi \, dS - \sum_{x_i \in \partial\Omega} \mathbf{P}_i \cdot \phi_i = 0 \quad (18)$$

En donde $\boldsymbol{\sigma}$ corresponde al tensor de esfuerzos, $\boldsymbol{\varepsilon}(\phi)$ las deformaciones virtuales, \mathbf{f} las fuerzas de cuerpo, \mathbf{T}^n al vector de esfuerzos, \mathbf{P} a las cargas puntuales y ϕ al desplazamiento virtual. Esta ecuación coincide con el principio de trabajo virtual.

Comúnmente, esta formulación del problema elástico es utilizada para el método de elementos finitos. El detalle de esta aplicación se revisará en la siguiente sección.

2.2. Método de Elementos Finitos

El método de Elementos Finitos busca encontrar la mejor aproximación de la solución del problema a partir de un conjunto de funciones dadas. Para esto, se discretiza el dominio en elementos y se resuelve para el desplazamiento en los nodos de la malla. La solución en los elementos es interpolada por los desplazamientos nodales y las funciones de forma seleccionadas. El detalle del método se visualizará al demostrar las discretizaciones en 1D y 2D.

Con lo dicho anteriormente, se busca resolver la ecuación 18 en el dominio discretizado. De esta forma, se obtiene el siguiente sistema de ecuaciones.

$$\sum_e \int_e \boldsymbol{\sigma} : \boldsymbol{\varepsilon}(\boldsymbol{\phi}) \, dV = \sum_e \int_e \boldsymbol{f} \cdot \boldsymbol{\phi} \, dV + \sum_e \int_e \boldsymbol{T}^n \cdot \boldsymbol{\phi} \, dS + \sum_e \sum_{x_i \in \partial\Omega} \boldsymbol{P}_i \cdot \boldsymbol{\phi}_i \quad (19)$$

En donde e hace referencia a cada elemento. Este sistema de ecuaciones debe ser trabajado para cada aplicación. Sin embargo, se podrá llegar al siguiente sistema de ecuaciones lineal:

$$\boldsymbol{K}\boldsymbol{Q} = \boldsymbol{F} \quad (20)$$

En donde la matriz \boldsymbol{K} corresponde a la matriz de rigidez global ensamblada a partir de las matrices de rigidez de cada elemento, \boldsymbol{Q} el vector de desplazamientos en cada nodo, y \boldsymbol{F} el vector de fuerzas ensamblado, el cual considera las fuerzas de cuerpo y las fuerzas de superficie externas en cada elemento.

2.2.1. Discretización en 1D

Para la discretización en 1 dimensión, se discretiza el dominio en n elementos unidimensionales de la siguiente forma: Para cada elemento, se tendrá un vector de desplazamientos nodales $\mathbf{q} = [q_1, q_2]^T$ que tiene la información del desplazamiento en cada nodo del elemento (en numeración local).

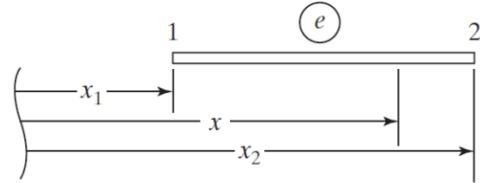


Figura 6: Elemento 1D

El desplazamiento al interior del elemento será interpolado con las siguientes funciones de forma en coordenadas normalizadas ($\xi = -1$ en x_1 y $\xi = 1$ en x_2).

$$N_1(\xi) = \frac{1 - \xi}{2} ; \quad N_2(\xi) = \frac{1 + \xi}{2} \quad (21)$$

De manera que el desplazamiento u puede ser escrito como:

$$u = \mathbf{N}\mathbf{q} \quad (22)$$

Con $\mathbf{N} = [N_1, N_2]$

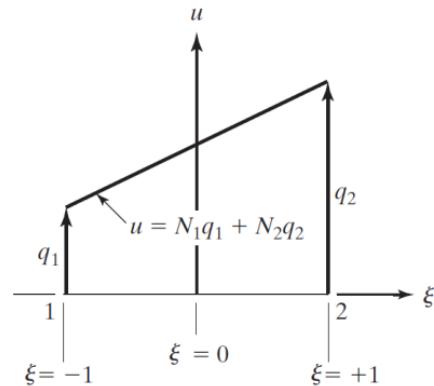


Figura 7: Elemento 1D

A partir de esta definición, la deformación en el elemento puede ser calculada como:

$$\boldsymbol{\varepsilon} = \mathbf{B}\mathbf{q} \quad (23)$$

En donde la matriz \mathbf{B} toma la siguiente forma:

$$\mathbf{B} = \frac{1}{l_e} [1, -1] \quad (24)$$

Siendo $l_e = x_2 - x_1$ el largo del elemento. Por ultimo, el esfuerzo normal dentro del elemento puede ser calculado con la Ley de Hooke.

$$\sigma = E\mathbf{B}\mathbf{q} \quad (25)$$

Por otra parte, el desplazamiento virtual será escrito a partir de las mismas funciones de forma \mathbf{N} . Así:

$$\boldsymbol{\phi} = \mathbf{N}\boldsymbol{\psi} ; \quad \boldsymbol{\varepsilon}(\boldsymbol{\phi}) = \mathbf{B}\boldsymbol{\psi} \quad (26)$$

Siendo $\boldsymbol{\psi}$ los desplazamientos nodales arbitrarios. Con estas definiciones, las integrales para cada elemento dadas en la ecuación 19 pueden ser aproximadas de la siguiente manera:

1. Término relacionado con el tensor de esfuerzos:

$$\int_e \sigma \boldsymbol{\varepsilon}(\boldsymbol{\phi}) A dx = \boldsymbol{\psi}^T \mathbf{k}^e \mathbf{q} \quad (27)$$

De donde se obtiene la matriz de rigidez local \mathbf{k}^e para cada elemento como:

$$\mathbf{k}^e = \mathbf{B}^T \mathbf{B} \int_e E A dx \quad (28)$$

2. Término relacionado con las fuerzas de cuerpo y fuerzas externas:

$$\int_e f \boldsymbol{\phi} A dx = \boldsymbol{\psi}^T \mathbf{f}_{body}^e \quad (29)$$

En donde el vector de fuerzas \mathbf{f}_{body}^e puede calcularse como:

$$\mathbf{f}_{body}^e = \int_e f \mathbf{N}^T A dx \quad (30)$$

3. Términos relacionados con las fuerzas externas:

$$\int_e T\phi \, dx + \sum_{x_i \in \partial\Omega} P_i \phi_i = \boldsymbol{\psi}^T \mathbf{f}_{ext}^e \quad (31)$$

En donde el vector de fuerzas \mathbf{f}_{ext}^e puede calcularse como:

$$\mathbf{f}_{ext}^e = \int_e T^n \mathbf{N}^T \, dx + \sum_i P_i \quad (32)$$

Con esta discretización, la matriz \mathbf{K} se obtiene al ensamblar todas las matrices de rigidez de cada elemento \mathbf{k}^e y \mathbf{F} a partir de los vectores de fuerzas de cuerpo y externas para cada elemento $\mathbf{f}_{body}^e, \mathbf{f}_{ext}^e$. De esta manera, el problema puede ser resuelto utilizando la ecuación 20.

2.2.2. Discretización 2D

Para el caso 2D, se utilizará la siguiente numeración local para cada elemento. Por simplicidad, se considerarán solo elementos de 3 vértices. De esta manera, el vector de desplazamientos nodales local queda como: $\mathbf{q} = [q_1, q_2, \dots, q_6]^T$.

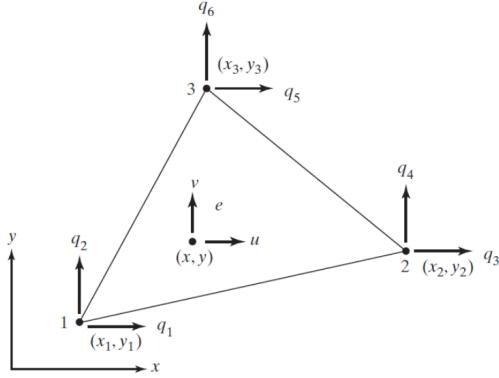


Figura 8: Elemento 2D

Para las funciones de forma, se utilizarán funciones lineales de la forma:

$$N_1(\xi, \eta) = \xi ; \quad N_2(\xi, \eta) = \eta ; \quad N_3(\xi, \eta) = 1 - \xi - \eta \quad (33)$$

De esta manera, el desplazamiento en el interior del elemento puede ser calculado como:

$$\mathbf{u} = \mathbf{N}\mathbf{q} \quad (34)$$

En donde la matriz \mathbf{N} tiene la siguiente forma:

$$\mathbf{N} = \begin{bmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 \end{bmatrix} \quad (35)$$

Con esta definición, las deformaciones en el elemento pueden ser calculadas como:

$$\boldsymbol{\varepsilon} = \mathbf{B}\mathbf{q} \quad (36)$$

En donde la matriz \mathbf{B} puede ser calculada como:

$$\mathbf{B} = \frac{1}{x_{13}y_{23} - x_{23}y_{13}} \begin{bmatrix} y_{23} & 0 & y_{31} & 0 & y_{12} & 0 \\ 0 & x_{32} & 0 & x_{13} & 0 & x_{21} \\ x_{32} & y_{23} & x_{13} & y_{31} & x_{21} & y_{12} \end{bmatrix} \quad (37)$$

Por último, los esfuerzos pueden ser calculados como:

$$\boldsymbol{\sigma} = \mathbf{C}\mathbf{B}\mathbf{q} \quad (38)$$

En donde \mathbf{C} corresponde a la matriz de constantes elásticas dada por:

$$\mathbf{C} = \frac{E}{1-\nu^2} \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2} \end{bmatrix} \quad (39)$$

Considerar que para esta formulación en 2D, se utiliza el vector de desplazamiento $\mathbf{u} = [u_x, u_y]^T$, las deformaciones en forma de vector $\boldsymbol{\varepsilon} = [\varepsilon_{xx}, \varepsilon_{yy}, \varepsilon_{xy}]^T$ y los esfuerzos en forma de vector $\boldsymbol{\sigma} = [\sigma_{xx}, \sigma_{yy}, \sigma_{xy}]^T$. Lo anterior permite mayor simplicidad sin pérdida de generalidad.

De la misma forma que en el caso 1D, se plantea un desplazamiento virtual ϕ que se escribe como:

$$\phi = \mathbf{N}\psi ; \quad \boldsymbol{\varepsilon}(\phi) = \mathbf{B}\psi \quad (40)$$

En donde ψ corresponde a los desplazamientos nodales arbitrarios en numeración local.

Con estas definiciones, las integrales para cada elemento dadas en la ecuación 19 pueden ser aproximadas de la siguiente manera:

1. Término relacionado con el tensor de esfuerzos:

$$\int_e \boldsymbol{\sigma}^T \boldsymbol{\varepsilon}(\phi) t \, dA = \psi^T \mathbf{k}^e \mathbf{q} \quad (41)$$

De donde se obtiene la matriz de rigidez local \mathbf{k}^e para cada elemento como:

$$\mathbf{k}^e = \mathbf{B}^T \int_e \mathbf{C} t \, dA \mathbf{B} \quad (42)$$

2. Término relacionado con las fuerzas de cuerpo y fuerzas externas:

$$\int_e \boldsymbol{\phi}^T \mathbf{f}_t \, dA = \boldsymbol{\psi}^T \mathbf{f}_{body}^e \quad (43)$$

En donde el vector de fuerzas \mathbf{f}_{body}^e puede calcularse como:

$$\mathbf{f}_{body}^e = \int_e \mathbf{N}^T \mathbf{f}_t \, dA \quad (44)$$

3. Términos relacionados con las fuerzas externas:

$$\int_e \boldsymbol{\phi}^T \mathbf{T}^n \, dl + \sum_{x_i \in \partial\Omega} \boldsymbol{\phi}_i^T \mathbf{P}_i = \boldsymbol{\psi}^T \mathbf{f}_{ext}^e \quad (45)$$

En donde el vector de fuerzas \mathbf{f}_{ext}^e puede calcularse como:

$$\mathbf{f}_{ext}^e = \int_e \mathbf{N}^T \mathbf{T}^n \, dl + \sum_i \mathbf{P}_i \quad (46)$$

Con esta discretización, la matriz \mathbf{K} se obtiene al ensamblar todas las matrices de rigidez de cada elemento \mathbf{k}^e y \mathbf{F} a partir de los vectores de fuerzas de cuerpo y externas para cada elemento $\mathbf{f}_{body}^e, \mathbf{f}_{ext}^e$. De esta manera, el problema puede ser resuelto utilizando la ecuación 20.

2.3. Metodología aplicado a cada problema

En la siguiente sección se presenta la metodología para la resolución de cada problema. Principalmente se plantea una formulación teórica del problema basado en la sección anterior, y se entregan detalles de como se implementará el código. Todos los problemas se resolverán utilizando MEF con códigos elaborados en Matlab.

2.4. Problema N°1 y N°2

Para la determinación de la matriz de transformación, se requiere, en primer lugar, considerar un elemento compuesto por los nodos 1 y 2, que posee su propio sistema de coordenadas local x' - y' . Este sistema local se reflejará en el sistema de coordenadas global X-Y.

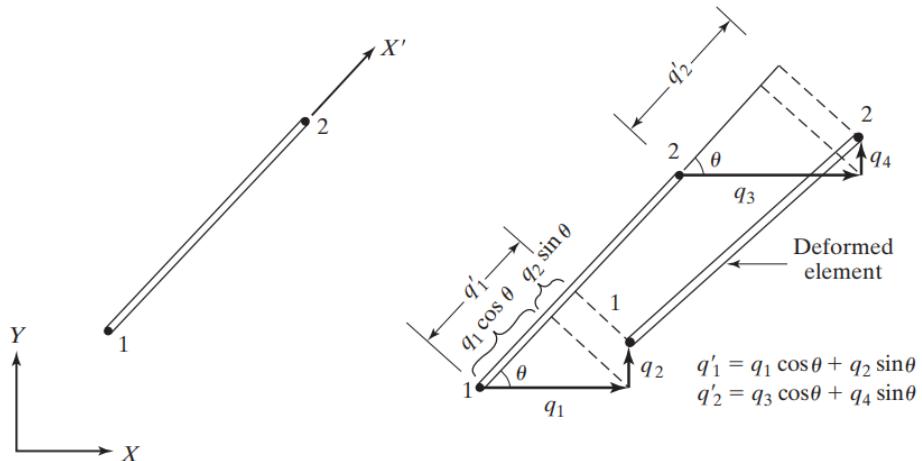


Figura 9: Elemento bidimensional

En relación con los desplazamientos en cada nodo, se observa que el nodo 1 presenta un desplazamiento representado como q'_1 en sus coordenadas locales, mientras que el nodo 2 tiene un desplazamiento q'_2 en esas mismas coordenadas locales. Estos desplazamientos locales se expresan de forma vectorial como:

$$q' = [q'_1, q'_2]^T \quad (47)$$

Sin embargo, también es posible representar el vector de desplazamiento en el sistema de coordenadas global, denotado como:

$$q = [q_1, q_2, q_3, q_4]^T \quad (48)$$

Existe una relación entre los vectores q' y q , la cual se establece mediante la suma de las proyecciones de cada nodo.

$$q'_1 = q_1 \cos \theta + q_2 \sin \theta \quad (49)$$

$$q'_2 = q_3 \cos \theta + q_4 \sin \theta \quad (50)$$

Al escribir de forma matricial la relación entre los desplazamientos locales q' y los desplazamientos globales q , se obtiene:

$$\begin{bmatrix} q'_1 \\ q'_2 \end{bmatrix} = \begin{bmatrix} \cos \theta & \sin \theta & 0 & 0 \\ 0 & 0 & \cos \theta & \sin \theta \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} \quad (51)$$

Esta relación puede ser expresada de manera más compacta como $q' = L q$, donde L representa la matriz de transformación.

Para facilitar los cálculos, se introduce el coseno director l y m , los cuales corresponden a los ángulos que forman las coordenadas locales con respecto a las coordenadas globales.

$$L = \begin{bmatrix} \cos \theta & \sin \theta & 0 & 0 \\ 0 & 0 & \cos \theta & \sin \theta \end{bmatrix} = \begin{bmatrix} l & m & 0 & 0 \\ 0 & 0 & l & m \end{bmatrix} \quad (52)$$

Los cosenos directores l y m son calculados según las siguientes expresiones:

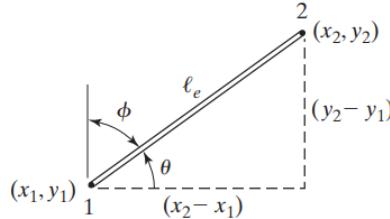


Figura 10: Cosenos directores

$$l = \cos \theta = \frac{x_2 - x_1}{l_e} \quad (53)$$

$$m = \sin \theta = \frac{y_2 - y_1}{l_e} \quad (54)$$

Donde l_e representa la longitud del elemento y se define como:

$$l_e = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad (55)$$

Para determinar los desplazamientos, deformaciones, esfuerzos y fuerzas en cada uno de los elementos que conforman la armadura, como se muestra en la Figura 2, es necesario establecer la matriz de rigidez del sistema de coordenadas global. Para llevar a cabo este proceso, se inicia analizando la matriz de rigidez en el elemento, estableciendo su relación con la energía de deformación del mismo.

$$U_e = \frac{1}{2} q'^T k'^k q' \quad (56)$$

Sustituyendo la relación $q' = Lq$ en la ecuación 56, se tiene:

$$U_e = \frac{1}{2} q^T [L^T k'^k L] q \quad (57)$$

La energía de deformación global se expresa mediante la siguiente ecuación:

$$U_e = \frac{1}{2} q^T k^k q \quad (58)$$

Donde $k^k = L^T k'^k L$ representa la matriz de rigidez global de cada elemento y k'^k es la matriz de rigidez local de cada elemento. Utilizando la nomenclatura de cosenos de dirección, la matriz de rigidez global k^k se puede calcular como:

$$k^k = \frac{E^k A^k}{h^k} \begin{bmatrix} l^2 & lm & -l^2 & -lm \\ lm & m^2 & -lm & -m^2 \\ -l^2 & -lm & l^2 & lm \\ -lm & -m^2 & lm & m^2 \end{bmatrix} \quad (59)$$

Donde E_k , A_k y $h_k = l_e$ representan el módulo de Young, la sección transversal y la longitud del elemento, respectivamente.

Para determinar la deformación en cada elemento, se establece la relación:

$$\varepsilon = B' q'$$

La matriz de deformación-desplazamiento del elemento, B' , se define como:

$$B' = \frac{1}{l_e} \begin{bmatrix} -1 & 1 \end{bmatrix} \quad (60)$$

Para determinar el esfuerzo en el elemento, se aplica la ley de Hooke, expresada como:

$$\sigma = E^k B' q' = E^k B' L q \quad (61)$$

Esto se traduce en la siguiente expresión matricial:

$$\sigma = \frac{E^k}{l_e} \begin{bmatrix} -l & -m & l & m \end{bmatrix} q \quad (62)$$

Después de obtener la matriz de rigidez de cada elemento, se procede a ensamblar las matrices para obtener la matriz de rigidez global. Este procedimiento permite determinar las fuerzas internas y externas de la armadura, mediante un sistema de ecuaciones representado de la siguiente manera:

$$R = KQ - F \quad (63)$$

Donde K es la matriz de rigidez global, Q el desplazamiento nodal global, F representa las fuerzas externas aplicadas en la estructura y R son las reacciones en los soportes.

Cabe mencionar que el sistema de ecuaciones planteado debe cumplir con la condición de frontera de Dirichlet. Esta condición establece que no debe haber desplazamiento en los soportes, lo cual se expresa mediante las restricciones nodales en el desplazamiento, es decir, $Q_3 = Q_4 = Q_5 = Q_6 = Q_7 = Q_8 = 0$. Este requisito garantiza la consistencia del análisis estructural y refleja las condiciones físicas de inmovilidad en los puntos de apoyo.

Para resolver el problema 2, es necesario en primer lugar establecer el módulo de Young (E), su área transversal (A), y las fuerzas que actúan sobre la armadura.

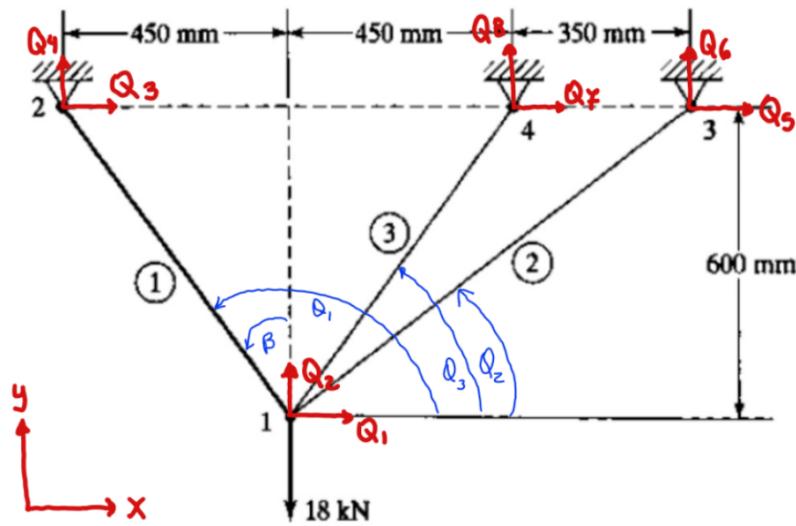


Figura 11: Análisis estructural

Basándose en la figura 11, se procede a tabular las coordenadas nodales y la información correspondiente a los elementos de la estructura.

Tabla 1: Coordenadas nodales

Nodo	x [mm]	y [mm]
1	0	0
2	-450	600
3	800	600
4	450	600

Tabla 2: Conectividad de elementos

Elemento	1	2
1	1	2
2	1	3
3	1	4

Utilizando esta nomenclatura y las expresiones elaboradas, se obtendrá con un script en Matlab, el desplazamiento en los nodos; las fuerzas, deformaciones y esfuerzos en los elementos; y las reacciones en los apoyos.

2.5. Problema N°3

2.5.1. Desarrollo teórico

Para este problema, se tendrá un sólido de revolución de acero que soporta una carga P . Debido a la geometría de este sólido, la carga P junto al peso propio del sólido generan un esfuerzo normal cortante en cualquier sección, incluida la sección superior. Se considerará $y = 0$ en la superficie superior del sólido, con dirección positiva apuntando hacia abajo del sólido. Esto con el fin de dejar todo en función de la profundidad.

El esfuerzo en la cara superior, puede ser calculado como:

$$\sigma_0 = \frac{P}{\pi r_0^2} \quad (64)$$

Así, el esfuerzo en cualquier sección en función de la profundidad y se calcula como:

$$\sigma(y) = \sigma_0 \quad (65)$$

La fuerza total sobre cada sección F considera la carga P y el peso de la columna superior.

$$F(y) = P + \gamma \int_0^y A(y') dy' \quad (66)$$

En donde $A(y)$ es la sección en función de la profundidad y y γ el peso específico. Utilizando esta notación, el esfuerzo en cada sección puede ser escrito como:

$$\sigma(y) = \frac{P + \gamma \int_0^y A(y') dy'}{A(y)} \quad (67)$$

Y derivando respecto a la profundidad y , y utilizando el hecho de que el esfuerzo es constante se obtiene:

$$\sigma_0 \frac{dA}{dy} = \gamma A(y) \quad (68)$$

La solución de esta ecuación diferencial obtiene la expresión de la sección A en función de la profundidad:

$$A(y) = A_0 e^{ky} \quad (69)$$

En donde $A_0 = \pi r_0^2$ y $k = \gamma/\sigma_0$. Utilizando esta expresión, se puede obtener fácilmente la expresión

para el radio de la sección en función de la profundidad $r(y)$:

$$r(y) = r_0 e^{ky/2} \quad (70)$$

A partir del resultado anterior, se pueden obtener las expresiones para el desplazamiento y la deformación teórica. Producto que el esfuerzo tiene la siguiente expresión:

$$\sigma(y) = \sigma_0 \quad (71)$$

La expresión para la deformación ε es obtenible utilizando la Ley de Hooke. De esta manera:

$$\varepsilon(y) = \frac{\sigma_0}{E} \quad (72)$$

Por último, el desplazamiento v puede ser obtenido como:

$$\begin{aligned} v(y) &= \int_y^L \varepsilon(y') dy' \\ v(y) &= \frac{\sigma_0}{E} (L - y) \end{aligned} \quad (73)$$

En donde se nota claramente la relación lineal del desplazamiento y la profundidad. Además, $v(0) = \sigma_0 L / E$ y $v(L) = 0$.

2.5.2. Formulación de MEF

Para adaptar este problema para ser aplicado bajo el método de Elementos Finitos, se discretizará el sólido en elementos unidimensionales de igual longitud l_e . Se utilizará la numeración global partiendo desde el nodo N°1 en la superficie del sólido. Debido a esto, para cada elemento se obtendrá lo siguiente:

- Matriz de rigidez local: Se debe resolver la expresión dada en la ecuación 28.

$$\begin{aligned} \mathbf{k}^e &= \mathbf{B}^T \mathbf{B} \int_e EA \, dx \\ \mathbf{k}^e &= \mathbf{B}^T \mathbf{B} E \int_{y_1}^{y_2} A_0 e^{ky} \, dy \\ \mathbf{k}^e &= \mathbf{B}^T \mathbf{B} E A_0 \frac{1}{k} (e^{ky_2} - e^{ky_1}) \\ \mathbf{k}^e &= \frac{E}{k} \mathbf{B}^T \mathbf{B} (A(y_2) - A(y_1)) \end{aligned} \quad (74)$$

En donde se prefiere dejar la matriz de rigidez en función de las áreas de los nodos del elemento, simplemente por simplicidad a la hora de programar el método.

- Vector de fuerzas de cuerpo: Se debe resolver la expresión dada en la ecuación 30.

$$\mathbf{f}_{body}^e = \int_e f \mathbf{N}^T A \, dx \quad (75)$$

Donde cada componente $j = 1, 2$ se calcula como:

$$\begin{aligned} f_j &= \gamma \int_{y_1}^{y_2} \frac{y_j - y}{l_e} (-1)^j A_0 e^{ky} \, dy \\ f_j &= \frac{\gamma}{l_e} (-1)^j \left[\frac{y_j}{k} (A(y_2) - A(y_1)) - \frac{1}{k^2} ((ky_2 - 1)A(y_2) - (ky_1 - 1)A(y_1)) \right] \end{aligned} \quad (76)$$

De la misma forma que para la matriz de rigidez, se prefiere dejar las componentes del vector de fuerzas de cuerpo en función del área de la sección en cada nodo.

Por último, se fijará como condición de borde:

$$v(L) = Q_1 = 0 \quad (77)$$

Las cuales corresponden a condiciones de Dirichlet que serán aplicadas en primer nodo.

Para cada solución obtenida, se calculará el error L_2 relativo respecto a la solución teórica para el desplazamiento y el esfuerzo.

$$\|e\|_2 = \sqrt{\frac{\sum_i (\alpha_{i,aprox} - \alpha_{i,teorica})}{\sum_i \alpha_{i,teorica}}} \quad (78)$$

Donde $\|e\|_2$ corresponde al error L_2 relativo y α la variable de interés.

Además, se estudiará el cambio de numeración de nodos en el comportamiento de la matriz de rigidez global del sistema y el efecto del algoritmo “*symmetric reverse Cuthill-McKee ordering*” en la resolución del sistema de ecuaciones ensambladas.

2.6. Problema N°4

2.6.1. Desarrollo teórico

Como se planteó en el enunciado, la solución teórica del problema se puede desarrollar a partir de la función de esfuerzos de Airy, en donde se obtiene en coordenadas polares:

$$\sigma_r = \frac{P}{N} \left[r + \frac{a^2 b^2}{r^3} - \frac{a^2 + b^2}{r} \right] \sin \theta, \quad (79)$$

$$\sigma_\theta = \frac{P}{N} \left[3r - \frac{a^2 b^2}{r^3} - \frac{a^2 + b^2}{r} \right] \sin \theta, \quad (80)$$

$$\sigma_r = -\frac{P}{N} \left[r + \frac{a^2 b^2}{r^3} - \frac{a^2 + b^2}{r} \right] \cos \theta, \quad (81)$$

donde $N = a^2 - b^2 + (a^2 + b^2) \ln b/a$. La solución referida a los desplazamientos está dada por,

$$u_r = \frac{P}{NE} \left(\left[\frac{1}{2}(1-3\nu)r^2 - \frac{a^2 b^2(1+\nu)}{2r^2} - (a^2 + b^2)(1-\nu) \right] \sin \theta + (a^2 + b^2)(2\theta - \pi) \cos \theta \right) - K \sin \theta \quad (82)$$

$$u_\theta = \frac{P}{NE} \left(\left[\frac{1}{2}(5-\nu)r^2 - \frac{a^2 b^2(1+\nu)}{2r^2} - (a^2 + b^2)(1-\nu) \ln r + (1-\nu) \right] \sin \theta + (a^2 + b^2)(2\theta - \pi) \cos \theta \right) - K \cos \theta \quad (83)$$

donde, la variable K se obtiene considerando la condición $u_r(a, \pi/2) = 0$, esto es,

$$K = \frac{P}{NE} \left[\frac{1}{2}(1-3\nu)a^2 - \frac{b^2(1+\nu)}{2} - (a^2 + b^2)(1-\nu) \ln a \right], \quad (84)$$

2.6.2. Formulación de MEF

Para adaptar este problema para ser aplicado bajo el método de Elementos Finitos, se discretizará el sólido en elementos bidimensionales triangulares. Este problema no tiene fuerzas de cuerpo ni fuerzas externas, por lo que será de interés solo la matriz de rigidez. Para esto, para cada elemento se tendrá lo siguiente:

- Matriz de rigidez local: Se debe resolver la expresión dada en la ecuación 28.

$$\begin{aligned}\mathbf{k}^e &= \mathbf{B}^T \int_e C t \, dA \, \mathbf{B} \\ \mathbf{k}^e &= t_e A_e \mathbf{B}^T \mathbf{C} \mathbf{B}\end{aligned}\tag{85}$$

Por último, se fijará como condición de borde:

$$u(x, 0) = u_0 ; \quad u(0, y) = v(0, y) = 0\tag{86}$$

Las cuales corresponden a condiciones de Dirichlet que serán aplicadas en los nodos respectivos.

Para cada solución obtenida, se calculará el error L_2 relativo respecto a la solución teórica para el desplazamiento y el esfuerzo en cada nodo, y respecto al esfuerzo de Von Mises máximo.

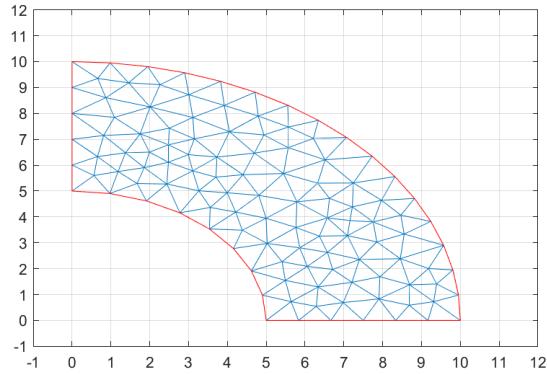
$$\|e\|_2 = \sqrt{\frac{\sum_i (\alpha_{i,aprox} - \alpha_{i,teorica})}{\sum_i \alpha_{i,teorica}}}\tag{87}$$

Donde $\|e\|_2$ corresponde al error L_2 relativo y α la variable de interés.

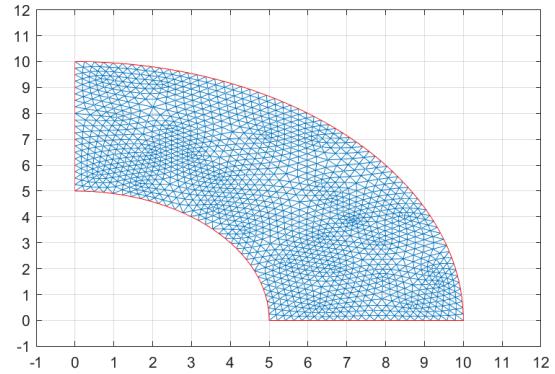
Adicionalmente, se realizará un estudio de convergencia de malla. Este se realizará monitorizando el valor del esfuerzo de Von Mises máximo y el desplazamiento máximo en todo el sólido. Con esto se revisa la convergencia de una variable primaria y una secundaria. Además, se estimará el error a “posteriori” de la malla utilizando la expresión desarrollada en [2].

Finalmente, se estudiará el cambio de numeración de nodos en el compartimiento de la matriz de rigidez global del sistema y el efecto del algoritmo “*symmetric reverse Cuthill-McKee ordering*” en la resolución del sistema de ecuaciones ensambladas.

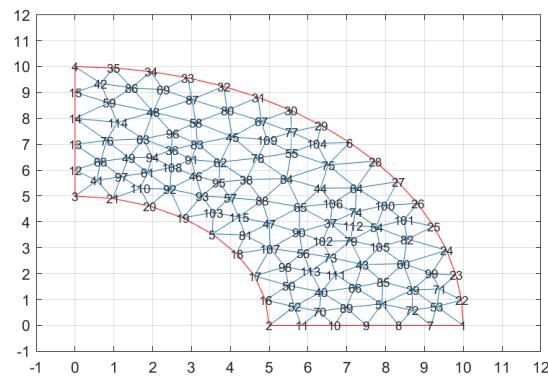
Para la discretización del sólido se utilizará la herramienta de PDETool de Matlab. Este software permite mallar fácilmente con elementos triangulares.



(a) Malla de $N = 115$



(b) Malla de $N = 1615$



(c) Numeración de nodos para malla de $N = 115$

Figura 12: Discretización del sólido utilizando herramienta de PDETool

3. Resultados

3.1. Problema N°1 y N°2

A partir de la figura 11, se pueden determinar los ángulos de inclinación, con los siguientes resultados:

$$\theta_1 = 90^\circ + \beta = 90^\circ + \tan^{-1} \left(\frac{450}{600} \right) = 126.87^\circ$$

$$\theta_2 = \tan^{-1} \left(\frac{600}{800} \right) = 36.87^\circ$$

$$\theta_3 = \tan^{-1} \left(\frac{600}{400} \right) = 53.13^\circ$$

Introduciendo los cosenos de dirección, se obtienen los siguientes valores para cada elemento, como se muestra en la tabla 3:

Tabla 3: Cosenos de dirección

Elemento	l_e [mm]	l	m
1	750	-0.6	0.8
2	1000	0.8	0.6
3	750	0.6018	0.7986

Es relevante señalar que se ha desarrollado un programa en Matlab para obtener los resultados del problema. No obstante, en esta sección se proporciona en detalle el procedimiento seguido.

Se inicia el cálculo de la matriz de rigidez para cada elemento utilizando la ecuación 59.

$$k_1 = \frac{30 \cdot 10^6 \cdot 250}{750} \begin{bmatrix} 0.3600 & -0.4800 & -0.3600 & 0.4800 \\ -0.4800 & 0.6400 & 0.4800 & -0.6400 \\ -0.3600 & 0.4800 & 0.3600 & -0.4800 \\ 0.4800 & -0.6400 & -0.4800 & 0.6400 \end{bmatrix} \quad (88)$$

$$k_2 = \frac{30 \cdot 10^6 \cdot 250}{1000} \begin{bmatrix} 0.6400 & 0.4800 & -0.6400 & -0.4800 \\ 0.4800 & 0.3600 & -0.4800 & -0.3600 \\ -0.6400 & -0.4800 & 0.6400 & 0.4800 \\ -0.4800 & -0.3600 & 0.4800 & 0.3600 \end{bmatrix} \quad (89)$$

$$k_3 = \frac{30 \cdot 10^6 \cdot 250}{750} \begin{bmatrix} 0.3622 & 0.4806 & -0.3622 & -0.4806 \\ 0.4806 & 0.6378 & -0.4806 & -0.6378 \\ -0.3622 & -0.4806 & 0.3622 & 0.4806 \\ -0.4806 & -0.6378 & 0.4806 & 0.6378 \end{bmatrix} \quad (90)$$

Es crucial destacar que, si bien la matriz de rigidez generada para cada elemento es de dimensiones 4x4, el ensamblaje total resulta en una matriz de 8x8. Para comprender el orden del ensamblaje, se presenta una tabla con la numeración global en la siguiente forma:

Tabla 4: Numeración global

Nodo	Q_x	Q_y
1	Q_1	Q_2
2	Q_3	Q_4
3	Q_5	Q_6
4	Q_7	Q_8

Posteriormente, se procede al ensamblaje de la matriz de rigidez K , sumando las contribuciones de los elementos y considerando la conectividad.

$$K = 10^5 \begin{bmatrix} 0.8015 & 0.2404 & -0.2400 & 0.3200 & -0.3200 & -0.2400 & -0.2415 & -0.3204 \\ 0.2404 & 1.0319 & 0.3200 & -0.4267 & -0.2400 & -0.1800 & -0.3204 & -0.4252 \\ -0.2400 & 0.3200 & 0.2400 & -0.3200 & 0 & 0 & 0 & 0 \\ 0.3200 & -0.4267 & -0.3200 & 0.4267 & 0 & 0 & 0 & 0 \\ -0.3200 & -0.2400 & 0 & 0 & 0.3200 & 0.2400 & 0 & 0 \\ -0.2400 & -0.1800 & 0 & 0 & 0.2400 & 0.1800 & 0 & 0 \\ -0.2415 & -0.3204 & 0 & 0 & 0 & 0 & 0.2415 & 0.3204 \\ -0.3204 & -0.4252 & 0 & 0 & 0 & 0 & 0.3204 & 0.4252 \end{bmatrix} \quad (91)$$

Para reducir el sistema, es necesario aplicar las condiciones de contorno. Basándose en la figura 11, se observa que no hay desplazamiento en los nodos 2, 3 y 4, es decir, $Q_3 = Q_4 = Q_5 = Q_6 = Q_7 = Q_8 = 0$. Además, se aplica una fuerza en dirección negativa de 18.000 N en el eje Y.

Entonces el sistema reducido queda como:

$$10^5 \cdot \begin{bmatrix} 0.8 & 0.24 \\ 0.24 & 1.04 \end{bmatrix} \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} = \begin{bmatrix} 0 \\ -18000 \end{bmatrix} \quad (92)$$

Se obtiene el vector de desplazamiento global:

$$Q = \begin{bmatrix} 0.0558 & -0.1860 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T [mm] \quad (93)$$

Los esfuerzos sobre los elementos son:

$$\sigma = \begin{bmatrix} 48.6080 & 13.3921 & 30.6573 \end{bmatrix}^T [kPa] \quad (94)$$

La deformación unitaria es:

$$\varepsilon = 10^6 \begin{bmatrix} 9.7216 & 2.6784 & 6.1315 \end{bmatrix}^T [mm/mm] \quad (95)$$

La fuerza en cada elemento es:

$$F = \begin{bmatrix} 12152 & 3348 & 7664 \end{bmatrix}^T [N] \quad (96)$$

Reacciones en los soportes (aplicable desde el nodo 2 al 4):

$$R = \begin{bmatrix} 0 & 0 & -7.2912 & 9.7216 & 2.6784 & 2.0088 & 4.6125 & 6.1210 \end{bmatrix}^T [kN] \quad (97)$$

3.2. Problema N°3

Basado en el desarrollo teórico presentado en la sección 2.5, se pudo obtener la función analítica que describe el radio del sólido en función de la profundidad y . Además, se obtuvieron las expresiones teóricas para los desplazamientos, deformaciones y esfuerzos. Estas expresiones fueron utilizadas en el código disponible en la sección 6.2.

Se presentan los resultados para los desplazamientos y esfuerzos para distintas discretizaciones, junto a la solución analítica.

Se utilizaron la cantidad de nodos igual a $N = 3, 5, 10, 50, 100, 200, 500, 1000$.

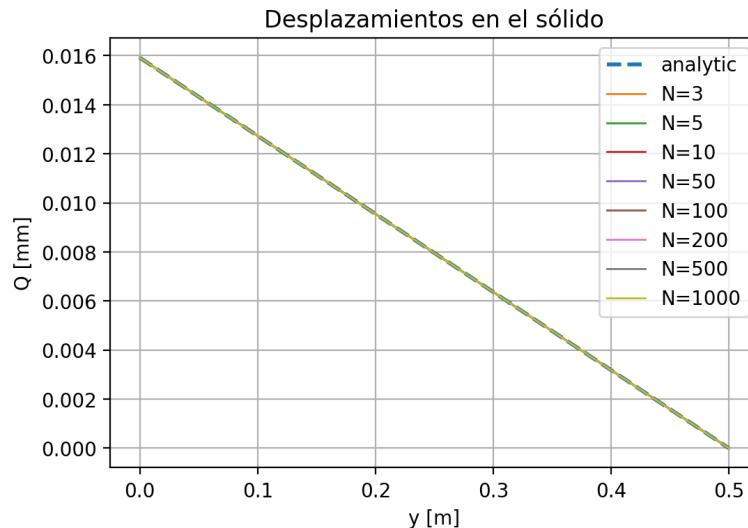


Figura 13: Desplazamientos en el sólido para distintas discretizaciones

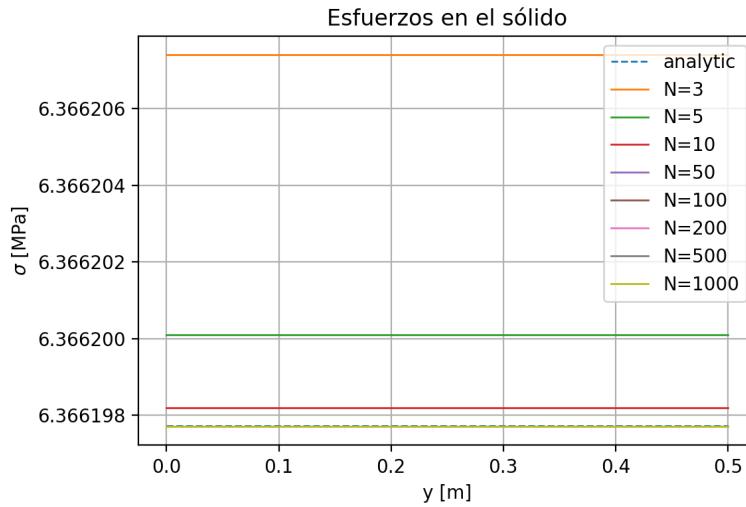


Figura 14: Esfuerzo en el sólido para distintas discretizaciones

Para evaluar la convergencia de la malla, se graficó el esfuerzo promedio en el sólido (el cual debería ser constante) respecto a la cantidad de nodos. Además, respecto a estos mismos valores de nodos para la malla, se graficó el error L2 de cada solución con la solución analítica. Estos resultados se muestran en las siguientes figuras:

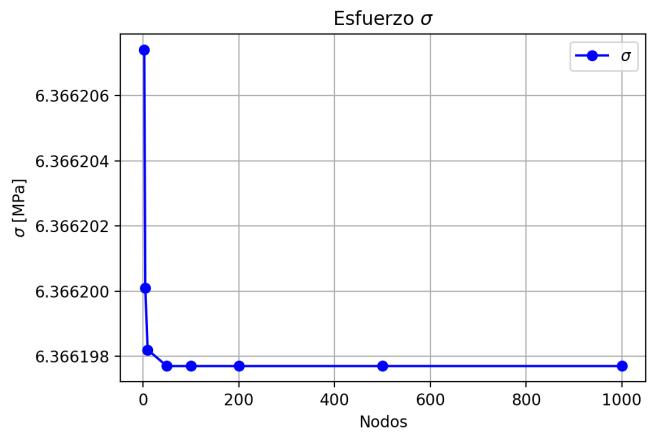


Figura 15: Esfuerzo en sólido respecto a la malla

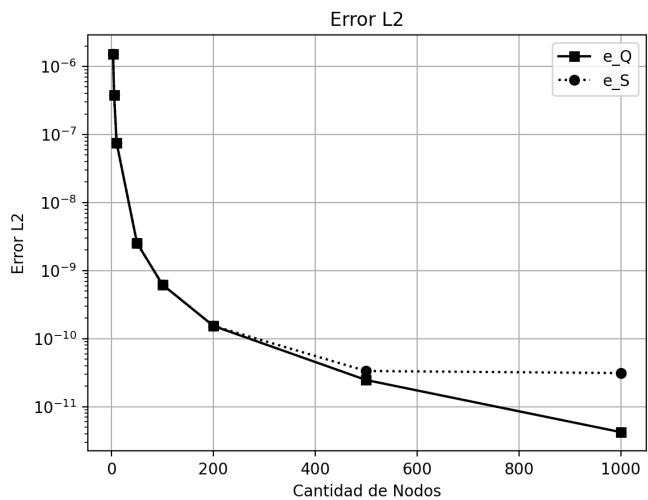


Figura 16: Error L2 para distintas discretizaciones

A continuación, se presentan los resultados para revisar la matriz de rigidez con la función “spy” de Matlab y la aplicación del algoritmo *symmetric reverse Cuthill-McKee ordering*.

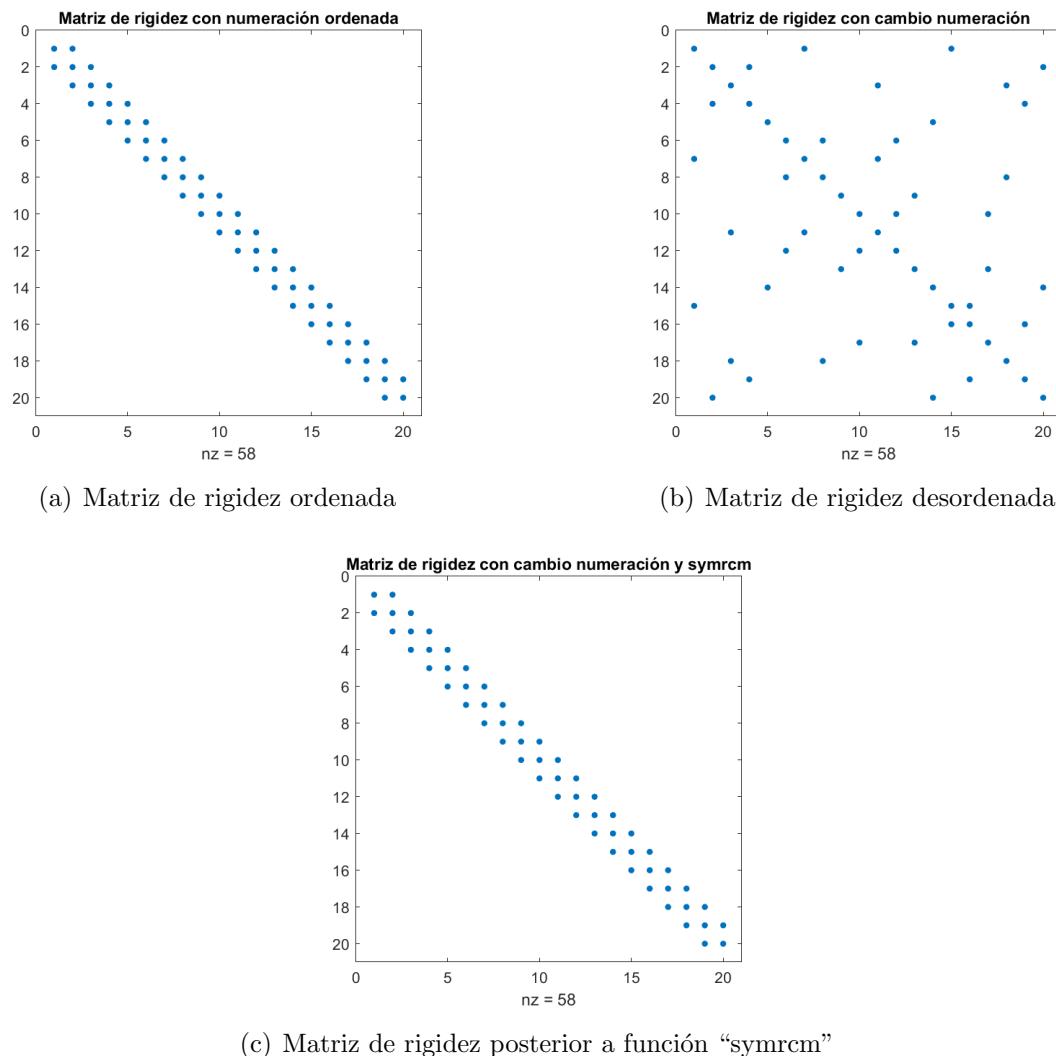


Figura 17: Matriz de rigidez de referencia para una malla de $N = 20$

Por último, se presenta una tabla con el tiempo de cálculo respecto al ordenamiento de los nodos.

Tabla 5: Tiempos de cálculo respecto a ordenamiento nodos

Nodos	Tiempo ordenado [s]	Tiempo desordenado [s]	Tiempo symrcm [s]
1000	0.043	0.065	0.064
5000	1.53	2.11	2.05
10000	15.34	17.3	17.9

3.3. Problema N°4

A continuación se presentan los resultados para los esfuerzos y desplazamientos en el sólido. El código programado se basa en la metodología propuesta en la sección 2.6. Principalmente, se mostrarán los resultados para mallas de $N = 115, 1615, 6317$.

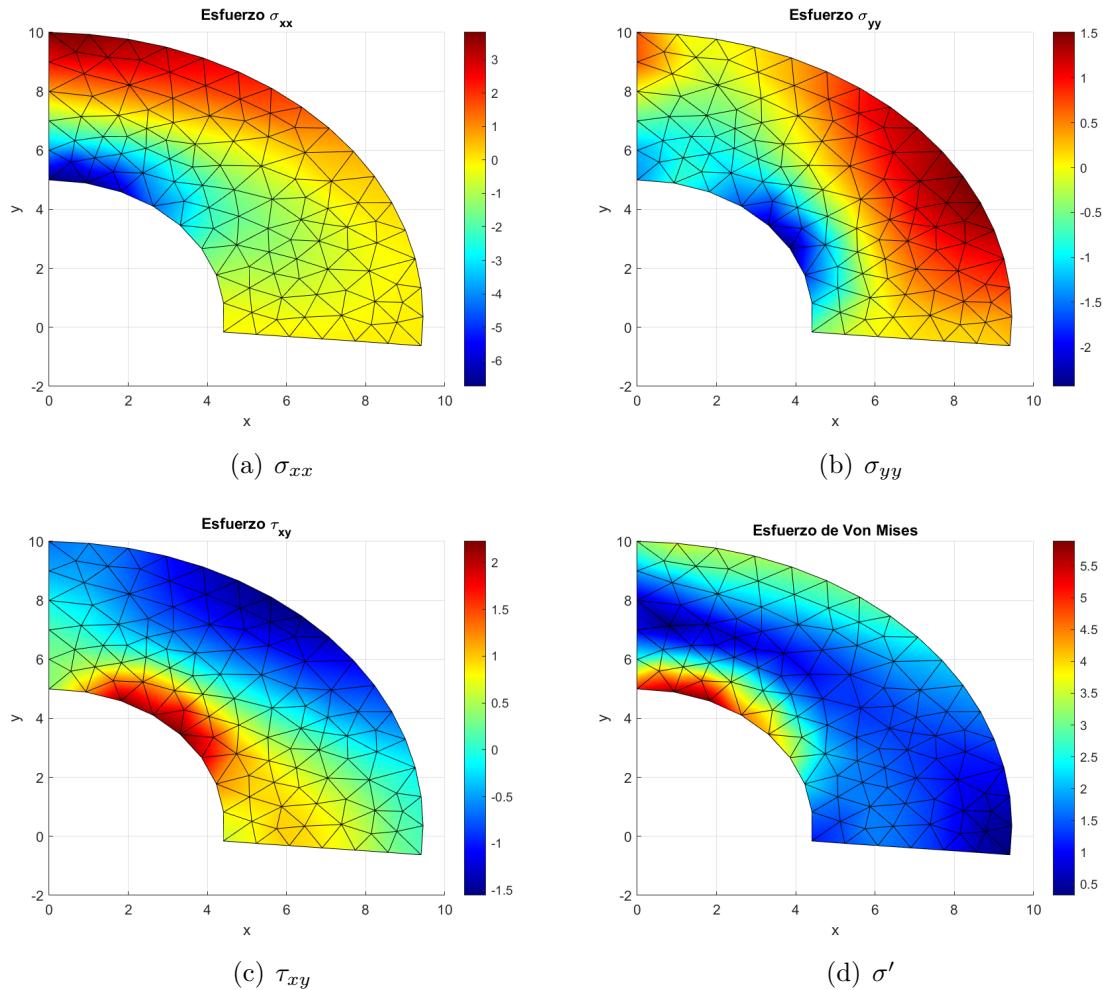


Figura 18: Esfuerzos para malla de $N=115$ nodos

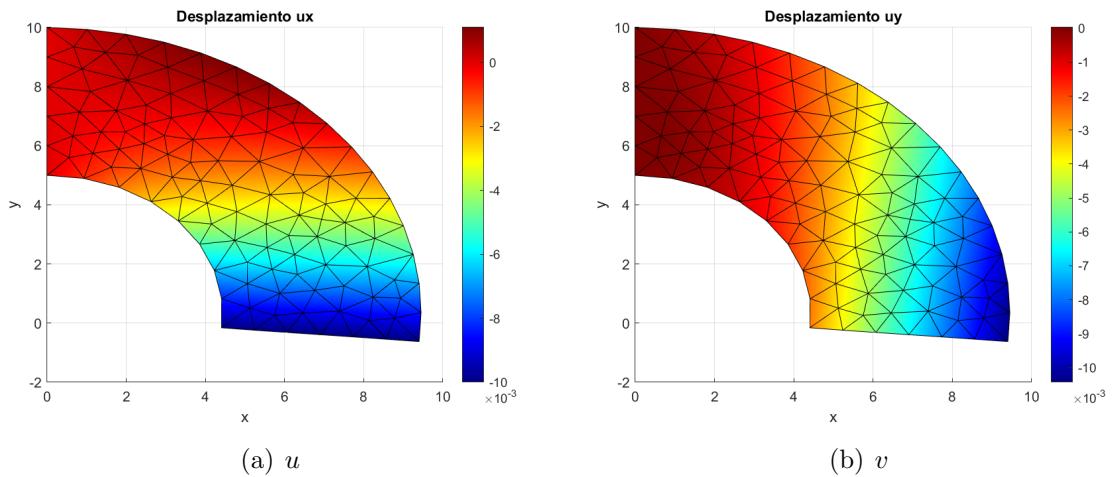


Figura 19: Desplazamientos para malla de $N=115$ nodos

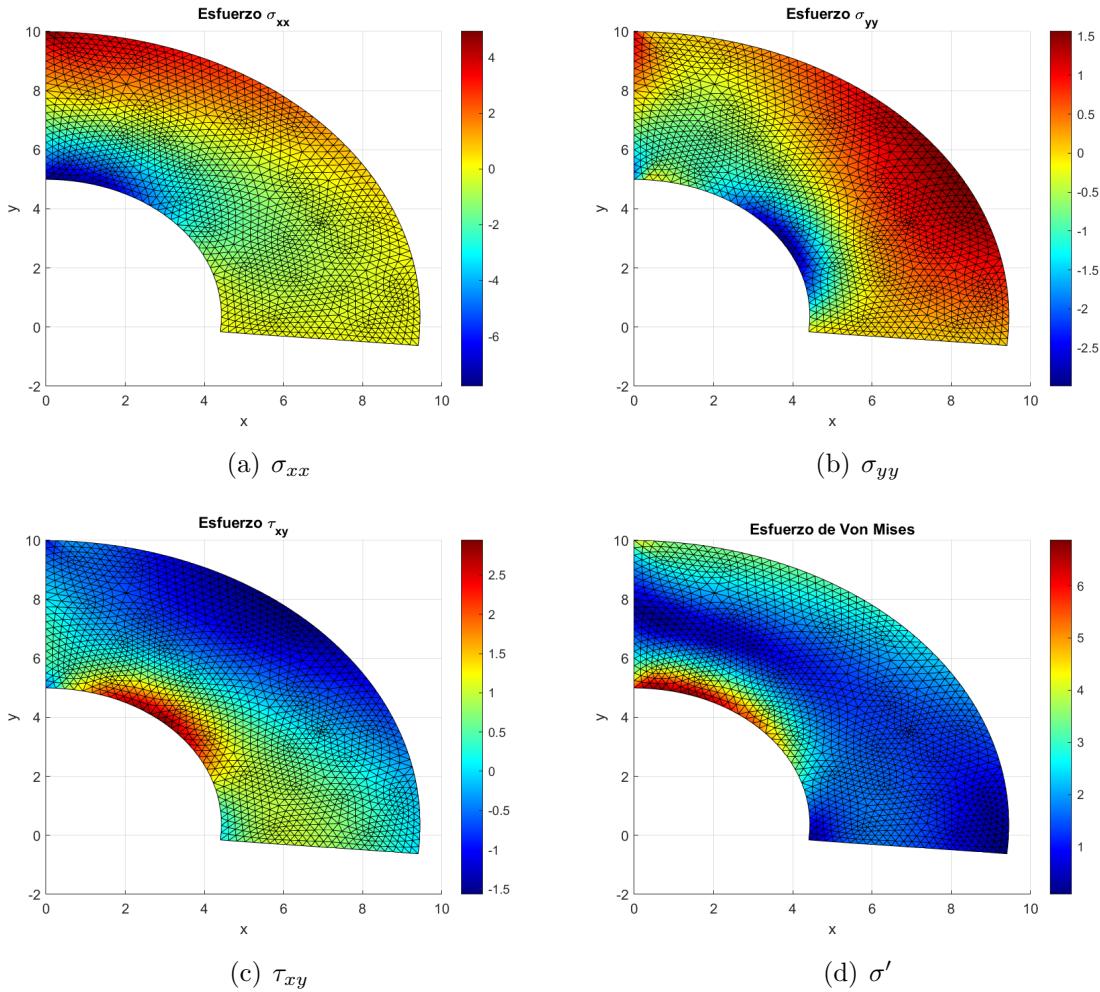


Figura 20: Esfuerzos para malla de N=1615 nodos

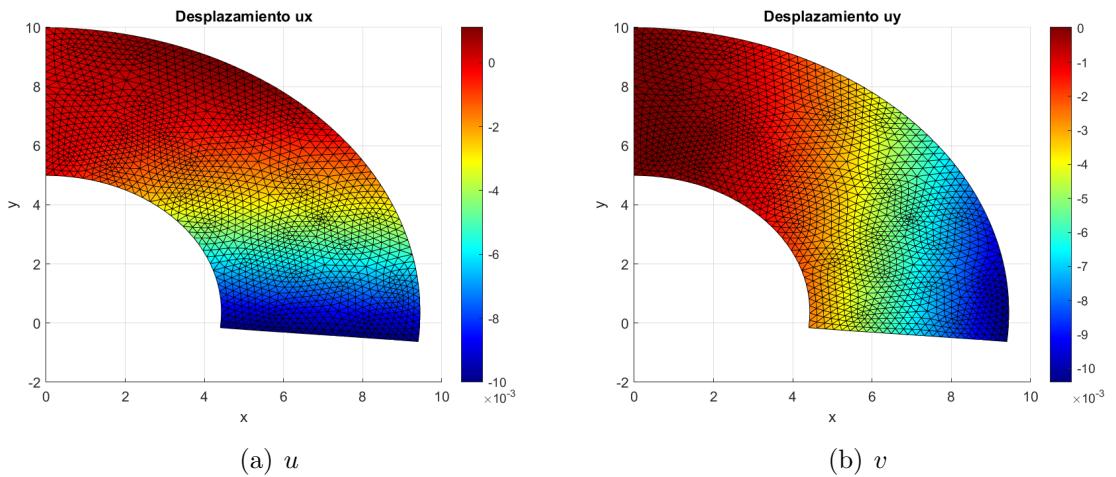


Figura 21: Desplazamientos para malla de $N=1615$ nodos

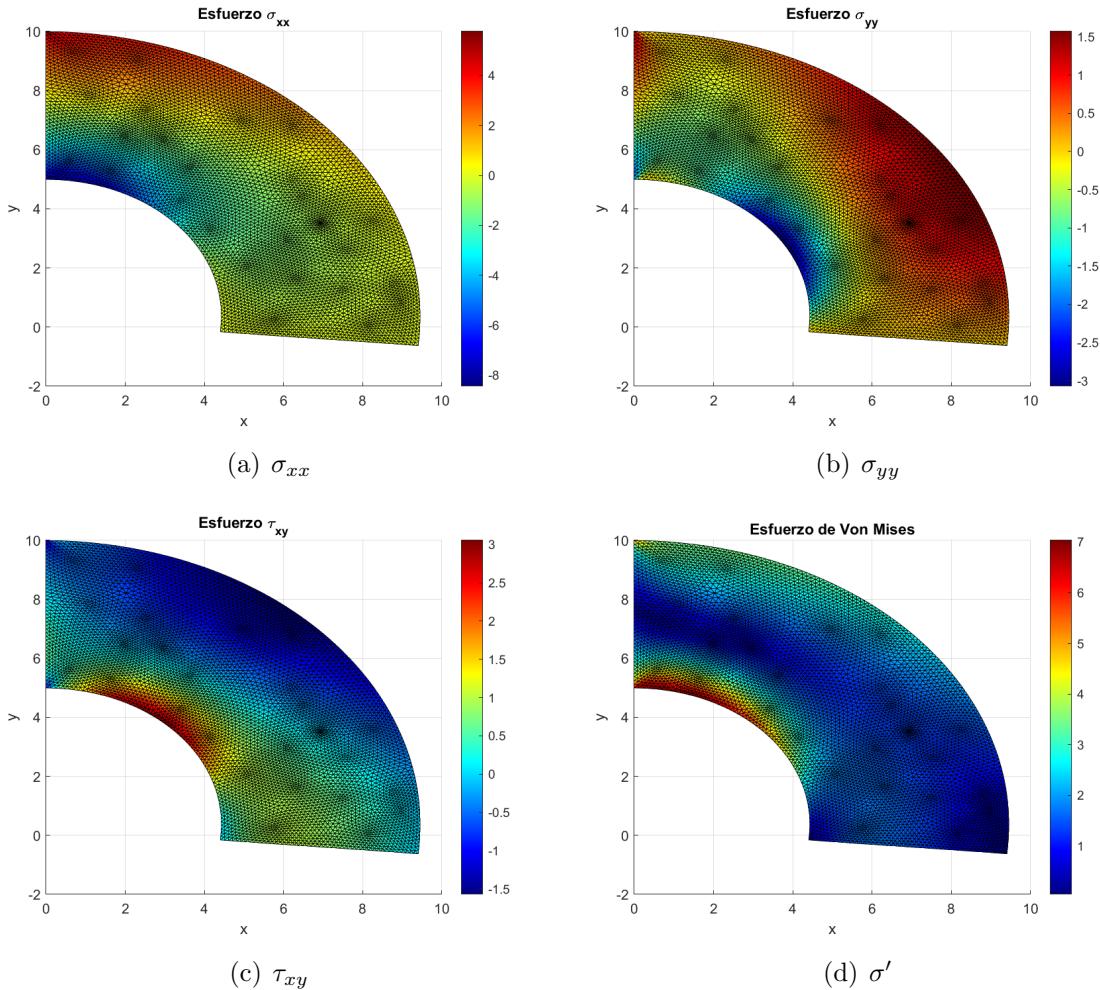


Figura 22: Esfuerzos para malla de $N=6317$ nodos

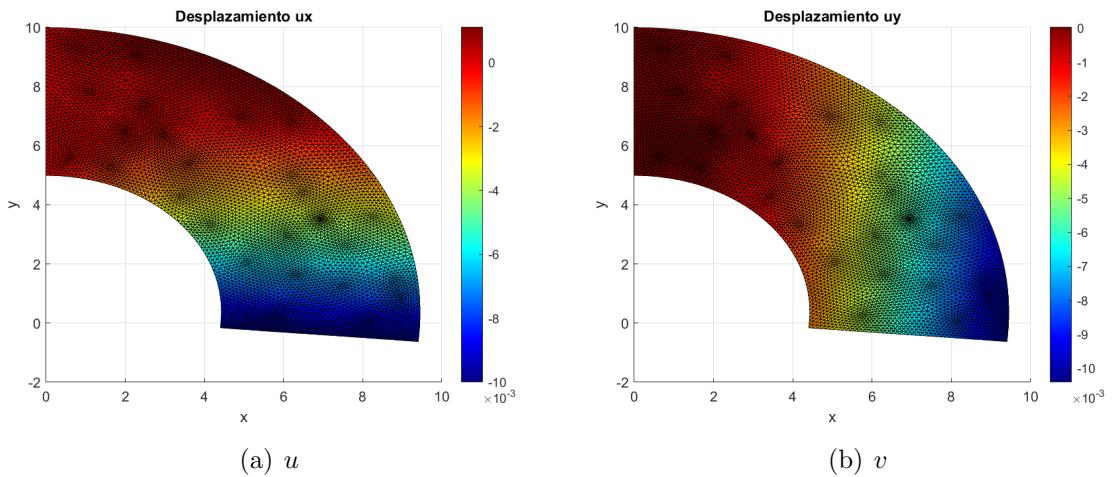


Figura 23: Desplazamientos para malla de $N=6317$ nodos

Se presenta el estudio de convergencia de malla realizado. Principalmente se evaluó el esfuerzo de Von Mises máximo, junto con el desplazamiento máximo en el sólido, respecto a las mallas empleadas.

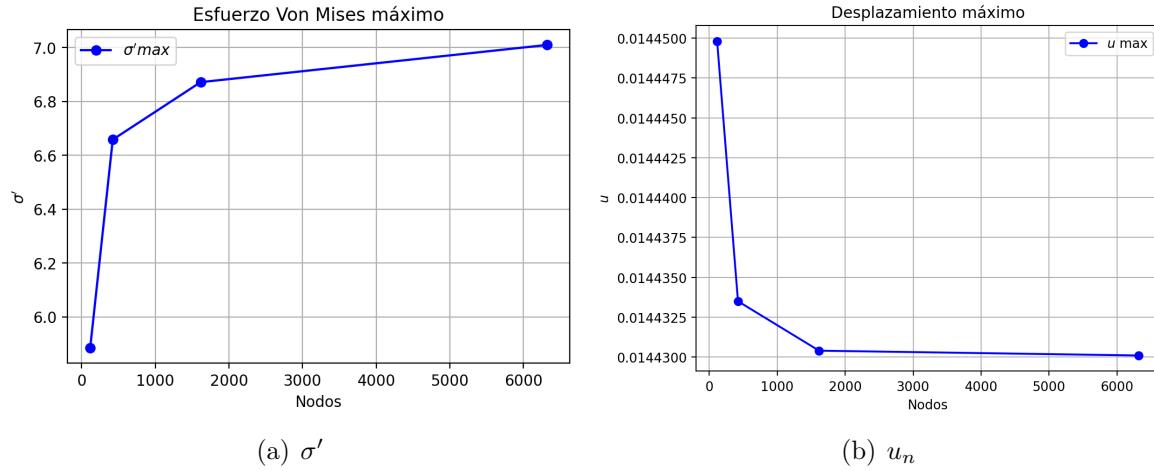


Figura 24: Convergencia de malla

Por último, se muestra el error L2 respecto a la solución analítica para el esfuerzo de Von Mises y el desplazamiento. Además, se agrega el error a “posteriori” de la malla. Todo lo anterior respecto a la cantidad de nodos de la malla.

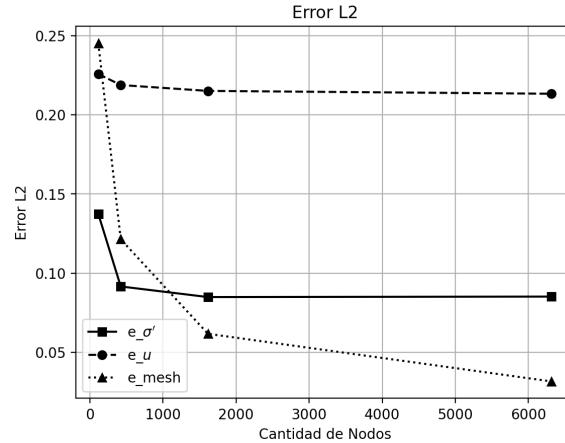


Figura 25: Error L2

A continuación, se presentan 2 tablas con los resultados y los errores en función de la malla utilizada.

Tabla 6: Resultados para distintas discretizaciones

Nodos	Von Mises máx	Desp. máx
115	5.884	14.450
422	6.685	14.434
1615	6.871	14.430
6317	7.028	14.430

Tabla 7: Errores para distintas discretizaciones

Nodos	L2 Von Mises	L2 Desp.	Error malla
115	0.137	0.226	0.245
422	0.089	0.219	0.120
1615	0.085	0.215	0.061
6317	0.085	0.213	0.031

Por último, se presentan los esquemas de la matriz de rigidez y su ordenamiento utilizando el algoritmo “symmetric reverse Cuthill-McKee ordering”.

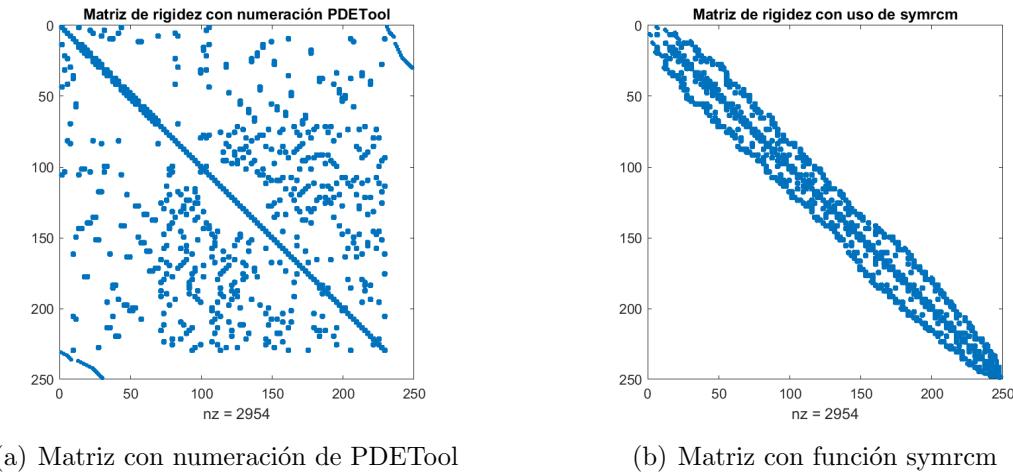


Figura 26: Esquemas de la matriz de rigidez global

Además, se presenta una tabla con el tiempo de cálculo respecto al ordenamiento de los nodos.

Tabla 8: Tiempos de cálculo respecto a ordenamiento nodos

Nodos	Tiempo desordenado [s]	Tiempo symrcm [s]
1615	0.0365	0.0360
6317	0.1499	0.1611

4. Análisis

4.1. Problema N°1 y N°2

En relación con los resultados obtenidos, se destaca que la armadura analizada genera un sistema de ecuaciones lineales con 8 variables, ya que debe cumplir con el equilibrio de fuerzas. La implementación del método de elementos finitos ha demostrado simplificar el cálculo, proporcionando una mayor precisión en los resultados, gracias a la aplicación del principio de Saint Venant, el cual postula que el esfuerzo en cada barra de un elemento es constante.

La decisión de unir el problema 1 y 2 se debe a la necesidad de la matriz de transformación en la segunda parte, la cual relaciona los resultados nodales con los globales. Este enfoque permitió obtener la matriz de rigidez global, posibilitando la determinación de desplazamientos, deformaciones, esfuerzos y fuerzas.

Como resultado, se obtuvieron dos desplazamientos globales y tres esfuerzos, los cuales permiten determinar las fuerzas y deformaciones mediante la relación fundamental del módulo de Young ($\sigma = E\varepsilon$). El desplazamiento total del nodo 1 alcanzó los 0.194 [mm] (0.0558 [mm] en dirección horizontal y -0.186 [mm] en dirección vertical). Además, se obtuvo que el elemento 1 es el con mayor requerimiento de esfuerzo, con un valor de 48.60 [kPa] y una fuerza de tensión de 12.15 [kN].

A modo de reflexión, la asignación de números a los elementos y nodos facilita la creación de sistemas de ecuaciones, los cuales pueden simplificarse mediante las condiciones de contorno correspondientes. Estos sistemas pueden ser programados para obtener resultados precisos y aplicable. Aunque el procedimiento empleado parece simple de aplicar, puede complicarse bastante para grandes armaduras con decenas o cientos de elementos. Sin embargo, las expresiones para cada elemento seguirán siendo las mismas, y la única variación será la gran matriz de rigidez ensamblada que se puede formar.

4.2. Problema N°3

En la sección 2.5 se obtuvieron los desplazamientos, deformaciones y esfuerzos teóricos. Por la naturaleza del problema (esfuerzo constante en todas las secciones del sólido), el radio de las secciones transversales del sólido sigue una función exponencial, dada en la ecuación 70. Esta expresión fue útil para poder encontrar las expresiones para la matriz de rigidez local y el vector de fuerzas de cada elemento. Notar que este procedimiento no fue trivial dado que se tuvo que integrar sobre cada

elemento debido a la variación de área con respecto a la profundidad.

Respecto a los resultados, se demuestra que el código de MEF programado modela con bastante precisión el problema. En la figura 13 se nota como todas las discretizaciones empleadas replican el comportamiento de la solución analítica (relación lineal decreciente). Incluso, para el ojo humano, esta diferencia es casi imprevisible. Lo interesante es que con una malla de 3 nodos, se replica ya la solución del problema.

De la misma forma, para los resultados obtenidos respecto al esfuerzo de compresión en el sólido, figura 14, todas las discretizaciones entregaron un esfuerzo constante, que está de acuerdo a la solución analítica. Sin embargo, se notas leves diferencias ya en la 5 cifra significativa respecto a la solución analítica. Dicho esto, las discretizaciones entregan un valor de esfuerzo de 6.336 [MPa].

Para evaluar la convergencia de la malla, figura 15, se graficó el esfuerzo en el sólido respecto al número de nodos de la malla. En este gráfico se nota como el valor del esfuerzo se estabiliza para la malla de 50 nodos y permanece constante hasta la malla de 1000 nodos. Lo anterior demuestra la no necesidad de una malla tan fina para llegar a la solución del problema. Sin embargo, en la figura 16 se nota como el error L2 en el esfuerzo y desplazamiento disminuyen en varios ordenes de magnitud al aumentar la cantidad de nodos. Con la malla de 1000 nodos se alcanza un error L2 de 2E-12 y 3E-11 para el desplazamiento y esfuerzo respectivamente. Es válido destacar que en muchos problemas, no se tiene a “priori” la solución analítica, función necesaria para la creación de esta gráfica. Es por esto que se deben realizar los estudios de convergencia como el de la figura 15 revisando hasta que cifra significativa se quiere tener independencia de la malla.

Por último, se evalúo el cambio de numeración en los nodos y el uso del algoritmo “symmetric reverse Cuthill-McKee ordering” para intentar formar una banda en la matriz de rigidez global. Para problemas 1D, como lo es este caso, es bastante sencillo formular un ordenamiento simple para los nodos. Al plantearlos ordenadamente, la matriz de rigidez global quedará tridiagonal. La ventaja de esta estructura es que se pueden utilizar algoritmos de resolución con menor costo computacional como lo es el algoritmo de Thomas. Sin embargo, al “desordenar” la numeración en los nodos, la matriz de rigidez pierde esta tridiagonal, y se vuelve desordenada, como se ve en la figura 17b. El uso del algoritmo “symmetric reverse Cuthill-McKee ordering” permite “acercar” los valores a la diagonal. Para este caso, se vuelve a la tridiagonal. La aplicación de este algoritmo es bastante útil en casos que se tenga una matriz bien dispersa producto de una numeración desordenada. Lo anterior permite por ejemplo utilizar algoritmos de resolución del sistema lineal con menor costo computacional. En la tabla 5 no se visualiza una gran diferencia en los tiempos de cálculo debido a que en todos los casos

se resolvió utilizando la factorización LU. Se espera ver grandes diferencias de tiempo si se hubiera utilizado el algoritmo de Thomas para los casos en que la matriz era tridiagonal.

4.3. Problema N°4

Para la viga circular en tensión plana, se adaptó el método de elementos finitos en 2D según lo desarrollado en la sección 2.6. Esta formulación fue programada en Matlab con la ayuda de los scripts proporcionados en [2]. Para la creación de la malla, se utilizó el software de PDETool, donde en base a un simple script, se adaptaron los archivos exportables para ser leídos por el programa principal. Se utilizaron distintas mallas de elementos triangulares.

Respecto a los resultados, se nota que el código elaborado logra replicar los distintos componentes de esfuerzos y desplazamientos en el sólido. Para una representación completa del estado de esfuerzo, se graficaron los esfuerzos normales en ambas direcciones, el esfuerzo cortante y el esfuerzo de Von Mises. Además, se graficaron los desplazamientos en la dirección horizontal y vertical. De las graficas, los resultados modelan bastante bien el estado de esfuerzos, e incluso, comparado con la solución analítica.

Para evaluar la convergencia de la malla, se graficó el esfuerzo de Von Mises máximo y el desplazamiento máximo en el sólido (figura 24). Se eligieron estas dos variables con el propósito de evaluar la convergencia de una variable primaria y secundaria. En estas gráficas se nota como se alcanza una cierta estabilización en la malla de $N=1615$ nodos. Esta estabilización es bien notoria para el desplazamiento máximo, que se estabiliza en 14.43. El esfuerzo de Von Mises no se logra estabilizar del todo en el estudio, mostrando un crecimiento a medida que se refina la malla. La explicación que se puede tener para este fenómeno es que quizás las mallas empleadas no son lo suficientemente “buenas” para replicar el problema. Se debería haber refinado localmente la parte cóncava del sólido en donde ocurre la concentración de esfuerzos. Se espera que con esta mejora se pueda ver una convergencia de la malla a medida que se aumenta la cantidad de nodos. Esto no pudo ser implementado debido a que no es parte del software de PDETool. De todas formas, se estima que el esfuerzo de Von Mises máximo está cercano a 7.02.

Por otra parte, en la figura 25 se puede apreciar la evolución del error L2 y el error a “posteriori” de la malla respecto al número de nodos y la solución analítica. Se nota que el error L2 para el esfuerzo de Von Mises y el desplazamiento se estabiliza en los $N=1615$ nodos. Además, se nota que el error L2 es bastante alto para ambos casos (0.085 y 0.213 respectivamente). Esto se puede deber al mismo

fenómeno ya explicado del refinamiento. El software empleado no fue capaz de refinar localmente las zonas en donde ocurre la concentración de esfuerzos, y por ende, replica una solución alejada de la realidad. De todas formas, el error a “posteriori” de la malla sigue decreciendo con la refinación, llegando a un valor de 0.031.

Por último, se estudió el uso del algoritmo “symmetric reverse Cuthill-McKee ordering” en la resolución del sistema lineal de ecuaciones ensambladas. De manera preliminar, al tratarse de un problema 2D, la numeración entregada por PDETool no genera un matriz tridiagonal como la creada en el problema N°3. Se nota en la figura 26 que la distribución de las entradas de la matriz de rigidez son bastante desordenadas, pero siempre manteniendo la simetría. La utilización del algoritmo “symrcm” permitió ordenar la matriz en torno a la diagonal. Pese a no formar una matriz tridiagonal, se podría considerar como una matriz tridiagonal por bloques. Lo anterior permite utilizar modificaciones del algoritmo de Thomas que reducen el costo computacional del problema. Al igual que en el problema N°3, ambos sistemas se resolvieron utilizando el método LU, por lo que no se aprecian diferencias de tiempo considerables en la tabla 8.

5. Conclusiones

A modo de conclusión, el método de elementos finitos pudo ser aplicado en las 3 situaciones planteadas. En todos los casos se obtuvo resultados con significado físico, que concuerdan y se aproximan a las solución teóricas.

En el caso de la armadura (problema N°1 y N°2), se formuló un sistema de ecuaciones lineales con 8 variables, resuelto mediante el principio de Saint Venant. La obtención de la matriz de rigidez global facilitó la determinación precisa de desplazamientos, deformaciones, esfuerzos y fuerzas para los nodos y elementos, logrando incluso replicar el comportamiento de la solución analítica.

Por otra parte, para el sólido de revolución en compresión (problema N°3), se obtuvieron resultados con errores de 2E-12 y 3E-11 para el desplazamiento y esfuerzo respectivamente (malla fina). Esto demuestra una buena implementación del método. Además, lo interesante de este problema es que con mallas bastante gruesas (3 nodos), ya la solución se approxima bastante bien a la analítica. En cuanto a la convergencia de la malla, se observa que no es necesario contar con una malla extremadamente fina para alcanzar la solución del problema. Sin embargo, la reducción del error L2 en esfuerzo y desplazamiento al aumentar la cantidad de nodos sugiere que el refinamiento local podría mejorar la reproducción del comportamiento del problema, dependiendo de las cifras significativas que uno requiera.

En el análisis de la viga circular en tensión plana (problema N°4), al aplicar el método de elementos finitos en 2D, se obtuvieron resultados que replicaban de manera satisfactoria los diferentes componentes de esfuerzos y desplazamientos en el sólido. Sin embargo, la falta de refinamiento local en áreas críticas afectó la convergencia de la malla, resaltando la necesidad de mejorar la representación de estas zonas para obtener resultados más precisos. De todas formas, se cumplió el objetivo propuesto. Este problema logró reflejar la no trivialidad de la elección de la malla, dado su efecto que tiene en la convergencia del método.

En cuanto al uso del algoritmo "simétrico inverso de Cuthill-McKee ordering", en los distintos problemas, se observa que, aunque no se logra formar en todos los casos una matriz tridiagonal, su aplicación podría beneficiar la eficiencia computacional en problemas con matrices desordenadas.

Para finalizar, lo realizado en este documento demuestra la aplicabilidad del método de elementos finitos a problemas de mecánica de sólidos en 1D y 2D. Se espera con esta base, lograr generar el conocimiento requerido para poder resolver problemas en 3D.

Referencias

- [1] SP Timoshenko y JN Goodier. “Theory of Elasticity, McGraw-Hill, New York, 1970”. En: *Fok-Ching Chong received the BS degree from the Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan, in (1971)*.
- [2] J. Alberty, C. Carstensen, S.A. Funken, y R. Klose. “Matlab Implementation of the Finite Element Method in Elasticity”. En: *Computing* 69 (2002), págs. 239-263.
- [3] Jonas Koko. “Vectorized Matlab codes for linear two-dimensional elasticity”. En: *Scientific Programming* 15.3 (2007), págs. 157-172.
- [4] J. Alberty, C. Carstensen y S.A. Funken. “Remarks around 50 lines of Matlab: short finite element implementation”. En: *Numerical Algorithms* 20 (1999), págs. 117-137.
- [5] Junuthula Narasimha Reddy. *An introduction to the finite element method*. Vol. 3. McGraw-Hill New York, 2013.
- [6] Junuthula Narasimha Reddy. *An introduction to continuum mechanics*. Cambridge university press, 2013.
- [7] Martin H Sadd. *Elasticity: theory, applications, and numerics*. Academic Press, 2009.
- [8] Tirupathi R. Chandrupatla, Ashok D. Belegundu, T. Ramesh y Chaitali Ray. *Introduction to finite elements in engineering*. Vol. 10. Prentice Hall Upper Saddle River, NJ, 2002.
- [9] Steven C Chapra. *Numerical methods for engineers*. Mcgraw-hill, 2010.
- [10] Kenneth Franklin Riley, Michael Paul Hobson y Stephen John Bence. *Mathematical methods for physics and engineering*. 1999.

6. Anexos

6.1. Problema N°1 y N°2

Código 1: Código principal problema N°1 y N°2

```
1 %Parametros:%
2 Area=250;
3 E=200e3;
4 matrizL=ones(4);
5 MatrizK=zeros(8);
6
7 %Elemento 1%
8 V1=[1,2]; % nodo 1 a 2%
9 Ang1=126.87; %Angulo respecto a x%
10 Le1=750;
11 Klocal1=((E*Area)/Le1)/1e5;
12 [ML1,l1,m1]=Matriztrans(Ang1,matrizL);
13 K1=Klocal1*ML1;
14 KT1=transfglobal(MatrizK,V1,K1); %Integracion de matrix 4x4 a 8x8%
15 disp(['Para elemento 1: l=', num2str(l1), ' y m=', num2str(m1)]);
16
17
18 %Elemento 2%
19 V2=[1,3]; %nodo 1 a 3%
20 Ang2=36.87; %Angulo respecto a x%
21 Le2=1000;
22 Klocal2=((E*Area)/Le2)/1e5;
23 [ML2,l2,m2]=Matriztrans(Ang2,matrizL);
24 K2=Klocal2*ML2;
25 KT2=transfglobal(MatrizK,V2,K2);%Integracion de matrix 4x4 a 8x8%
26 disp(['Para elemento 2: l=', num2str(l2), ' y m=', num2str(m2)]);
27
28
29 %Elemento
30 V3=[1,4]; %nodo 1 a 4;
31 Ang3=53; %Angulo respecto a x
32 13;
33 Le3=750;
34 Klocal3=((E*Area)/Le3)/1e5;
```

```

35 [ML3 ,l3 ,m3 ]=Matriztrans(Ang3 ,matrizL);
36 K3=Klocal3*ML3;
37 KT3=transfglobal(MatrizK ,V3 ,K3); %Integraci n de matrix 4x4 a 8x8%
38 disp(['Para elemento 3: l=', num2str(l3), ' y m=', num2str(m3)]);
39
40 %Matriz de rigidez global%
41 KTransf=(KT1+KT2+KT3);
42
43
44 %Resolviendo sistema de ecuaciones reducido:%
45 A=(1e5)*[0.8 ,0.24;0.24 ,1.04];
46 B=[0 ;-18000];
47 Q=inv(A)*B;
48 disp(['Desplazamiento global: Q1=', num2str(Q(1)), ' y Q2=', num2str(Q(2))]);
49
50 %Calculo de esfuerzos:%
51 Sigma1=(E/Le1)*[-11 ,-m1 ,11 ,m1]*[0.0558 ;-0.1860 ;0;0];
52 Sigma2=(E/Le2)*[-12 ,-m2 ,12 ,m2]*[0.0558 ;-0.1860 ;0;0];
53 Sigma3=(E/Le3)*[-13 ,-m3 ,13 ,m3]*[0.0558 ;-0.1860 ;0;0];
54 disp(['Esfuerzo en los elemento: Sigma1=', num2str(Sigma1), ', Sigma2=',
      num2str(Sigma2), ' y Sigma3=', num2str(Sigma3)]);
55
56 %Deformaci n unitaria%
57 e1=Sigma1*E;
58 e2=Sigma2*E;
59 e3=Sigma3*E;
60 disp(['Deformaci n unitaria: e1=', num2str(e1), ', e2=', num2str(e2),
      ' y e3=', num2str(e3)]);
61
62 %Fuerza en cada elemento:%
63 F1=Sigma1*Area;
64 F2=Sigma2*Area;
65 F3=Sigma3*Area;
66 disp(['Fuerza por elemento: F1=', num2str(F1), ', F2=', num2str(F2),
      ' y F3=', num2str(F3)]);
67
68 %reacciones%
69 C=[0.0558 ;-0.1860 ;0;0;0;0;0;0];
70 FF=[0 ;-18000;0;0;0;0;0;0];
71 R_prev=((1e5)*KTransf)*C-FF;

```

```

72 | Reacciones=R_prev(3:8)
73 |
74 | function [ML, ll, mm] =Matriztrans(Ang, matrizL)
75 |     L=matrizL;
76 |     l=cosd(Ang);
77 |     m=sind(Ang);
78 |     for i=1:4
79 |         if i==1
80 |             L(i,1)=l^2;
81 |             L(i,2)=l*m;
82 |             L(i,3)=-l^2;
83 |             L(i,4)=-l*m;
84 |         elseif i==2
85 |             L(i,1)=l*m;
86 |             L(i,2)=m^2;
87 |             L(i,3)=-l*m;
88 |             L(i,4)=-m^2 ;
89 |         elseif i==3
90 |             L(i,1:4)=-L(1,1:4);
91 |         elseif i==4
92 |             L(i,1:4)=-L(2,1:4);
93 |         end
94 |     end
95 |     ML=L;
96 |     ll=l;
97 |     mm=m;
98 | end
99 |
100 | function Trans=transfglobal(MatrizK, V, K)
101 |     MM=MatrizK;
102 |     i=V(1);
103 |     j=V(2);
104 |     if i==1 & j==2
105 |         MM(1:2,1:2)=K(1:2,1:2);
106 |         MM(1:2,3:4)=K(1:2,3:4);
107 |         MM(3:4,1:2)=K(3:4,1:2);
108 |         MM(3:4,3:4)=K(3:4,3:4);
109 |     elseif i==1 & j==3
110 |         MM(1:2,1:2)=K(1:2,1:2);
111 |         MM(1:2,5:6)=K(1:2,3:4);
112 |         MM(5:6,1:2)=K(3:4,1:2);

```

```
113 |     MM(5:6,5:6)=K(3:4,3:4);  
114 | elseif i==1 & j==4  
115 |     MM(1:2,1:2)=K(1:2,1:2);  
116 |     MM(1:2,7:8)=K(1:2,3:4);  
117 |     MM(7:8,1:2)=K(3:4,1:2);  
118 |     MM(7:8,7:8)=K(3:4,3:4);  
119 | end  
120 | Trans=MM;  
121 | end
```

6.2. Problema N°3

Código 2: Código principal problema N°3

```
1 format compact
2 clc
3
4 % Inputs del problema %
5 P = 50e3;
6 L = 500e-3;
7 r_0 = 50e-3;
8
9 % Propiedades del material %
10 rho = 7850;
11 g = 9.81;
12 gamma = rho*g;
13 E = 200e9;
14
15 A_0 = pi*r_0^2;
16 sigma_0 = P/A_0;
17
18 prop = [rho, gamma, E];
19 prob = [P, L, r_0, A_0, sigma_0];
20
21 % Discretizacion %
22 N = 10000;
23 y = linspace(0, L, N);
24
25 % Solucion teorica %
26 delta_teo = (P/(E*A_0))*L - (P/(E*A_0))*y;
27 def_teo = -sigma_0/E*ones(1, N-1);
28 sigma_teo = def_teo*E;
29
30 % Calculos %
31 A = Area(y, prop, prob);
32 r = Radius(y, prop, prob);
33
34 H = [1 -1; -1 1];
35 le = y(2)-y(1);
36
37 % Matriz de conectividad %
```

```

38 | desordenar = false;
39 | ordenar = false;
40 |
41 | Nodos = 1:N;
42 | if desordenar
43 |     Nodos = Nodos(randperm(N));
44 | end
45 |
46 | M = zeros(N-1,2);
47 | for i=1:N-1
48 |     M(i,1) = Nodos(i);
49 |     M(i,2) = Nodos(i+1);
50 | end
51 |
52 |
53 | % Matriz de rigidez K %
54 | K = zeros(N,N);
55 | for k=1:N-1
56 |     ke = (1/le^2)*E*H*sigma_0/gamma*(Area(y(k+1),prop,prob)-Area(y(k),
57 |                                         prop,prob));
58 |     for i=1:2
59 |         for j=1:2
60 |             I = M(k,i);
61 |             J = M(k,j);
62 |             K(I,J) = K(I,J) + ke(i,j);
63 |         end
64 |     end
65 | end
66 | % Vector de fuerzas F %
67 | F = zeros(N,1);
68 | for k=1:N-1
69 |     dA_e = sigma_0/gamma*(Area(y(k+1),prop,prob)-Area(y(k),prop,prob));
70 |     dyA_e = (sigma_0/gamma)^2*((gamma/sigma_0*y(k+1)-1)*Area(y(k+1),
71 |                                         prop,prob)-(gamma/sigma_0*y(k)-1)*Area(y(k),prop,prob));
72 |     fe_1 = -gamma/le*(y(k)*dA_e - dyA_e);
73 |     fe_2 = gamma/le*(y(k+1)*dA_e - dyA_e);
74 |     I = M(k,1);
75 |     J = M(k,2);
76 |     F(I) = F(I) + fe_1;
    F(J) = F(J) + fe_2;

```

```

77 | end
78 | F(M(1,1)) = F(M(1,1)) + P;
79 |
80 |
81 | % ordenar seg n algoritmo symrcm %
82 | if ordenar
83 |     r = symrcm(K);
84 |     K = K(r,r);
85 |     F = F(r);
86 | end
87 |
88 | % Eliminar ultimo nodo %
89 | K(M(end,2),:) = [];
90 | K(:,M(end,2)) = [];
91 | F(M(end,2)) = [];
92 |
93 | % Resolucion sistema lineal %
94 | Q = zeros(1,N-1);
95 | Q = linsolve(K,F);
96 |
97 | % Agregar nodo eliminado %
98 | Q = [Q(1:M(end,2)-1); 0; Q(M(end,2):end)];
99 |
100 | % Calculo de deformaciones y esfuerzos %
101 | epsilon = zeros(1,N-1);
102 | B = 1/le*[-1 1];
103 | for k=1:N-1
104 |     q = [Q(M(k,1)) ; Q(M(k,2))];
105 |     epsilon(k) = B*q;
106 | end
107 | sigma = E*epsilon;
108 |
109 | % Calculo de errores %
110 | eQ = calc_L2_error(Q,delta_teo);
111 | eS = calc_L2_error(sigma,sigma_teo);
112 |
113 | save_files(sigma,Q,N);
114 |
115 |
116 | function A = Area(y,prop,prob)
117 |     r = num2cell(prop);

```

```

118 |     [~,gamma,~] = deal(r{:});
119 |     r = num2cell(prob);
120 |     [~,~,~,A_0,sigma_0] = deal(r{:});
121 |     A = A_0*exp(gamma/sigma_0*y);
122 | end
123 |
124 function R = Radius(y,prop,prob)
125     r = num2cell(prop);
126     [~,gamma,~] = deal(r{:});
127     r = num2cell(prob);
128     [~,~,r_0,~,sigma_0] = deal(r{:});
129     R = r_0*exp(gamma/(2*sigma_0)*y);
130 | end
131 |
132 function [error] = calc_L2_error(u_aprox,u_teo)
133     dif_u = (u_aprox - u_teo);
134     error = sqrt(sum(dif_u.^2)/sum(u_teo.^2));
135 | end
136 |
137 function save_files(sigma,Q,N)
138     directory = 'results';
139     filename = fullfile(directory, sprintf('Sigma_%d.dat', N));
140     save(filename,"sigma", "-ascii")
141
142     filename = fullfile(directory, sprintf('Q_%d.dat', N));
143     save(filename,"Q", "-ascii")
144 | end

```

6.3. Problema N°4

Código 3: Código principal problema N°4

```
1 format compact
2 clc
3
4 % Propiedades %
5 E = 1e4; nu = 0.25;
6 mu = E/ (2* (1+nu)) ; lambda = E*nu/((1+nu) * (1-2*nu)) ;
7
8 % Cargar malla, se debe correr codigo save_variables.m %
9 load vars/coordinates.dat;
10 eval("load vars/elements3.dat;", "elements3 = [];");
11 eval("load vars/elements4.dat;", "elements4 = [];");
12 eval("load vars/neumann.dat;", "neumann = []");
13 load vars/dirichlet.dat;
14
15 % Matriz de rigidez %
16 K = sparse(2*size(coordinates, 1),2*size(coordinates, 1));
17 for j = 1:size(elements3, 1)
18     I = 2*elements3(j, [1 1 2 2 3 3]) -[1 0 1 0 1 0];
19     K(I,I) = K(I,I) + stima3(coordinates(elements3(j,:)), :, lambda, mu)
20         ;
21 end
22 F = zeros(2*size(coordinates, 1), 1);
23
24 % Fuerzas de cuerpo %
25 for j = 1:size (elements3, 1)
26     I = 2*elements3(j,[1 1 2 2 3 3]) - [1 0 1 0 1 0];
27     fs = f(sum(coordinates(elements3(j,:)),:))/3)';
28     F(I) = F(I) + det([1, 1, 1 ; coordinates(elements3(j,:)),:]')*[fs;
29         fs;fs] /6;
30 end
31 % Condiciones de neumann %
32 if ~isempty (neumann)
33     n = (coordinates (neumann (: ,2) , :) -coordinates (neumann (:, 1)
34         ,:))* [0, -1;1,0];
35     for j = 1: size (neumann, 1)
```

```

35     I = 2*neumann (j, [1,1,2,2]) - [1,0,1,0];
36     gm = g (sum(coordinates (neumann (j,:), :)) /2, n(j, :) /norm
37         (n(j,:)));
38     F(I) = F(I) + norm(n(j, :)) * [gm; gm] /2;
39 end
40
41 % Condiciones de dirichlet %
42 DirichletNodes = unique(dirichlet);
43 [W,M] = u_d(coordinates(DirichletNodes, :));
44 B = sparse(size(W, 1) ,2*size(coordinates, 1));
45 for k = 0:1
46     for l = 0:1
47         B(1+l:2:size(M, 1) ,2*DirichletNodes-1+k) = diag(M(1+l:2:size(M,
48             1), 1+k));
49     end
50 end
51 mask = find(sum(abs(B')));
52 K = [K, B(mask,:)' ; B(mask,:), sparse(length(mask), length(mask))];
53 F = [F; W(mask, :)];
54
55 % Ordenar segun symrcm %
56 ordenar = false;
57 if ordenar
58     r = symrcm(K);
59     K = K(r,r);
60     F = F(r);
61 end
62
63 % Resolver sistema lineal %
64 x = K \ F;
65
66 % Devolver numeracion si se reordeno con symrcm %
67 if ordenar
68     [~,r_2] = sort(r);
69     x = x(r_2);
70 end
71
72 % Obtener desplazamientos %
73 u = x(1:2*size(coordinates,1));

```

```

74 % Calcular esfuerzos y errores %
75 [AvE, Eps3, Eps4, AvS, Sigma3, Sigma4] = avmatrix(coordinates,elements3
76 ,elements4,u,lambda,mu);
77 [Sigma_principal,von_mises,AvC] = extra_stresses(AvS,lambda,mu);
78 [L2_error_u, L2_error_von_mises, mesh_error] = calculate_errors(
79 coordinates, elements3, elements4, AvE, Eps3, Eps4, AvS, Sigma3,
80 Sigma4, u, lambda, mu);
81 % Graficar solucion %
82 plot_solutions(elements3, elements4, coordinates, AvS, Sigma_principal,
83 von_mises, AvC, u, lambda, mu);
84 % Imprimir resultados %
85 disp("Numero de nodos")
86 disp(size(coordinates,1))
87 disp("Von mises max")
88 disp( max(von_mises))
89 disp("un max")
90 disp(max(sqrt(u(1:2:end,1).^2 + u(2:2:end,1).^2))*10^3)
91
92 % Condicion de borde de dirichlet %
93 function [W,M] = u_d(x)
94     M = zeros(2*size(x,1),2);
95     W = zeros(2*size(x,1),1);
96     M(1:2:end,1) = 1;
97     temp = find(x(:,1)<1e-9);
98     M(2*temp,2) = 1;
99     value = u_value(x);
100    W(1:2:end,1)= value(:,1);
101    W(2*temp,1) = value(temp,2);
102 end
103
104 % Fuerzas de cuerpo %
105 function volforce = f(x)
106     volforce = zeros(size(x,1),2);
107 end
108
109 % Fuerzas de superficie %
110 function sforce = g(x,n)

```

```

111     sforce = zeros(size(x,1),2);
112 end
113
114 % Matriz rigidez local %
115 function stima3=stima3(vertices, lambda, mu)
116     PhiGrad = [1, 1, 1; vertices']\ [zeros(1,2) ; eye(2)];
117     R = zeros (3,6);
118     R([1,3], [1,3,5]) = PhiGrad';
119     R([3,2], [2,4,6]) = PhiGrad';
120     C = mu*[2,0,0;0,2, 0; 0, 0, 1] + lambda* [1, 1,0;1, 1, 0;0,0,0];
121     stima3 = det([1,1,1;vertices'])/2*R'*C*R;
122 end
123
124
125 % Solucion teorica para desplazamientos %
126 function value = u_value(x)
127     [phi,r] = cart2pol(x(:,1),x(:,2));
128     E = 1e4; nu = 0.25;
129     mu = E/ (2* (1+nu)) ; lambda = E*nu/((1+nu) * (1-2*nu)) ;
130     a = 5; b = 10; u0 = -0.01;
131     ab = a^2 + b^2;
132     PN = -u0*E/(pi*ab);
133
134     K = PN/E*( 0.5*(1-3*nu)*a^2 - b^2*(1+nu)/2 - ab*(1-nu)*log(a) );
135     ur = PN/E*( (0.5*(1-3*nu)*r.^2 - a^2*b^2*(1+nu)/(2*r.^2) - ab*(1-nu)
136         )*log(r)).*sin(phi) + ab*(2*phi-pi).*cos(phi) - K*sin(phi) ;
137     ut = -PN/E*( (0.5*(5+nu)*r.^2 - a^2*b^2*(1+nu)/(2*r.^2) + ab*(1-nu)
138         *log(r) + (1+nu) ).*cos(phi) + ab*(2*phi-pi).*sin(phi) - K*cos(
139             phi) ;
140     value = [ur.*cos(phi)-ut.*sin(phi), ur.*sin(phi)+ut.*cos(phi)];
141 end
142
143
144 % Solucion teorica para esfuerzos %
145 function sig_cartesian = analytic_Sig(x)
146     [phi,r] = cart2pol(x(:,1),x(:,2));
147     E = 1e4; nu = 0.25;
148     mu = E/ (2* (1+nu)) ; lambda = E*nu/((1+nu) * (1-2*nu)) ;
149     a = 5; b = 10; u0 = -0.01;
150     ab = a^2 + b^2;
151     PN = -u0*E/(pi*ab);

```

```

149
150    srr = PN*(r + a^2*b^2./r.^3 - ab./r).*sin(phi);
151    stt = PN*(3*r - a^2*b^2./r.^3 - ab./r).*sin(phi);
152    str = -PN*(r + a^2*b^2./r.^3 - ab./r).*cos(phi);
153    Sig_polar = [srr str str stt];
154
155    cos_phi = cos(phi);
156    sin_phi = sin(phi);
157    Q = [cos_phi -sin_phi sin_phi cos_phi];
158
159    SQ = zeros(size(r,1),4);
160    SQ(:,1) = Sig_polar(:,1).*Q(:,1)+Sig_polar(:,2).*Q(:,3);
161    SQ(:,2) = Sig_polar(:,1).*Q(:,2)+Sig_polar(:,2).*Q(:,4);
162    SQ(:,3) = Sig_polar(:,3).*Q(:,1)+Sig_polar(:,4).*Q(:,3);
163    SQ(:,4) = Sig_polar(:,3).*Q(:,2)+Sig_polar(:,4).*Q(:,4);
164    SC = zeros(size(r,1),4);
165    SC(:,1) = Q(:,1).*SQ(:,1)+Q(:,3).*SQ(:,3);
166    SC(:,2) = Q(:,1).*SQ(:,2)+Q(:,3).*SQ(:,4);
167    SC(:,3) = Q(:,2).*SQ(:,1)+Q(:,4).*SQ(:,3);
168    SC(:,4) = Q(:,2).*SQ(:,2)+Q(:,4).*SQ(:,4);
169
170    sig_cartesian = SC;
171 end
172
173
174 % Calculo de esfuerzos y deformaciones %
175 function [AvE, Eps3, Eps4, AvS, Sigma3, Sigma4] = avmatrix(coordinates,
176 elements3, elements4, u, lambda, mu)
177 Eps3 = zeros(size(elements3, 1), 4);
178 Sigma3 = zeros(size(elements3, 1), 4);
179 Eps4 = zeros(size(elements4, 1), 4);
180 Sigma4 = zeros(size(elements4, 1), 4);
181 AreaOmega = zeros(size(coordinates, 1), 1);
182 AvS = zeros(size(coordinates, 1), 4);
183 AvE = zeros(size(coordinates, 1), 4);
184 for j = 1:size(elements3, 1)
185     area3 = det ([1,1,1;coordinates(elements3(j,:),:)']);
186     AreaOmega(elements3(j,:)) = AreaOmega (elements3 (j,:)) +area3;
187     PhiGrad = [1, 1,1; coordinates(elements3(j,:), :)'];
188     PhiGrad = [zeros
189                 (1,2) ; eye(2)];
190     U_Grad = u([1;1]*2*elements3(j,:)-[1;0]*[1, 1, 1])*PhiGrad;

```

```

188 Eps3(j,:) = reshape((U_Grad+U_Grad')/2, 1,4);
189 Sigma3(j,:) = reshape(lambda*trace(U_Grad)*eye(2) +2*mu*(U_Grad+U_Grad')/2,1,4) ;
190 AvE(elements3(j,:),:) = AvE(elements3(j,:),:) +area3*[1;1;1]*Eps3(j,:);
191 AvS(elements3(j,:),:) = AvS(elements3(j,:), :) +area3*[1;1;1]*Sigma3(j,:);
192 end
193 AvE = AvE./ (AreaOmega*[1, 1, 1, 1]);
194 AvS = AvS./ (AreaOmega*[1,1,1,1]);
195 end
196
197 % Calculo de esfuerzos principales y von mises %
198 function [Sigma_principal,von_mises,AvC] = extra_stresses(AvS,lambda,mu)
199 n = size(AvS);
200 sum_Sig = AvS(:,1) + AvS(:,4);
201 dif_Sig = AvS(:,1) - AvS(:,4);
202 Sigma_principal = zeros(size(AvS, 1),3);
203 Sigma_principal(:,1) = 1/2*sum_Sig + sqrt(1/4*dif_Sig.^2 + AvS(:,2).^2);
204 Sigma_principal(:,2) = 1/2*sum_Sig - sqrt(1/4*dif_Sig.^2 + AvS(:,2).^2);
205 Sigma_principal(:,3) = lambda/(2*(mu+lambda))*sum_Sig;
206 von_mises = sqrt( (Sigma_principal(:,1)-Sigma_principal(:,2)).^2 + (Sigma_principal(:,2)-Sigma_principal(:,3)).^2 + (Sigma_principal(:,1)-Sigma_principal(:,3)).^2)/sqrt(2);
207 AvC=(mu/ (24* (mu+lambda)^2)+1/(8*mu))*(AvS(:, 1) + AvS(:,4)).^2+1/(2*mu)*(AvS(:,2).^2-AvS(:,1).*AvS(:,4)) ;
208 end
209
210
211 % Calculo de errores %
212 function [L2_error_u, L2_error_von_mises, mesh_error] =
213 calculate_errors(coordinates, elements3, elements4, AvE, Eps3, Eps4,
214 AvS, Sigma3, Sigma4, u, lambda, mu)
215 ux = u(1:2:end,1);
216 uy = u(2:2:end,1);
217 u_n = sqrt(ux.^2 + uy.^2)*10^3;
218 u_teo = u_value(coordinates);
219 ux_teo = u_teo(:,1);

```

```

218    uy_teo = u_teo(:,2);
219    u_n_teo = sqrt(ux_teo.^2 + uy_teo.^2)*10^3;
220    L2_error_u = calc_L2_error(u_n,u_n_teo);
221
222    sig_teo = analytic_Sig(coordinates);
223    [~,von_mises_teo,AvC_teo] = extra_stresses(sig_teo,lambda,mu);
224    [~,von_mises,AvC] = extra_stresses(AvS,lambda,mu);
225    L2_error_von_mises = calc_L2_error(von_mises,von_mises_teo);
226
227    mesh_error = aposteriori(coordinates, elements3, elements4, AvE,
228                               Eps3, Eps4, AvS, Sigma3, Sigma4, u, lambda, mu);
228 end
229
230 function [error] = calc_L2_error(u_aprox,u_teo)
231     dif_u = (u_aprox - u_teo);
232     error = sqrt(sum(dif_u.^2)/sum(u_teo.^2));
233 end
234
235 % Calculo del error de la malla %
236 function estimate=aposteriori(coordinates, elements3, elements4,AvE,
237                                 Eps3, Eps4, AvS, Sigma3, Sigma4, u, lambda, mu);
237     eta3 = zeros(size(elements3,1),1) ;
238     eta4 = zeros(size(elements4,1),1);
239     e3 = zeros(size(elements3,1),1);
240     e4 = zeros(size (elements4,1), 1);
241     for j=1:size(elements3,1)
242         Area3=det([1,1,1;coordinates(elements3(j,:)),:]')/2;
243         for i=1:4
244             eta3(j) = eta3(j) + Area3 * (Sigma3(j,i)*Eps3(j,i) + AvS(
245                 elements3(j,:),i)'*[2,1,1;1,2,1;1,1,2]*AvE(elements3(j
246                 ,:),i)/12 - AvS(elements3(j,:),i)'*[1;1;1]*Eps3(j,i)/3
247                 - AvE(elements3 (j,:) ,i)'*[1;1;1]*Sigma3 (j,i) /3) ;
248             e3(j) = e3(j) + Area3 * AvS(elements3(j,:),i)
249                 *[2,1,1;1,2,1;1,1,2]*AvE(elements3(j,:),i)/12;
250         end
251     end
252     estimate = sqrt (sum (eta3))/ sqrt (sum(e3));
253 end
254
255 % Funcion para graficar solido deformado %

```

```

253 | function show(elements3, elements4, coordinates, X, u, lambda, mu,
254 |   titulo)
255 |   factor=60;
256 |   colormap("jet")
257 |   trisurf(elements3, factor*u(1:2:size(u, 1))+coordinates(:, 1),
258 |     factor*u(2:2:size(u, 1))+coordinates(:, 2), zeros(size(
259 |       coordinates, 1), 1), X, 'facecolor', 'interp');
260 |   hold on
261 |   view (0,90)
262 |   hold off
263 |   colorbar("vert")
264 |   xlabel('x')
265 |   ylabel('y')
266 |   title(titulo)
267 | end
268 |
269 | % Creacion de graficos %
270 | function plot_solutions(elements3, elements4, coordinates, AvS,
271 |   sigma_principal, von_mises, AvC, u, lambda, mu)
272 |   n = size(coordinates,1);
273 |   ux = u(1:2:end,1);
274 |   uy = u(2:2:end,1);
275 |   sx = AvS(:,1);
276 |   sy = AvS(:,4);
277 |   txy = AvS(:,2);
278 |
279 |   var_plot = von_mises;
280 |   titulo = 'Esfuerzo de Von Mises';
281 |   show(elements3, elements4, coordinates, var_plot, u, lambda, mu,
282 |     titulo);
283 |   filename = fullfile('plots', sprintf('von_%d.png', n));
284 |   saveas(gcf,filename)
285 |
286 |   var_plot = ux;
287 |   titulo = 'Desplazamiento ux';

```

```

288 show(elements3, elements4, coordinates, var_plot, u, lambda, mu,
      titulo);
289 filename = fullfile('plots', sprintf('uy_%d.png', n));
290 saveas(gcf, filename)
291
292 var_plot = sx;
293 titulo = 'Esfuerzo \sigma_{xx}';
294 show(elements3, elements4, coordinates, var_plot, u, lambda, mu,
      titulo);
295 filename = fullfile('plots', sprintf('sx_%d.png', n));
296 saveas(gcf, filename)
297 var_plot = sy;
298 titulo = 'Esfuerzo \sigma_{yy}';
299 show(elements3, elements4, coordinates, var_plot, u, lambda, mu,
      titulo);
300 filename = fullfile('plots', sprintf('sy_%d.png', n));
301 saveas(gcf, filename)
302
303 var_plot = txy;
304 titulo = 'Esfuerzo \tau_{xy}';
305 show(elements3, elements4, coordinates, var_plot, u, lambda, mu,
      titulo);
306 filename = fullfile('plots', sprintf('txy_%d.png', n));
307 saveas(gcf, filename)
308 end

```

Código 4: Código para exportar malla PDETool

```

1 format compact
2
3 % Cargar y guardar malla de PDETool %
4 save('vars/vars.mat','p','t','e')
5 load('vars/vars.mat','p','t','e')
6
7 % Vertices %
8 cords = p';
9 save vars/coordinates.dat cords -ascii
10
11 % Elementos %
12 el3 = t(1:3,:);
13 save vars/elements3.dat el3 -ascii
14

```

```
15 % Condiciones borde dirichlet %
16 e2 = e';
17 de = e2(e2(:, 5) == 1 | e2(:, 5) == 2, 1:2);
18 dirich = de;
19 save vars/dirichlet.dat dirich -ascii
20
21 % Condiciones borde neumann %
22 ne = e2(e2(:, 5) == 3 | e2(:, 5) == 4, 1:2);
23 neum = ne;
24 save vars/neumann.dat neum -ascii
```