Método de Elementos Finitos

Julio A. Medina
Universidad de San Carlos
Escuela de Ciencias Físicas y Matemáticas
Maestría en Física
julioantonio.medina@gmail.com

Resumen

En este reporte se hace una introducción teórica al Método de Elementos Finitos para resolver ecuaciones diferenciales parciales con condiciones en la frontera. También se incluyen implementaciones relativamente rápidas para resolver ecuaciones diferenciales parabólicas, hiperbólicas y elípticas. Se discuten algunos detalles de como funciona el proceso de teselación o discretización del dominio y se hace una comparación de los paquetes computacionales de `Mathematica` y `Python` para abordar estos problemas.

1. Generalidades del Método de Elementos Finitos

Hay varios acercamientos teóricos para abordar el método de elementos finitos en la introducción que se da en este documento se utiliza el acercamiento funcional, donde el objetivo es minimizar un funcional que se construye a partir del problema especifico a resolver, este el método presentado en [1], análogo al método de Ritz.
También se incluye una breve reseña de cómo abordar el problema por medio de lo que se conoce como formulación débil. Aquí se parte de la ecuación diferencial parcial y se quiere obtener una integral que cumple ciertos requerimientos para una función de prueba, este formalismo es necesario para resolver problemas utilizando la librería de `Python`, `FEniCS`. Vale mencionar que estos dos acercamientos se basan en el cálculo de variaciones.

2. Introducción al Método de Elementos Finitos

Este método para aproximar las soluciones de ecuaciones diferenciales parciales fue originalmente desarrollado para su uso en problemas de ingeniería civil pero en la actualidad su uso es ubicuo para aproximar soluciones en todas las áreas de la matemática aplicada y en muchas aplicaciones de la física. Su uso es intensivo en el modelado de sistemas complejos para el diseño aerodinámico de piezas y dispositivos de exploración espacial así como también en aplicaciones avanzadas de la ingeniería como las competiciones automovilísticas en el caso de la Formula 1.

Una de las ventajas del método de elementos finitos sobre el método de diferencias finitas es la relativa facilidad con la que el método de elementos

finitos maneja las condiciones de frontera. Muchos problemas físicos(pragmáticos) tienen condiciones de frontera que involucran derivadas y contornos de formas irregulares, sin simetrías aparentes. Condiciones de frontera de este tipo son difíciles de manejar con los métodos de diferencias finitas ya que cada condición de frontera que involucra a alguna derivada tiene que aproximarse con un cociente de diferencias en los puntos del retículo o malla, pero el hecho que se tienen bordes irregulares hace que la definición de un retículo adecuado sea poco trivial. El método de diferencias finitas por el otro lado incluye a las condiciones de frontera como integrales en un funcional que debe minimizarse, de esta manera la construcción del procedimiento del método de elementos finitos es independiente de las condiciones particulares de frontera del problema.

La ecuación diferencial parcial a atacar es la siguiente

$$\frac{\partial}{\partial x}\left(p(x,y)\frac{\partial u}{\partial x}\right)+\frac{\partial}{\partial y}\left(q(x,y)\frac{\partial u}{\partial y}\right)+r(x,y)u(x,y)=f(x,y) \tag{1}$$

con $(x,y)\in\mathcal{D}$, donde $\mathcal{D}$ es una región plana con frontera $\mathcal{S}$.

Condiciones de frontera de la forma

$$u(x,y)=g(x,y) \tag{2}$$

se imponen en una porción $\mathcal{S}_1$ de la frontera. En el resto de la frontera, $\mathcal{S}_2$, se requiere que la solución satisfaga

$$p(x,y)\frac{\partial u}{\partial x}(x,y)\cos\theta_1+q(x,y)\frac{\partial u}{\partial y}(x,y)\cos\theta_2+g_1(x,y)u(x,y)=g_2(x,y), \tag{3}$$

donde θ_1 y θ_2 son los ángulos de dirección de las normales salientes en el punto (x,y), ver figura 1 Algunos problemas en física en áreas como mecánica de sólidos y elasticidad tienen ecuaciones diferenciales parciales asociadas similares a 1. La solución a este problemas de este tipo típicamente minimizan cierto funcional, que involucra integrales definidas, sobre una clase de funciones determinada por el problema.

Al suponer que p,q,r y f son todas continuas en $\mathcal{D}\cup\mathcal{S}$, p y q tiene primeras derivadas parciales continuas, y g_1 y g_2 son continuas en $\mathcal{S}_2$. Adicionalmente se supone que $p(x,y)>0$, $q(x,y)>0$, $r(x,y)\leq 0$ y $g_1(x,y)>0$. Entonces la solución de 1 minimiza el funcional $I[w]$ únicamente, i.e. está es la única función que minimiza al funcional.

$$I[w]=\int\int_{\mathcal{D}}\left\{\frac{1}{2}\Big[p(x,y)\Big(\frac{\partial w}{\partial x}\Big)^2+q(x,y)\Big(\frac{\partial w}{\partial y}\Big)^2-r(x,y)w^2\Big]+f(x,y)w\right\}dxdy + \int_{\mathcal{S}_2}\left\{-g_2(x,y)w+\frac{1}{2}g_1(x,y)w^2\right\}dS \tag{4}$$

sobre todas las funciones dos veces diferenciables y continuas w que satisfacen la ecuación 2 en $\mathcal{S}_1$. El método de elementos finitos aproxima está solución al minimizar al funcional I 4 sobre una clase mas pequeña de funciones, similar al acercamiento utilizado el método de Rayleigh-Ritz(ver Burden. [1]).

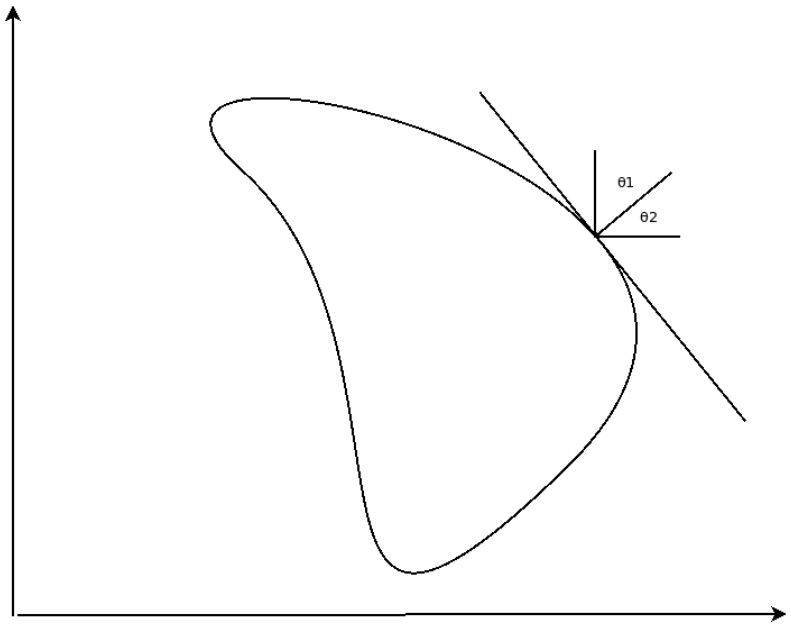

Figura 1: Visualización ángulos θ_1 y θ_2

2.1. Definiendo los elementos finitos

El primer paso es de definir los elementos finitos para construir la aproximación, esto consiste en dividir la región en un número finito de secciones o elementos con una forma regular, pueden ser triángulos, rectángulos, o cualquier figura regular que "tesele.º discretice la región por completo(*tiling the region*).

El conjunto de funciones regularmente usadas para la aproximación es un conjunto de polinomios por partes de grado fijo en x y y, y la aproximación requiere que los polinomios sean unidos de tal manera que la función resultante sea continua con primera y segunda derivadas integrables sobre la región entera. polinomios de tipo linear en x y y

$$\phi(x, y) = a + bx + cy \tag{5}$$

se utilizan comúnmente con elementos triangulares, mientras que polinomios de tipo bilinear en x y y

$$\phi(x, y) = a + bx + cy + dxy \tag{6}$$

están asociados al uso de elementos rectangulares.

Suponiendo que la región $\mathcal{D}$ ha sido subdividida en elemento triangulares. El conjunto de triángulos se denota como D, y los vértices de dichos triángulos se llaman nodos. El método busca una aproximación de la forma

$$\phi(x, y) = \sum_{i=1}^{m} \gamma_i \phi_i(x, y), \tag{7}$$

donde $\phi_1, \phi_2, \ldots, \phi_m$ son polinomios lineales por partes linealmente independientes, y las $\gamma_1, \gamma_2, \ldots, \gamma_m$ son constantes. Algunas de estas constantes, por ejemplo $\gamma_{n+1}, \gamma_{n+2}, \ldots, \gamma_m$ se utilizan para asegurar que la condición de frontera

$$\phi(x, y) = g(x, y) \tag{8}$$

se satisfaga en $\mathcal{S}_1$, las constantes restantes $\gamma_1, \gamma_2, \ldots, \gamma_n$, se utilizan para minimizar el funcional $I[\sum_{i=1}^{m} \gamma_i \phi_i]$. Usando la forma de 8 para la aproximación de w en la expresión 4 se obtiene

$$\begin{aligned} I[\phi] &= I\Big[\sum_{i=1}^{m} \gamma_i \phi_i\Big] \\ &= \int\int_{\mathcal{D}} \left(\frac{1}{2} \left\{ p(x,y) \Big[\sum_{i=1}^{m} \gamma_i \frac{\partial \phi_i}{\partial x}(x,y) \Big]^2 + q(x,y) \Big[\sum_{i=1}^{m} \gamma_i \frac{\partial \phi_i}{\partial y}(x,y) \Big]^2 \right. \right. \\ &\left. \left. - r(x,y) \Big[\sum_{i=1}^{m} \gamma_i \phi_i(x,y) \Big]^2 \right\} + f(x,y) \sum_{i=1}^{m} \gamma_i \phi_i(x,y) \right) dy\, dx \\ &+ \int_{\mathcal{S}_2} \left\{ - g_2(x,y) \sum_{i=1}^{m} \gamma_i \phi_i(x,y) + \frac{1}{2} g_1(x,y) \Big[\sum_{i=1}^{m} \gamma_i \phi_i(x,y) \Big]^2 \right\} dS \end{aligned} \tag{9}$$

Como se menciono anteriormente se utiliza $\gamma_i, \gamma_2, \ldots, \gamma_n$ para minimizar el funcional, con esto al considerar a I como una función de $\gamma_i, \gamma_2, \ldots, \gamma_n$, i.e. $I[\gamma_i, \gamma_2, \ldots, \gamma_n]$. Para que obtener un mínimo se tiene que cumplir

$$\frac{dI}{d\gamma_j} = 0, \ \forall\, j = 1, 2, \ldots, n. \tag{10}$$

Diferenciando 9 con respecto γ_i se obtiene

$$\begin{aligned} \frac{\partial I}{\partial \gamma_i} = \int\int_{\mathcal{D}} & \left\{ p(x,y) \sum_{i=1}^{m} \gamma_i \frac{\partial \phi_i}{\partial x}(x,y) \frac{\partial \phi_j}{\partial x}(x,y) \right. \\ & + q(x,y) \sum_{i=1}^{m} \gamma_i \frac{\partial \phi_i}{\partial y}(x,y) \frac{\partial \phi_j}{\partial y}(x,y) \\ & \left. - r(x,y) \sum_{i=1}^{m} \gamma_i \phi_i(x,y) \phi_j(x,y) + f(x,y) \phi_j(x,y) \right\} dx\, dy \\ & + \int_{\mathcal{S}_2} \left\{ - g_2(x,y) \phi_j(x,y) + g_1(x,y) \sum_{i=1}^{m} \gamma_i \phi_i(x,y) \phi_j(x,y) \right\} dS, \end{aligned} \tag{11}$$

con esto la condición para minimizar el funcional se convierte en

$$\begin{aligned}0 = \sum_{i=1}^{m} \Bigg[\int\int_{\mathcal{D}} \Bigg\{ & p(x,y)\gamma_i \frac{\partial \phi_i}{\partial x}(x,y)\frac{\partial \phi_j}{\partial x}(x,y) + q(x,y)\gamma_i \frac{\partial \phi_i}{\partial y}(x,y)\frac{\partial \phi_j}{\partial y}(x,y) \\ & - r(x,y)\phi_i(x,y)\phi_j(x,y) \Bigg\} dx\, dy \\ & + \int_{\mathcal{S}_2} g_1(x,y)\phi_i(x,y)\phi_j(x,y) dS \Bigg] \gamma_i \\ & + \int\int_{\mathcal{D}} f(x,y)\phi_j(x,y) dx\, dy - \int_{\mathcal{S}_2} g_2(x,y)\phi_j(x,y) dS.\end{aligned} \tag{12}$$

para cada $i = 1, 2, \ldots, n$. Este conjunto de ecuaciones puede expresarse con una ecuación matricial

$$\mathbf{A}\vec{c} = \vec{b}, \tag{13}$$

donde $\vec{c} = (\gamma_1, \ldots, \gamma_n)^t$. Con $\mathbf{A} = (\alpha_{ij})$ y $\vec{b} = (\beta_1, \ldots, \beta_n)^t$ son definidos a continuación

$$\begin{aligned}\alpha_{ij} = \int\int_{\mathcal{D}} \Bigg[& p(x,y)\frac{\partial \phi_i}{\partial x}(x,y)\frac{\partial \phi_j}{\partial x}(x,y) + q(x,y)\frac{\partial \phi_i}{\partial y}(x,y)\frac{\partial \phi_j}{\partial y}(x,y) \\ & - r(x,y)\phi_i(x,y)\phi_j(x,y) \Bigg] dx\, dy + \int_{\mathcal{S}_2} g_1(x,y)\phi_i(x,y)\phi_j(x,y) dS,\end{aligned} \tag{14}$$

para cada $i = 1, 2, \ldots, n$ y $j = 1, 2, \ldots, m$, y

$$\beta_i = -\int\int_{\mathcal{D}} f(x,y)\phi_i(x,y) dx\, dy + \int_{\mathcal{S}} g_2(x,y)\phi_i(x,y) dS - \sum_{k=n+1}^{m} \alpha_{ik}\gamma_k. \tag{15}$$

para cada $i = 1, \ldots, n$.

La elección del conjunto de funciones base es de bastante importancia ya que la elección apropiada puede frecuentemente hacer que la matriz $\mathbf{A}$ sea positiva definida y tenga la forma de una matriz banda(*banded matrix*). Para el problema de segundo orden 1, se asume que $\mathcal{D}$ es poligonal, de tal manera que $\mathcal{D} = D$, y que $\mathcal{S}$ es un conjunto de lineas rectas contiguas.

2.2. Triangulando la región

Para empezar el procedimiento delineado anteriormente es crucial el paso de subdividir la región D en un conjunto de triángulos $T_1, T_2, \ldots, T_M$ donde el triángulo i(i-esimo) tiene tres vértices o nodos que se denotan

$$V_j^{(i)} = \left(x_j^{(i)}, y_j^{(i)}\right), \text{ para } j = 1, 2, 3. \tag{16}$$

Para simplificar la notación se escribe $V_j^{(i)} = \left(x_j^{(i)}, y_j^{(i)}\right)$ cuando se trabaja con el triángulo fijo T_i. En cada vértice V_j hay un polinomio lineal asociado

$$N_j^{(i)}(x,y) \equiv N_j(x,y) = a_j + b_j x + c_j y, \text{ donde } N_j^{(i)}(x_k, y_k) = \begin{cases} 1, & j = k \\ 0, & j \neq k \end{cases} \tag{17}$$

Este esquema produce un sistema linear de la forma

$$\begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix} \begin{bmatrix} a_j \\ b_j \\ c_j \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}. \tag{18}$$

En 18 $j = 2$ por lo que el elemento 1 ocurre en la fila 2, en general el elemento 1 ocurre en la fila j.

Sí se tienen un etiquetado de los nodos $E_1, \ldots, E_n$ que están en $D \cap \mathcal{S}$. Con cada nodo E_k se asocia una función ϕ_k que es lineal en cada triángulo , tiene valor 1 en E_k, y es 0 en los otros nodos(por construcción). Esta elección particular hace a ϕ_k idéntica a N_j^i en el triángulo T_i cuando el nodo E_k es el vértice denotado por $V_j^{(i)}$.

3. Formulación débil para FEM

Para empezar la discusión de la formulación débil se inicia planteando el problema original también conocido como formulación fuerte, en este caso se considera a la ecuación diferencial parcial elíptica es decir la ecuación de Poisson

$$\nabla^2 T = f(x) \tag{19}$$

Con condición de Dirichlet

$$T = u(x), \quad x \in \Gamma_D \tag{20}$$

y condición de Von Neumman

$$\nabla T \cdot \mathbf{n} = g(x), \quad x \in \Gamma_N \tag{21}$$

donde $\Omega \in \mathbb{R}$ es el dominio de la solución con contorno $d\Omega$, Γ_D es la parte del contorno donde se aplica la condición de Dirichlet y Γ_N es la parte del contorno con condiciones de Neumann, $T(x)$ es la función desconocida a encontrar o aproximar, $f(x), u(x), g(x)$ son funciones conocidas.
La forma débil de la ecuación 19 se construye al realizar la integral

$$\int_\Omega (\nabla^2 T - f) \cdot s \, d\Omega = 0 \tag{22}$$

donde s es una función de prueba, integrando por partes se obtiene

$$\begin{aligned} 0 = \int_\Omega (\nabla^2 T - f) \cdot s \, d\Omega = \int_\Omega \nabla \cdot \big(\nabla T\big) \cdot s \, d\Omega - \int_\Omega f \cdot s d\Omega = \\ = -\int_\Omega \nabla T \cdot \nabla s \, d\Omega + \int_\Omega \nabla \cdot \big(\nabla T \cdot s\big) \, d\Omega - \int_\Omega f \cdot s \, d\Omega \end{aligned} \tag{23}$$

aplicando la ley de Gauss se obtiene

$$\int_\Omega \nabla \cdot T \nabla s d\Omega = \int\limits_{\Gamma_D \cup \Gamma_N} s \cdot (\nabla \cdot \mathbf{n}) \, d\Gamma - \int_\Omega f \cdot s \, d\Omega \tag{24}$$

la integral de superficie se puede separar en una integral para la parte de la superficie donde aplica la condición de Dirichlet y otra para la parte donde aplica la condición de Neumann

$$\int_{\Omega} \nabla T \cdot \nabla s \, d\Omega = \int_{\Gamma_D} s \cdot (\nabla \cdot \mathbf{n}) \, d\Gamma + \int_{\Gamma_N} s \cdot (\nabla \cdot \mathbf{n}) \, d\Gamma - \int_{\Omega} f \cdot s \, d\Omega \tag{25}$$

La ecuación 25 es la forma débil inicial de la ecuación de Poisson, pero no se puede aplicar directamente sin considerar primero las condiciones de frontera.

3.1. Condiciones de contorno de Dirichlet

En la parte de la superficie donde se aplican las condiciones de Dirichlet se tienen dos restricciones. Una de ella es la condición de frontera $T(x) = u(x)$ y la segunda es la integral de superficie sobre Γ_D en la ecuación 25. Para evadir el término de la integral de superficie solo se usa la condición en la frontera, para lograr esto se toma a la función $T \in V(\Omega)$ y la función de prueba $s \in V_0(\Omega)$, donde

$$V(\Omega) = \{v(x) \in H^1(\Omega)\},$$
$$V_0(\Omega) = \{v(x) \in H^1(\Omega);\, v(x) = 0,\, x \in \Gamma_D\}.$$

Es decir que la función desconocida T debe ser continua conjuntamente con su gradiente en el dominio de interés. Por el otro lado la función de prueba s también debe ser continua al igual que su gradiente pero es cero en la superficie Γ_D.
Con este requerimiento el término relacionado a la integral de superficie se desvanece y la forma débil de la ecuación de Poisson para $T \in V(\Omega)$ y $s \in V_0(\Omega)$ es

$$\begin{aligned} \int_{\Omega} \nabla \cdot T \nabla s d\Omega = \int_{\Gamma_N} s \cdot (\nabla \cdot \mathbf{n}) \, d\Gamma - \int_{\Omega} f \cdot s \, d\Omega \\ T(x) = u(x), \quad x \in \Gamma_D. \end{aligned} \tag{26}$$

Por esta razón en la terminología de FEM(o en la amplia literatura sobre el método de elementos finitos) se conoce a las condiciones de Dirichlet como *Condiciones esenciales de contorno*, ya que no están incluidas naturalmente en la formulación débil y tienen que usarse que imponerse tal cual.

3.2. Condiciones de contorno de Neumann

Las condiciones de contorno de Neumann corresponden al flujo conocido $g(x) = \nabla T \cdot \mathbf{n}$. La integral sobre la superficie de Neumann en la ecuación 25 contiene exactamente el mismo flujo, por lo que en este caso se puede hacer uso directo de la función $g(x)$ en la integral

$$\int_{\Omega} \nabla T \cdot \nabla s \, d\Omega = \int_{\Gamma_N} g \cdot s \, d\Gamma - \int_{\Omega} f \cdot s \, d\Omega, \tag{27}$$

donde la función de prueba s también pertenece al espacio V_0.
Por esto en la terminología de FEM se conoce a las condiciones de Neumann como *Condiciones naturales de contorno* ya forman parte de la formulación débil.

3.3. Formulación débil para la ecuación de Poisson

Con los resultados anteriores se puede hacer la formulación débil de la ecuación de Poisson 19. Para cualquier función de prueba $s \in V_0(\Omega)$ hallar $T \in V(\Omega)$ tal que

$$\begin{aligned} \int_\Omega \nabla T \cdot \nabla s \, d\Omega = \int_{\Gamma_N} g \cdot s \, d\Gamma - \int_\Omega f \cdot s \, d\Omega, \\ T(x) = u(x), \;\; x \in \Gamma_D. \end{aligned} \tag{28}$$

4. Análisis de implementaciones

En está sección se hace un análisis de las implementaciones del método de diferencias finitas utilizando **Wolfram Mathematica 12** y **Python**, específicamente se utiliza la librería **FEniCS**(ver [3]). Para hacer las comparaciones se han elegido tres ecuaciones diferenciales parciales:

- Ecuación Elíptica
- Ecuación Hiperbólica
- Ecuación Parabólica

4.1. Ecuación Elíptica

La ecuación diferencial parcial elíptica a resolver es la siguiente

$$\begin{aligned} \nabla^2 u(x,y) &= xe^y \\ \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} &= xe^y, \;\; 0 < x < 2, \; 0 < y < 1 \end{aligned} \tag{29}$$

con condiciones de contorno

$$\begin{aligned} u(0,y) = 0, \;\; u(2,y) = 2e^y, \;\; 0 \leq y \leq 1, \\ u(x,0) = x, \;\; u(x,1) = ex, \;\; 0 \leq x \leq 2. \end{aligned} \tag{30}$$

La ecuación elíptica es la famosa ecuación de Poisson, en este caso se está considerando un problema bidimensional.

4.1.1. Implementación en Mathematica

Al resolver está ecuación 29 con el paquete `Needs["NDSolve‘FEM‘"]` de `Wolfram Mathematica 12` se obtiene la siguiente solución numérica que se puede visualizar como una superficie embedida en el espacio tridimensional.

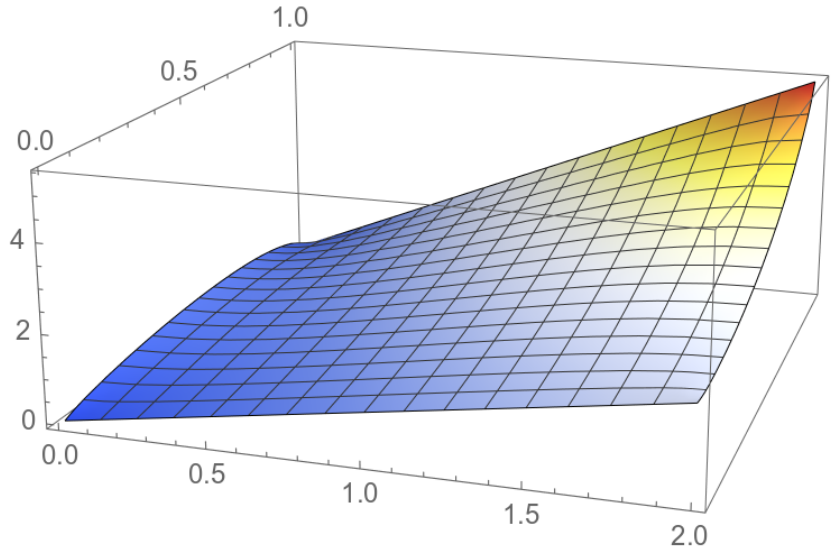

Figura 2: Visualización solución numérica de la EDP Elíptica

Para hacer uso de las funcionalidades del paquete de método de elementos finitos incluido en `Mathematica` se puede definir una región para resolver a la ecuación elíptica mucho más interesante, digamos que queremos resolver la ecuación en la región rectangular definida anteriormente pero se quieren excluir los puntos que están dentro del circulo definidos por la desigualdad

$$(x - 0{,}2)^2 + (y - 0{,}2)^2 \leq 0{,}3^2 \tag{31}$$

se obtiene la siguiente región en el plano

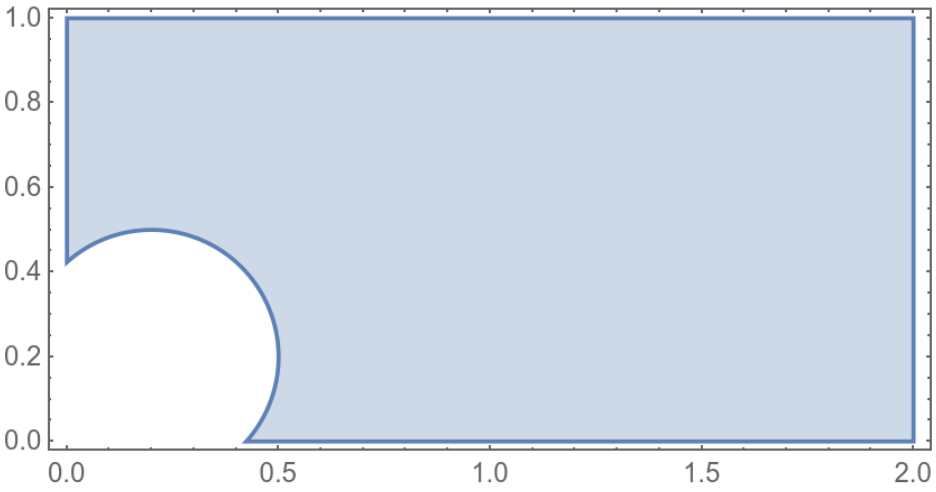

Figura 3: Región que excluye los punto en la expresión 31

Además de hacer la región más compleja se quieren imponer condiciones de Dirichlet de la forma

$$u(x,y) = \begin{cases} 0, & x = 2, \quad 0{,}8 \leq y \leq 1{,}0 \\ 10, & (x-0{,}2)^2 + (y-0{,}2)^2 \leq 0{,}3^2 \end{cases} \tag{32}$$

al resolver se obtiene las siguientes curvas de nivel

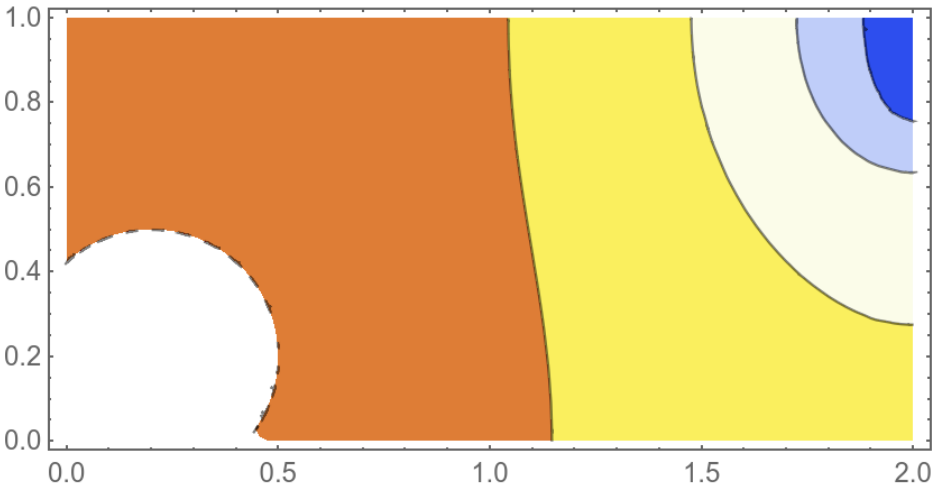

Figura 4: Curvas de nivel

y la solucion correspondiente en el espacio tridimensional se puede visualizar a continuación Se puede observar en este caso particular la teselación de la región

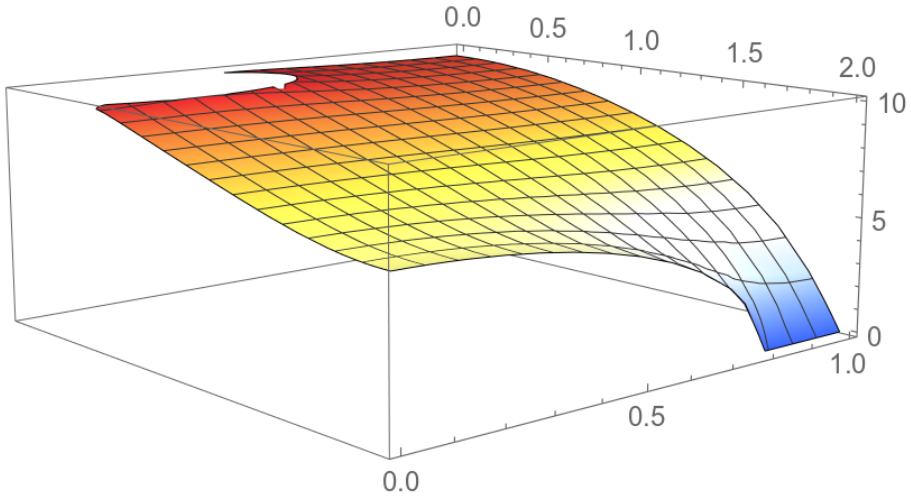

Figura 5: Solución numérica con condiciones de Dirichlet

de interés generada por el paquete anteriormente mencionado

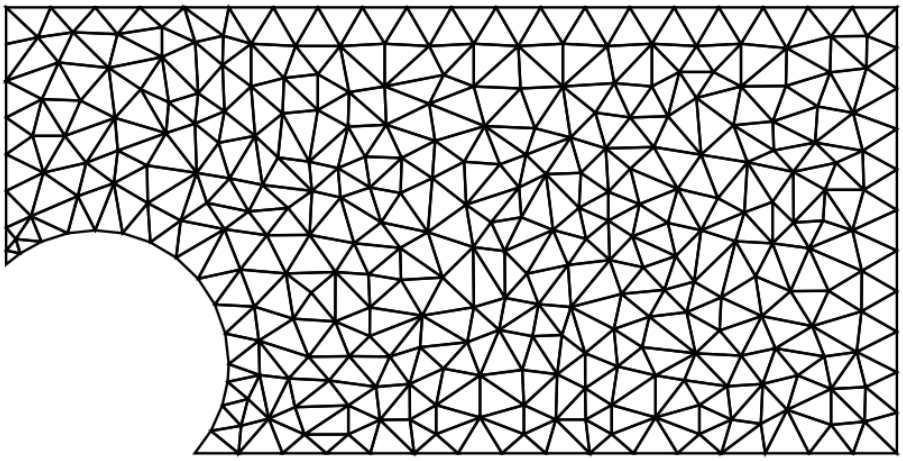

Figura 6: Teselación de la región de interés

4.1.2. Implementación en FEniCS

Para hacer uso del paquete `FEniCS` es necesario usar la formulación débil para EDP, como se expuso previamente para la ecuación elíptica o de Poisson se puede usar los términos en la integral 28 para escribir el código correspondiente. Usando el problema 29 se obtiene el siguiente resultado para la aproximación de la solución que se puede visualizar en la siguiente figura

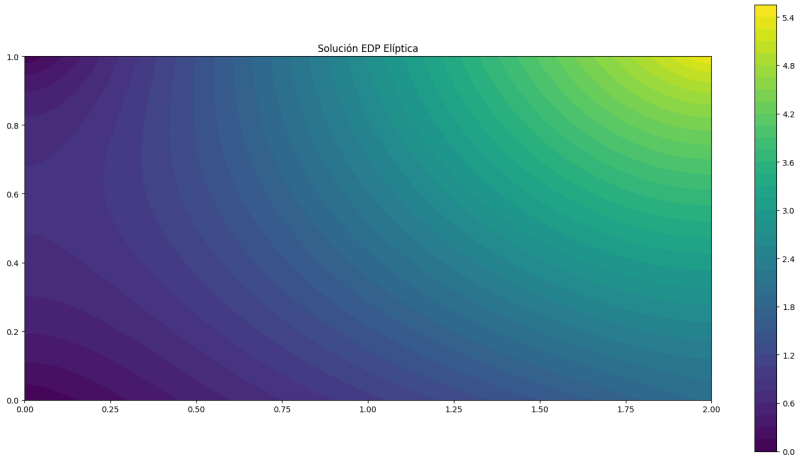

Figura 7: Solución numérica de la ecuación de Poisson

4.2. Ecuación Parabólica

Para la ecuación parabólica se tiene el siguiente problema

$$\frac{\partial u}{\partial t}(x,t) - \frac{\partial^2 u}{\partial x^2}(x,t) = 0, \quad 0 < x < 1, \quad t \geq 0 \tag{33}$$

con condiciones de frontera

$$u(0,t) = u(1,t) = 0, \quad t > 0 \tag{34}$$

y condiciones iniciales

$$u(x,0) = \sin(\pi x), \quad 0 \leq x \leq 1 \tag{35}$$

4.2.1. Implementación en Mathematica

De nuevo utilizando el paquete `Needs["NDSolve‘FEM‘"]` y la función `NDSolveValue` se puede especificar de una manera sencilla el problema con sus condiciones de frontera e iniciales, el resultado se puede visualizar de mejor manera como una gráfica de la aproximación numérica

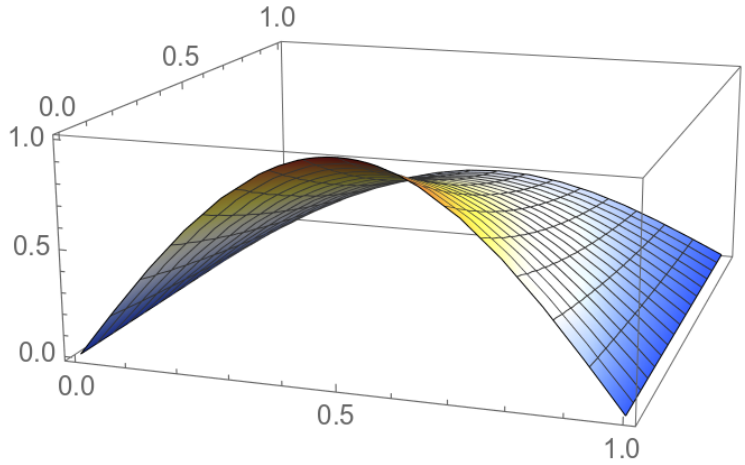

Figura 8: Aproximación numérica de la EDP parabólica

Adicionalmente es útil visualizar las curvas de nivel de la solución numérica aproximada

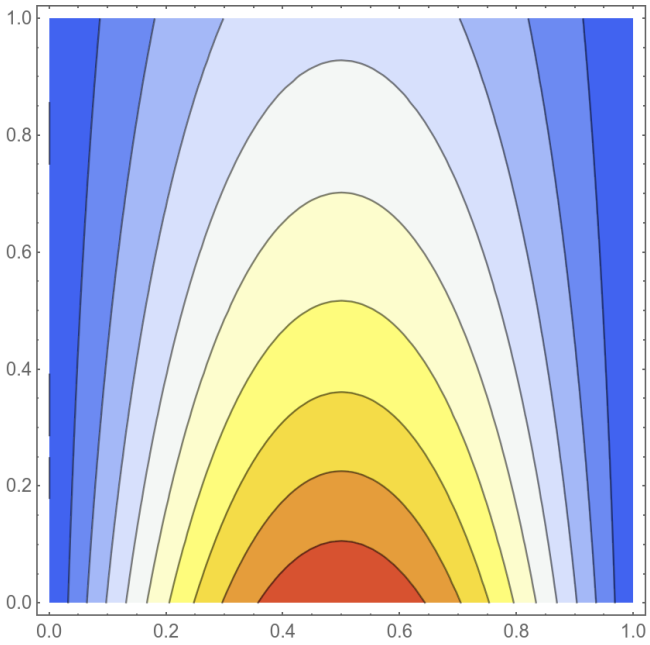

Figura 9: Curvas de nivel de la aproximación numérica de la EDP parabólica

4.2.2. Implementación en FEniCS

De nuevo para poder programar la resolución numérica de la ecuación es necesario formular la versión débil del problema 33, esto se puede revisar en el código de `Python` del repositorio de `GitHub`, ver [5].
El resultado obtenido con FEniCS se presenta en la siguiente figura

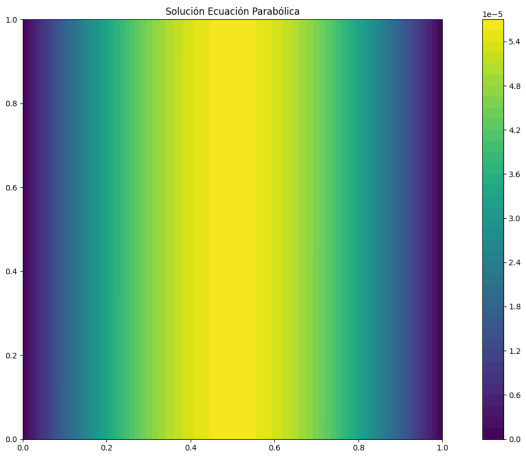

Figura 10: Aproximación para la solución de la EDP parabólica

Para poder apreciar el tipo de teselación del espacio del dominio de interés creado con `FEniCS` se usa el retículo resultante sobre la solución anterior

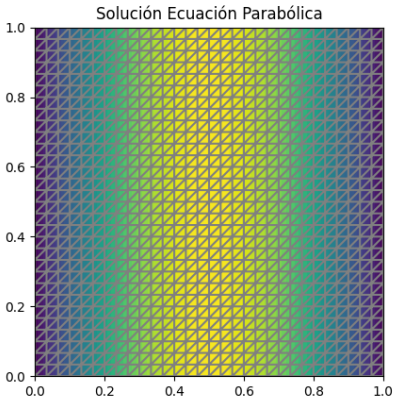

Figura 11: Teselación para la solución de la EDP parabólica

4.3. Ecuación Hiperbólica

La ecuación hiperbólica también conocida como ecuación de onda. El problema a resolver para la ecuación diferencial parcial hiperbólica es

$$\frac{\partial^2 u}{\partial t^2}(x,y) - 4\frac{\partial^2 u}{\partial x^2}(x,t) = 0, \;\; 0 < x < 1, \;\; t > 0 \tag{36}$$

con condiciones de frontera

$$u(0,t) = u(1,t) = 0, \text{ para, } t > 0 \tag{37}$$

y condiciones iniciales

$$u(x,0) = \sin(\pi x), \text{ y } \frac{\partial u}{\partial t}(x,0) = 0, \ 0 \leq x \leq 1 \tag{38}$$

4.3.1. Implementación en Mathematica

El resultado de resolver el problema 36 con las condiciones de frontera e iniciales en Mathematica se puede visualizar en la siguiente figura.

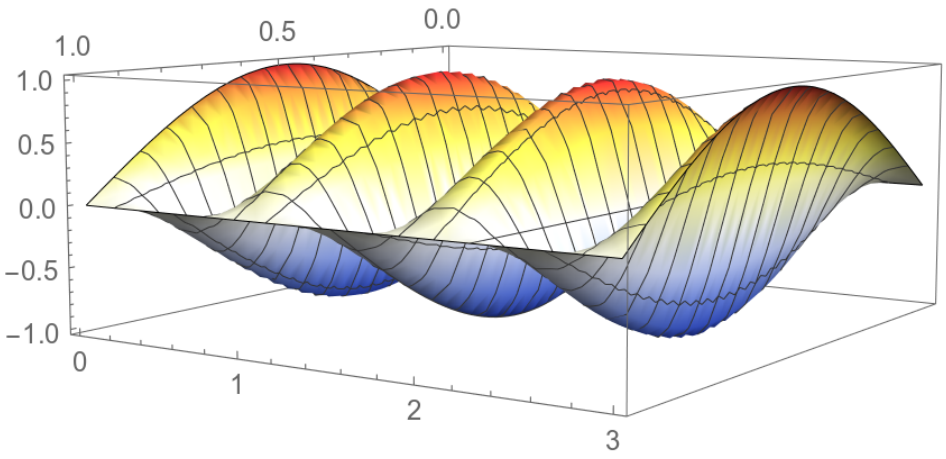

Figura 12: Solución numérica de la ecuación de onda

Las curvas de nivel correspondientes se ven en la siguiente figura

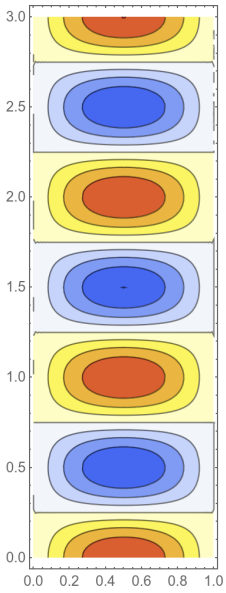

Figura 13: Curvas de nivel para la solución numérica de la ecuación de onda

4.4. Implementación en FEniCS

Utilizando la formulación débil de 36 se puede resolver el problema usando `FEniCS`, el resultado se presenta a continuación

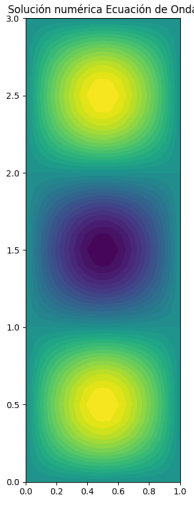

Figura 14: Curvas de nivel para la solución numérica de la ecuación de onda

5. Comparación de implementaciones

Para poder comparar las implementaciones realizadas para las soluciones de las ecuaciones diferenciales parciales cónicas presentadas anteriormente se compara la memoria RAM utilizada en promedio por cada paquete.

Ecuación Diferencial Parcial	`Wolfram Mathematica`	`Python/FEniCS`
Elíptica	188MB	266MB
Parabólica	178MB	148MB
Hiperbólica	178MB	152MB

5.1. Discretización del espacio

El paso de discretizar el dominio de interés es fundamental en el método de elementos finitos, hay muchos acercamientos teóricos de como lograr este objetivo. Como se discutió anteriormente la elección de la figura geométrica elemental para realizar la teselación tiene repercusiones directas en la construcción del método.

Para mencionar algunos de los métodos utilizado por el paquete `NeedsFEM` de `Wolfram Mathematica` se tienen los siguientes:

1. **Triangulación de Delaunay**: Este es un algoritmo común utilizado en la generación de mallas 2D. Una triangulación de un conjunto de puntos es de Delaunay si ningún punto está dentro del círculo circunscrito de cualquier triángulo. Las triangulaciones de Delaunay maximizan el ángulo mínimo de todos los ángulos de los triángulos, lo que resulta en triángulos 'bien formados'.

2. **Diagrama de Voronoi**: Este es un grafo dual a la triangulación de Delaunay. Dado un conjunto de puntos, para cada punto habrá un polígono donde todas las ubicaciones dentro del polígono están más cerca del punto dado que de cualquier otro. Los polígonos juntos forman un diagrama de Voronoi. Esto se puede usar en estrategias de refinamiento de mallas.

3. **Método Octree (Quadtree en 2D)**: Este es un método común en la generación de mallas 3D. La idea es dividir recursivamente el dominio en ocho (cuatro en 2D) partes iguales hasta que se alcance la granularidad deseada. La transición entre diferentes niveles de granularidad puede ser manejada por elementos de transición especiales.

4. **Método de Frente Avanzado**: Este método implica comenzar desde una malla de frontera y agregar gradualmente elementos al interior. Se mantiene un "frente.$^{\text{en}}$tre las áreas ya malladas y las áreas aún por mallas, y se agregan nuevos nodos de manera que este frente avance.

5. **Algoritmo de Ruppert**: Esto es un refinamiento de la triangulación de Delaunay que además tiene en cuenta un criterio sobre el ángulo mínimo, lo que lleva a triangulaciones de alta calidad.

`FEniCS` también utiliza algunas de las técnicas y marcos teóricos que utilizan en `Wolfram Mathematica` sin embargo siendo un proyecto *open-source*(código abierto) hay alguna fortalezas que se pueden mencionar de su uso:

1. **Generación de mallas**: Para el paso de generación de mallas, `FEniCS` utiliza bibliotecas como `mshr` y `CGAL` para crear y manipular mallas complejas.

2. **Montaje de Elementos Finitos**: La principal fortaleza de `FEniCS` reside en su montaje automatizado de elementos finitos. `FEniCS` utiliza el concepto de formas variacionales (la base matemática de la discretización de elementos finitos) que se puede definir en notación casi matemática directamente en un programa de Python o C++.

3. **Resolución de problemas**: `FEniCS` utiliza la biblioteca `PETSc` para la solución de sistemas de ecuaciones lineales y no lineales, una herramienta poderosa y escalable para estos cálculos.

6. Análisis estadístico en los resultados de una aproximación numérica

El análisis estadístico se puede realizar en los resultados de una aproximación numérica para comprender mejor las características y el rendimiento del método de aproximación, validar los resultados e inferir sobre el proceso que generó los datos. Algunas técnicas estadísticas utilizadas comúnmente :

1. **Estadísticas descriptivas**: Esto puede incluir la media, mediana, moda, desviación estándar, varianza, rango, mínimo, máximo, cuartiles y otras medidas que describen las características básicas de los datos. Esto puede dar una idea de la tendencia central, la dispersión y la distribución de los datos.

2. **Análisis de error**: Puede calcular los residuos (la diferencia entre los valores observados y predichos) y analizar su distribución. Podría calcular el error absoluto medio, el error cuadrado medio u otras métricas de error. Podría ser útil trazar residuos para inspeccionar visualmente los patrones.

3. **Intervalos de confianza**: Podría construir intervalos de confianza alrededor de sus parámetros estimados o predicciones para cuantificar la incertidumbre.

4. **Pruebas de hipótesis**: Si tiene una hipótesis específica sobre sus resultados, como si un método de aproximación es superior a otro, podría realizar una prueba de hipótesis. Por ejemplo, se podría utilizar una prueba t para comparar las medias de dos grupos.

5. **Análisis de correlación**: Si tiene varias variables en sus resultados, podría examinar las relaciones entre ellas utilizando coeficientes de correlación o diagramas de dispersión.

6. **Análisis de regresión**: Si sus resultados se basan en múltiples variables de entrada, puede usar el análisis de regresión para entender cómo estas variables influyen colectivamente en el resultado. Esto también puede ayudarle a predecir resultados futuros.

7. **Análisis de varianza (ANOVA)**: Si está comparando los resultados de diferentes métodos de aproximación o diferentes ajustes del mismo método, ANOVA se puede utilizar para determinar si las diferencias entre los grupos son estadísticamente significativas.

8. **Bootstrap o validación cruzada**: Estas técnicas se pueden usar para estimar la precisión de su método de aproximación al remuestrear los datos o dividirlos en subconjuntos.

9. **Análisis de Monte Carlo**: Si su aproximación numérica involucra aleatoriedad (como en la integración de Monte Carlo), entonces puede analizar la distribución de los resultados de múltiples ejecuciones para entender la media, la varianza y otras propiedades estadísticas.

7. Conclusiones

El método de elementos finitos se ha convertido en una herramienta fundamental en el modelado matemático, en las simulaciones físicas y en análisis de ingeniería. Es bien conocida la dificultad para abordar problemas de ecuaciones diferenciales parciales con condiciones en la frontera(Dirichlet o Neumann) y encontrar soluciones analíticas a estos, una ejemplo iconico en la física a nivel de postgrado se puede dar con el estudio de problemas de la teoría electromagnética encontrados en Jackson [6].

El uso de paquetes computacionales para abordar estos problemas es ubicuo en toda la ciencia, ya que la mayoría de problemas no se pueden resolver de manera analítica por lo que es obligatorio utilizar algún acercamiento numérico. Las ventajas expuestas en el manejo de las condiciones de frontera hacen que el método de elementos finitos sea una de las opciones más utilizadas en la práctica.

De los resultados obtenidos al resolver las ecuaciones cónicas se pueden establecer que `Wolfram Mathematica` y `FEniCS` tienen una implementación de FEM muy robusta y relativamente fácil de utilizar, cada uno tiene sus ventajas y desventajas. `Wolfram Mathematica` hace que la formulación del problema se pueda hacer casi de manera natural es decir que la sintaxis para definir al problema es muy similar a lo que se escribiría en una hoja de papel o en LaTeX, se escribe la EDP directamente. Para contrastar esto en `FEniCS` hay que elaborar un poco más y definir la formulación débil del problema. Uno es un software propietario y el otro de código abierto esto hace de `FEniCS` una opción más accesible y configurable a costo de un conocimiento más detallado de la programación. Las diferencias en el consumo de RAM especificamente para estos problemas no es significativa, sin embargo vale mencionar que tanto `Python` como `Mathematica` son bien conocidos por tener un consumo elevado de memoria RAM.
En cuanto a la rapidez de la soluciones el gran ganador es `Mathematica` ya resuelve los problemas casi instantáneamente mientras que en `Python` la solución puede tardar hasta un par de minutos dependiendo los parámetros utilizados. En este caso sería interesante usar `C++` para hacer una comparación de tiempos más razonable.

Referencias

[1] Richard L. Burden, J. Douglas Faires *Numerical Analysis*, (Ninth Edition). Brooks/Cole, Cengage Learning. 978-0-538-73351-9

[2] Cimrman, R., Lukeš, V., Rohan, E., 2019. *Multiscale finite element calculations in Python using SfePy.* Advances in Computational Mathematics 45, 1897-1921. https://doi.org/10.1007/s10444-019-09666-0

[3] M. S. Alnaes, J. Blechta, J. Hake, A. Johansson, B. Kehlet, A. Logg, C. Richardson, J. Ring, M. E. Rognes and G. N. Wells. *The FEniCS Project Version 1.5, Archive of Numerical Software 3 (2015).* [doi.org/10.11588/ans.2015.100.20553]

[4] A. Logg, K.-A. Mardal, G. N. Wells. *Automated Solution of Differential Equations by the Finite Element Method*, Springer(2012). [doi.org/10.1007/978-3-642-23099-8]

[5] Julio Medina. *Método de Elementos Finitos.* `https://github.com/Julio-Medina/Finite_Element_Method`

[6] John David Jackson. *Classical Electrodynamics.* Wiley, 1999.