//Segunda tarea de Métodos Numéricos

Estefanía Zuluaga Pastor,[1] Susana Toro Castaño,[1] Felipe Otálvaro Agudelo,[1] María Paula Rey[1, *]

[1]*Departamento de Ciencias Físicas, Universidad EAFIT, Medellín, Colombia.*
**Autor correspondiente: mpreyb@eafit.edu.co*

Publicado 01 Octubre 2021

La modelación matemática y computacional actualmente son indispensables para investigar el comportamiento de un sistema, para su entendimiento u optimización. Estas se utilizan en diversos campos como la industria, la ciencia y economía. Constituyen uno de los pilares fundamentales de la ciencia contemporánea. Gracias a ella, los investigadores pueden predecir satisfactoriamente el comportamiento de muchos sistemas desde el clima hasta la interacción de galaxias. En la presente entrega, se estudia la capacitancia de un sistema a partir de un potencial eléctrico, la ecuación de onda para una membrana circular, la ecuación que describe las vibraciones longitudinales y las soluciones computacionales de los ejercicios propuestos.

1. PRIMER PUNTO:

En la presente tarea se discutieron inicialmente los puntos entre los miembros del equipo. Luego, se discutió con estudiantes de la carrera de otros semestres, tales como Andrés Moreno, María José Gil y Adrián Herrera y compañeros del mismo curso como Juan Duarte y Waira Mona.

2. SEGUNDO PUNTO

Se propone un esquema de cuadratura Gaussiana para calcular la capacitancia de una malla en el dominio natural (en un dominio de cuadriláteros) utilizando la energía potencial eléctrica integral. En este método, se simplifica la integral que calcula el potencial y pasa a convertirse a una sumatoria ponderada con aproximaciones exactas, a continuación, se explica el algoritmo base del código utilizado para este problema.

El código fue desarrollado en Python. En primer lugar, se debe exportar el archivo del capacitor cilíndrico concéntrico empleando la librería *meshio*. Posteriormente se definen las variables del sistema, donde se incluye la permitividad eléctrica, la diferencia de potencial, la energía potencial, las funciones base de interpolación a un dominio canónico, los valores para s y r en el dominio natural y los diferenciales de la integral. Ahora se definen las funciones para poder calcular la cuadratura Gaussiana (los nombres de cada una corresponden a los nombres dados en el código):

A. *"mapping"*

Se construye la malla en donde se crean las conectividades entre los puntos para poder evaluar la geometría del sistema.

B. *"Jacobian"*

Se calcula el Jacobiano y su determinante para la malla y sus nodos.

C. *"potential_energy"*

Se define el potencial eléctrico con las variables físicas a calcular en cada uno de sus nodos.

D. *"gradient"*

Se calcula el gradiente de las funciones haciendo uso del Jacobiano, el potencial y los diferenciales en coordenadas naturales.

E. *"Electric_Field"*

Se procede a hacer el cálculo del campo eléctrico mediante su magnitud y la matriz de gradiente.

F. *"Gaussian_S"*

Se define la función donde se pasa la integral de energía potencial eléctrica a la forma de sumatoria para la cuadratura Gaussiana, haciendo uso de su ponderación con $n=2$ y un valor de s, $r=\pm\frac{1}{\sqrt{3}}$, con esto, se obtiene el valor del potencial eléctrico parcial (local) y se puede pasar a calcular la capacitancia.

Posteriormente se crea una función *for* donde se integran las funciones definidas anteriormente para la lista de elementos global, es decir, se calculan las funciones para todo el mallado y se obtiene el resultado de la integración de la energía potencial eléctrica. Finalmente, se utiliza la ecuación (1) para calcular la capacitancia del cilindro concéntrico, obteniendo el resultado $C=2.006657\times10^{-11}F$.

$$C=\frac{2U}{\Delta V^2} \tag{1}$$

Adicionalmente, se calcula la capacitancia teórica sin utilizar la cuadratura Gaussiana, y se compara con la obtenida en el código. La capacitancia teórica tuvo un valor de $C = 2.006518 \times 10^{-11}\ F$. El error obtenido es de $\varepsilon\% = 0{,}00691\%$

Este error es considerado despreciable, por lo cual se concluye que el método utilizado genera una aproximación aceptable.

```
La capacitancia obtenida es:  2.00665700793193e-11
La capacitancia teórica es:  2.006518396549341e-11
Porcentaje de error:  0.00690805441043307
```

Fig. 1. Resultado arrojado por el código.

3. TERCER PUNTO

Tenemos que los modos de vibración en una membrana circular están dados por la ecuación:

$$u_{mn}(\rho,\phi,t) = \cos(m\phi) J_n(\lambda_{mn}\rho) \sin(\lambda_{mn} t) \quad (2)$$

Donde $\lambda_{mn} = \alpha_{mn}/\alpha$, α_{mn} es la n-ésima raíz positiva de y α es el radio de la membrana circular. Como se trabaja con una membrana circular empotrada en su borde, como condición de frontera se debe cumplir que $u_{mn}(\alpha,\phi,t) = 0$. Obteniendo así:

$$\cos(m\phi) J_n(\lambda_{mn} a) \sin(\lambda_{mn} t) = 0 \quad (3)$$

$$J_n(\alpha_{mn}) = 0 \quad (4)$$

$$J_n(\alpha_{mn}) = 0 \quad (5)$$

Por lo que la n-ésima raíz positiva de J_m es una raíz de J_n, entonces n=m. Así, los modos de vibración para la membrana estudiada son de la forma.

$$u_k(\rho,\phi,t) = \cos(k\phi) J_k(\lambda_k \rho) \sin(\lambda_k t) \quad (6)$$

Donde $\lambda_k = \alpha_k/\alpha$, con α_k siendo la k-ésima raíz positiva de J_k y α es el radio del círculo.

Para realzar la visualización de estos modos de vibración, inicialmente se generó una geometría en Gmsh con dos círculos, uno al interior de otro, donde el objeto de estudio es el del interior que cuenta con un radio 1. Esta malla es luego leída en Python, donde se definen los 4 primeros modos de vibración de la malla para así generar archivos VTK que pueden ser leídos.

Este último paso no pudo ser posible ya que existieron problemas a la hora de agregar los paquetes de Gmish y Numpy a Python. Se intentó correr el código con diferentes interfaces y editores, además de diferentes computadores, sin embargo, en todos ocurrió el mismo problema donde no se podía agregar ninguno de los dos paquetes mencionados anteriormente. En la entrega se adjuntan los archivos de geometría y malla de la membrana y el código en Python para generar los archivos VTK.

La solución de la ecuación de onda para una membrana circular empotrada en su borde puede expresarse como una combinación lineal de los modos de vibración de la misma. En esta sección, se realizó una visualización en ParaView para estos modos. Tomando los modos de vibración como:

$$u_{mn}(\rho,\phi,t) = \cos(m\phi) J_n(\lambda_{mn}\rho) \sin(\lambda_{mn} t) \quad (6)$$

Donde $\lambda_{mn} = \frac{\alpha_{mn}}{a}$, α_{mn} es la n-ésima raíz positiva de J_m, y a es el radio del círculo.

Para realizar esta visualización, se generó una malla en Gmsh formada por 1000 triángulos. Luego, se leyó esta malla en Python y se exportaron los archivos VTK para 4 modos de vibración. Finalmente, se visualizaron en ParaView, como se muestra a continuación:

4. CUARTO PUNTO

Se tiene la ecuación que describe las vibraciones longitudinales de una barra de longitud L y sección variable, la cual está dada por:

$$\frac{\rho A(x)}{E} \frac{\partial^2 u}{\partial t^2} = \frac{\partial}{\partial x}\left[A(x) \frac{\partial u}{\partial x} \right]$$

Con las condiciones de frontera:

$$u(x=0) = 0$$

Y

$$\frac{\partial u}{\partial x} | x = L = 0$$

Para resolver la ecuación planteada se procede a simplificarla, reescribiéndola primero como:

$$\frac{\rho A(x)}{E} \frac{\partial^2 u}{\partial t^2} = A(x) \frac{\partial^2 u}{\partial x^2}$$

Luego, cancelando el término A(x) y pasando los términos ρ y E al otro lado de la ecuación:

$$\frac{\partial^2 u}{\partial t^2} = \frac{E}{\rho} \frac{\partial^2 u}{\partial x^2}$$

Por otro lado se sabe que la velocidad de propagación de las deformaciones longitudinales está dada por

$$\sqrt{\frac{E}{\rho}}=v$$

Por lo que la ecuación se puede escribir como

$$\frac{\partial^2 u}{\partial t^2}=v^2\frac{\partial^2 u}{\partial x^2}$$

a) A continuación se propone un esquema de diferencias finitas para resolver el problema. Se utilizan las siguientes ecuaciones de diferencias finitas de segunda derivada para aproximar la solución:

$$\frac{\partial^2}{\partial t^2}u\left(x_i,t_n\right)=\frac{u_i^{n+1}-2u_i^n+u_i^{n-1}}{\Delta t^2}$$

$$\frac{\partial^2}{\partial x^2}u\left(x_i,t_n\right)=\frac{u_{i+1}^n-2u_i^n+u_{i-1}^n}{\Delta x^2}$$

Reemplazando en nuestra ecuación se obtiene

$$u_i^{n+1}=C^2\left[u_{i+1}^n+2u_i^n+u_{i-1}^n\right]+2u_i^n-u_i^{n-1}$$

Así, es posible solucionar para obtener la deformación de un tiempo n. Sin embargo, no se tiene un valor para el tiempo "u_i^{n-1}" pues corresponde a un tiempo anterior al inicio del problema, por lo que se debe encontrar una expresión para poder reemplazar ese término. Utilizando la condición inicial u(x=0)=0 se puede llegar a la expresión:

$$u_i^{n+1}=u_i^{n-1}$$

Obtenemos entonces

$$u_i^{n+1}=C^2\left[u_{i+1}^n+2u_i^n+u_{i-1}^n\right]+2u_i^n-u_i^{n+1}$$

Ahora sin estar en la condición inicial u(x=0)=0, donde no conocemos u_i^{n-1}, podemos calcular su valor porque ha pasado cierto tiempo como nos indica la condición que aparece en la ecuación. Por lo que retomando

$$u_i^{n+1}=C^2\left[u_{i+1}^n-2u_i^n+u_{i-1}^n\right]+2u_i^n-u_i^{n-1}$$

Finalmente se puede resolver implementando un código de diferencias finitas para obtener las deformaciones en cada instante de tiempo.

b) Se adjunta código de la implementación en Python. Este código también puede ser visualizado en: https://github.com/estefaniazuluaga/Vibracion_longitudinal

Referencias

[1] Langtangen, H. P., & Linge, S. (2017). Finite difference computing with PDEs: a modern software approach. Springer Nature.

[2] Cursos Abiertos Universidad. Ecuaciones diferenciales parciales, mecanica, ecuacion de onda diferencias finitas python. 2019. URL: https://www.youtube.com/watch?v=1oKhj1eWq4t = 1937s

[3] Solving one-dimensional wave equation by finite difference method (python). URL: https://linuxtut.com/en/b109f960b0a109dc358e/