

DIFERENCIAS FINITAS

PROBLEMAS DE CALIBRACIÓN: BC NEUMMAN Y κ VARIABLE

Modelación computacional en las ciencias y las ingenierías como
apoyo en el proceso enseñanza-aprendizaje
(PAPIME-PE101019)

Instituto de Geofísica
Universidad Nacional Autónoma de México



Esta obra está bajo una [Licencia Creative Commons Atribución-NoComercial-CompartirIgual 4.0 Internacional](#).



- ① CALIBRACIÓN 1.
Ejercicio 3.
- ② CALIBRACIÓN 2: CONDICIONES TIPO NEUMMAN
Aproximación I.
Aproximación II.
Aproximación III.
Medición el error
Ejercicio 4.
- ③ CALIBRACIÓN 3: CONDUCTIVIDAD VARIABLE
Ejercicio 5.
- ④ REFERENCIAS
- ⑤ CRÉDITOS

CONTENIDO

- ① CALIBRACIÓN 1.
Ejercicio 3.
- ② CALIBRACIÓN 2: CONDICIONES TIPO NEUMMAN
Aproximación I.
Aproximación II.
Aproximación III.
Medición el error
Ejercicio 4.
- ③ CALIBRACIÓN 3: CONDUCTIVIDAD VARIABLE
Ejercicio 5.
- ④ REFERENCIAS
- ⑤ CRÉDITOS

MODELO MATEMÁTICO

Considere el siguiente problema:

$$\begin{aligned}
 -\left(\frac{d^2T(x)}{dx^2} + \omega^2T(x)\right) &= 0 \quad x \in [0, 1] \\
 T(0) &= 1 \\
 T(1) &= 1
 \end{aligned} \tag{1}$$

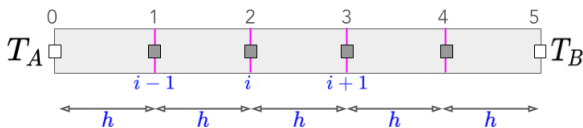
cuya solución analítica es: $T(x) = \frac{1 - \cos(\omega)}{\sin(\omega)} \sin(\omega x) + \cos(\omega x)$

donde $\omega = \text{constante}$.

MODELO NUMÉRICO

Recordemos que la segunda derivada se puede aproximar como sigue (véase la figura como referencia):

$$\left. \frac{d^2 T}{dx^2} \right|_i = \frac{T_{i+1} - 2T_i + T_{i-1}}{h^2} + \mathcal{O}(h^2) \quad (2)$$



Usando (2) en (1) obtenemos

$$-\left(\frac{T_{i+1} - 2T_i + T_{i-1}}{h^2} + \omega^2 T_i \right) = 0$$

que podemos escribir como:

$$-T_{i+1} + (2 - \omega^2/r_i)T_i - T_{i-1} = 0$$

donde $r_i = \frac{\kappa_i}{h^2}$ y en este caso $\kappa_i = 1, \forall i$.

MODELO NUMÉRICO

El sistema lineal para este caso, para cualquier N , es:

$$\underbrace{\begin{bmatrix} 2 - \omega^2/r & -1 & 0 & 0 & \dots & 0 \\ -1 & 2 - \omega^2/r & -1 & 0 & \dots & 0 \\ 0 & -1 & 2 - \omega^2/r & -1 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & -1 & 2 - \omega^2/r & -1 \\ 0 & \dots & 0 & 0 & -1 & 2 - \omega^2/r \end{bmatrix}}_{\underline{\underline{\mathbf{A}}}_{N \times N}} \underbrace{\begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ \vdots \\ T_{N-1} \\ T_N \end{bmatrix}}_{\mathbf{T}_N} = \frac{1}{r} \underbrace{\begin{bmatrix} S_1 \\ S_2 \\ S_3 \\ \vdots \\ S_{N-1} \\ S_N \end{bmatrix}}_{\mathbf{b}_N} + \underbrace{\begin{bmatrix} T_A \\ 0 \\ 0 \\ \vdots \\ 0 \\ T_B \end{bmatrix}}$$

donde $r = \frac{\kappa}{h^2}$ y $S_i = 0, \forall i$.

EJERCICIO 3.

Usando como base el código del Ejercicio 2, resuelva el sistema lineal de este problema de calibración. Para ello utilice el notebook `E03.Calibracion1.ipynb`, del repositorio [Mixbaal](#). Note que ya existe la siguiente función:

```
def buildMatrix(N, d):
```

Esta función construye la matriz del sistema y recibe como parámetros el tamaño del sistema N y el contenido de la diagonal principal, d , de la matriz, que en este caso debe ser: $2 - \omega^2/r$. Realice lo siguiente:

- 1 Agregue una celda para definir los parámetros físicos y numéricos del problema (de manera similar al ejercicio 2).
- 2 Agregue una celda para resolver el sistema lineal. Observe que en este caso la construcción de la matriz deberá realizarse como sigue:

```
A = buildMatrix(N, 2 - w**2/r)
```

- 3 Reproduzca la gráfica de la siguiente página en donde se usó $\omega = 2.5\pi$.

```
w = 2.5 * np.pi
```

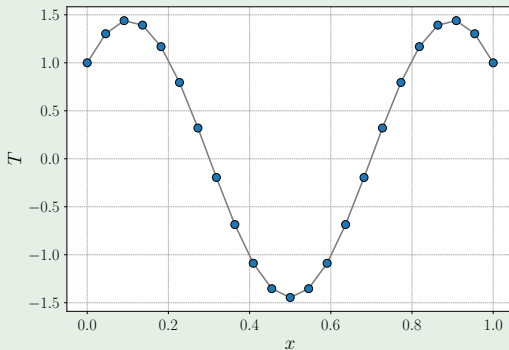
EJERCICIO 3.

Solución:

```

T = [ 1.          1.30272968  1.43942855  1.39267454  1.16842639  0.79526416
      0.32074678 -0.19464927 -0.68523757 -1.08849344 -1.35302256 -1.44511112
      -1.35302256 -1.08849344 -0.68523757 -0.19464927  0.32074678  0.79526416
      1.16842639  1.39267454  1.43942855  1.30272968  1.          ]

```



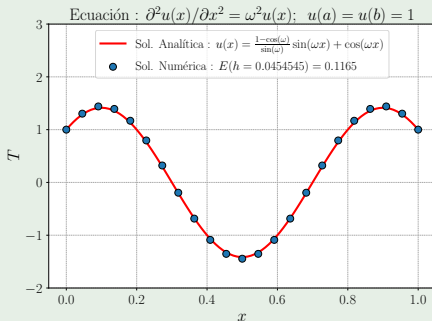
EJERCICIO 3.

- 4 Agregue una celda con la implementación de la solución exacta:

```
def solExact(x, w):
    return ((1.0 - np.cos(w))/np.sin(w)) * np.sin(w * x) + np.cos(w * x)
```

- 5 Calcule la solución exacta, compare con la solución numérica calculando la norma L-2 del error absoluto y reproduzca la gráfica de abajo:

```
Error = np.linalg.norm(solExact(x,w) - T, 2) # Norma L-2 del error absoluto
```



CONTENIDO

- ① CALIBRACIÓN 1.
Ejercicio 3.
- ② CALIBRACIÓN 2: CONDICIONES TIPO NEUMMAN
Aproximación I.
Aproximación II.
Aproximación III.
Medición el error
Ejercicio 4.
- ③ CALIBRACIÓN 3: CONDUCTIVIDAD VARIABLE
Ejercicio 5.
- ④ REFERENCIAS
- ⑤ CRÉDITOS

MODELO MATEMÁTICO

Considere el siguiente problema:

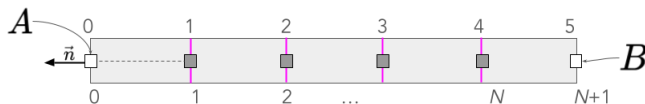
$$\begin{aligned} -\frac{d^2 u(x)}{dx^2} &= -e^x \quad x \in [0, 1] \\ \frac{du}{dn}(0) &= 0 \\ u(1) &= 3 \end{aligned} \tag{3}$$

cuya solución analítica es: $u(x) = e^x - x - e + 4$

Obsérvese que en este caso se proporciona una condición de tipo **Neumman** en la frontera izquierda del dominio ($x = 0$).

A continuación mostramos tres aproximaciones que se pueden usar para incorporar las condiciones de tipo Neumman en el sistema lineal.

MODELO NUMÉRICO: APROXIMACIÓN DE PRIMER ORDEN



- La condición de frontera en $x = 0$ indica que no se conoce la u , pero se da la condición: $\left. \frac{du}{dn} \right|_0 = A$, con $A = 0$ (condición tipo **Neumman**).
- Se observa en la figura que se considera la normal a la frontera \vec{n} hacia afuera del dominio, por lo tanto: $\left. \frac{du}{dn} \right|_0 = -\left. \frac{du}{dx} \right|_0$.
- Se puede entonces aproximar la derivada con DF hacia adelante:

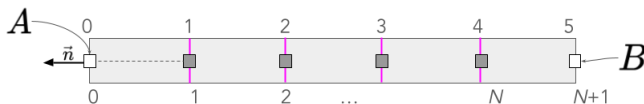
$$-\left. \frac{du}{dx} \right|_0 \approx -\frac{u_1 - u_0}{h} = A \Rightarrow \boxed{u_0 - u_1 = hA}$$

esta ecuación se debe incorporar al sistema lineal.

- La condición de frontera en $x = 1$ es $u_{N+1} = B$, con $B = 3$ (condición tipo **Dirichlet**) por lo tanto, para el nodo N se tiene la siguiente ecuación:

$$-u_{N-1} + 2u_N - u_{N+1} = r^{-1}f_N \Rightarrow \boxed{-u_{N-1} + 2u_N = r^{-1}f_N + B}$$

MODELO NUMÉRICO: APROXIMACIÓN DE PRIMER ORDEN



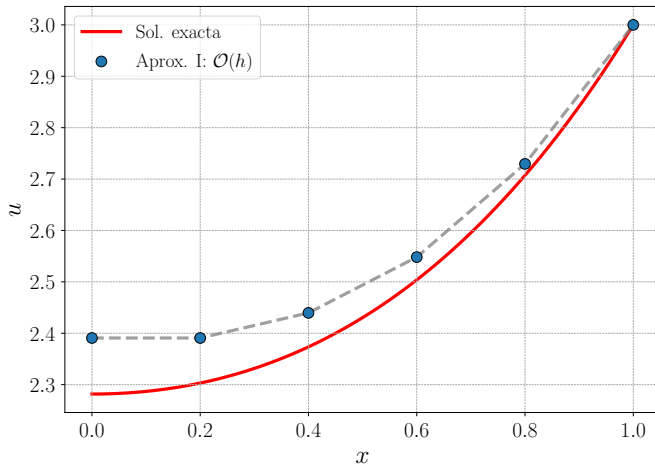
- Para los puntos 1 a N , la forma discreta de la ecuación (3) usando DF centradas es $-u_{i-1} + 2u_i - u_{i+1} = r^{-1}f_i$, donde $f(x) = -e^x$, es decir $f(x_i) = f_i = -e^{x_i}$.
- El **sistema lineal** con las condiciones de frontera incorporadas para esta aproximación es el siguiente:

$$\Rightarrow \underbrace{\begin{bmatrix} 1 & -1 & 0 & \dots & 0 & 0 \\ -1 & 2 & -1 & \dots & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\ 0 & \dots & -1 & 2 & -1 & 0 \\ 0 & \dots & 0 & -1 & 2 & -1 \\ 0 & \dots & 0 & 0 & -1 & 2 \end{bmatrix}}_{(N+1) \times (N+1)} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-2} \\ u_{N-1} \\ u_N \end{bmatrix} = \frac{1}{r} \begin{bmatrix} 0 \\ f_1 \\ \vdots \\ f_{N-2} \\ f_{N-1} \\ f_N \end{bmatrix} + \begin{bmatrix} hA \\ 0 \\ \vdots \\ 0 \\ 0 \\ B \end{bmatrix}$$

- El **orden** de la aproximación es lineal $\mathcal{O}(h)$.

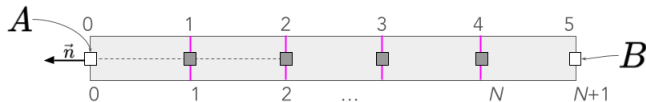
MODELO NUMÉRICO: APROXIMACIÓN DE PRIMER ORDEN

Esta aproximación genera el siguiente resultado:



MODELO NUMÉRICO: APROX. DE SEGUNDO ORDEN (TRES PUNTOS)

- Es posible aproximar la derivada normal usando tres puntos hacia adelante (véase el ejemplo 2 de la presentación *Diferencias Finitas: Introducción*):

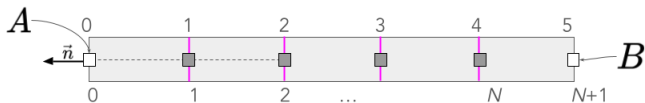


- Entonces, la fórmula de la aproximación queda como sigue:

$$\left. \frac{\partial u}{\partial n} \right|_0 = - \left. \frac{\partial u}{\partial x} \right|_0 \approx - \frac{-3u_0 + 4u_1 - u_2}{2h} = A \Rightarrow \boxed{3u_0 - 4u_1 + u_2 = 2hA}$$

esta ecuación se debe incorporar al sistema lineal.

MODELO NUMÉRICO: APROX. DE SEGUNDO ORDEN (TRES PUNTOS)



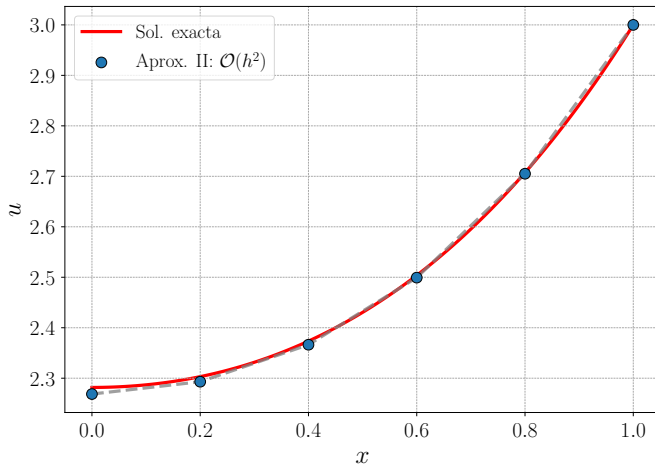
El **sistema lineal** para esta aproximación es el siguiente:

$$\Rightarrow \underbrace{\begin{bmatrix} 3 & -4 & 1 & \dots & 0 & 0 \\ -1 & 2 & -1 & \dots & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\ 0 & \dots & -1 & 2 & -1 & 0 \\ 0 & \dots & 0 & -1 & 2 & -1 \\ 0 & \dots & 0 & 0 & -1 & 2 \end{bmatrix}}_{(N+1) \times (N+1)} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-2} \\ u_{N-1} \\ u_N \end{bmatrix} = \frac{1}{r} \begin{bmatrix} 0 \\ f_1 \\ \vdots \\ f_{N-2} \\ f_{N-1} \\ f_N \end{bmatrix} + \begin{bmatrix} 2hA \\ 0 \\ \vdots \\ 0 \\ 0 \\ B \end{bmatrix}$$

El **orden** de la aproximación es cuadrático $\mathcal{O}(h^2)$.

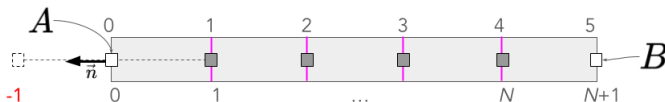
MODELO NUMÉRICO: APROX. DE SEGUNDO ORDEN (TRES PUNTOS)

Esta aproximación genera el siguiente resultado:



MODELO NUMÉRICO: APROX. DE SEGUNDO ORDEN (CENTRALES)

- Otra manera de obtener un orden cuadrático de aproximación en la derivada normal es mediante diferencias centrales. Para ello supondremos que existe un nodo a la izquierda del nodo 0, al cual generalmente se le conoce como nodo “fantasma”, y lo etiquetamos con el subíndice -1 :



- En este caso la fórmula de la aproximación queda como sigue:

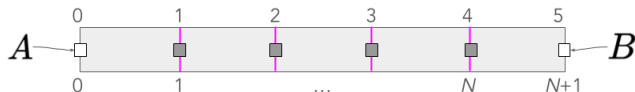
$$\left. \frac{\partial u}{\partial n} \right|_0 = - \left. \frac{\partial u}{\partial x} \right|_0 \approx - \frac{(u_1 - u_{-1})}{2h} = A \Rightarrow u_{-1} = 2hA + u_1$$

- Como no existe el nodo -1 , escribimos la ecuación discreta para el nodo 0: $-u_{-1} + 2u_0 - u_1 = r^{-1}f_0$ y sustituimos en ella la fórmula anterior para obtener:

$$\Rightarrow u_0 - u_1 = \frac{1}{2r}f_0 + hA$$

esta ecuación se debe incorporar al sistema lineal.

MODELO NUMÉRICO: APROX. DE SEGUNDO ORDEN (CENTRALES)



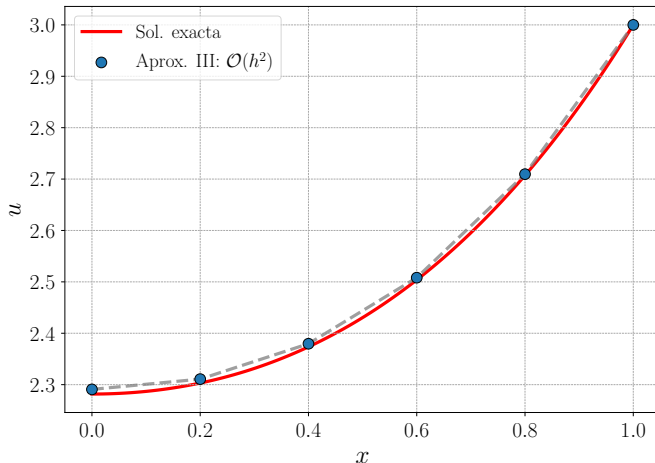
El **sistema lineal** para esta aproximación es el siguiente:

$$\Rightarrow \underbrace{\begin{bmatrix} 1 & -1 & 0 & \dots & 0 & 0 \\ -1 & 2 & -1 & \dots & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\ 0 & \dots & -1 & 2 & -1 & 0 \\ 0 & \dots & 0 & -1 & 2 & -1 \\ 0 & \dots & 0 & 0 & -1 & 2 \end{bmatrix}}_{(N+1) \times (N+1)} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-2} \\ u_{N-1} \\ u_N \end{bmatrix} = \frac{1}{r} \begin{bmatrix} f_0/2 \\ f_1 \\ \vdots \\ f_{N-2} \\ f_{N-1} \\ f_N \end{bmatrix} + \begin{bmatrix} hA \\ 0 \\ \vdots \\ 0 \\ 0 \\ B \end{bmatrix}$$

El **orden** de la aproximación es cuadrático $\mathcal{O}(h^2)$.

MODELO NUMÉRICO: APROX. DE SEGUNDO ORDEN (CENTRALES)

Esta aproximación genera el siguiente resultado:

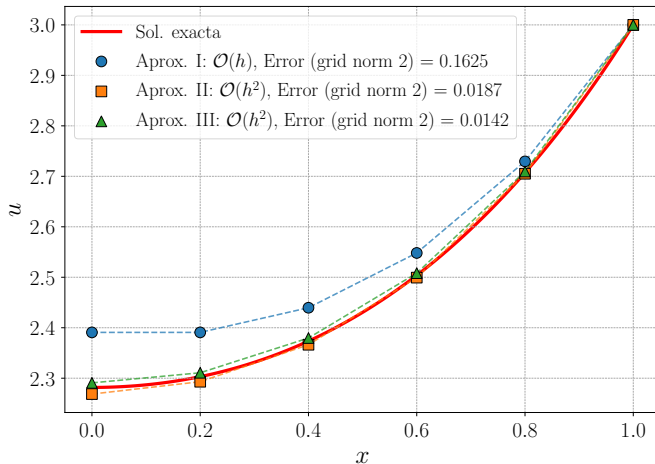


MEDICIÓN DEL ERROR

- Si $\hat{u}(x)$ representa la solución exacta del problema, definimos el error absoluto en el punto x_i de la aproximación como $e_i = u(x_i) - \hat{u}(x_i)$.
- El vector $\mathbf{E} = (e_0, e_1, \dots, e_{N+1})$ representa el error en todos los puntos de la malla.
- La magnitud (norma) de este vector nos proporciona el error global de la aproximación. Se puede usar cualquier norma por el teorema de equivalencia y para tener un significado más descriptivo del error usaremos las llamadas normas de malla (*grid norms*), véase [2].
- Grid norm ∞ : $\|E\|_\infty = \max_{1 \leq i \leq N} |e_i|$
- Grid norm 1: $\|E\|_1 = h \sum_{i=1}^N |e_i|$ donde $h \approx \frac{(b-a)}{N}$ representa el tamaño de la malla.
- Grid norm 2: $\|E\|_2 = \left(h \sum_{i=1}^N |e_i|^2 \right)^{1/2}$

MEDICIÓN DEL ERROR

Comparación de las tres aproximaciones.



EJERCICIO 4.

Abra el notebook `E04_Calibracion2.ipynb` del repositorio [Mixbaal](#). Analice y entienda cada una de las 7 celdas de código que están implementadas. Observe que en la celda 6 se resuelve el problema con las tres aproximaciones para la condición de Neumman y se genera la gráfica presentada en la página anterior (22). La celda 7 contiene la siguiente función:

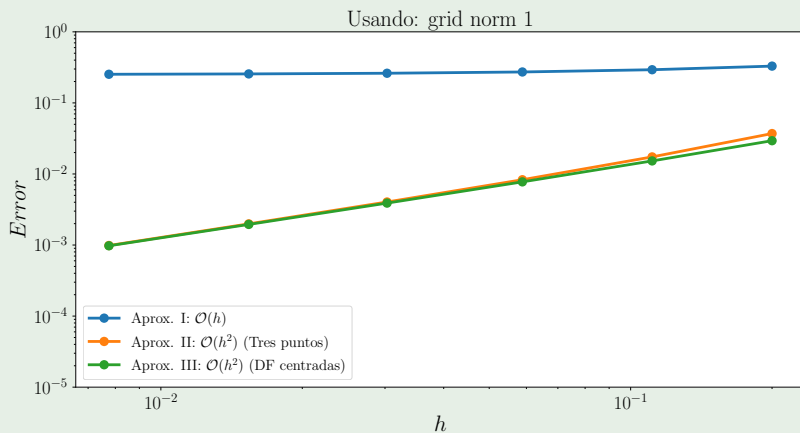
```
def meshRefining(fcondNeumman, nodes, norma):
```

Esta función permite realizar un estudio de refinamiento de malla. En este tipo de estudio se resuelve el problema para diferente número de nodos y se calcula el error con respecto a la solución exacta.

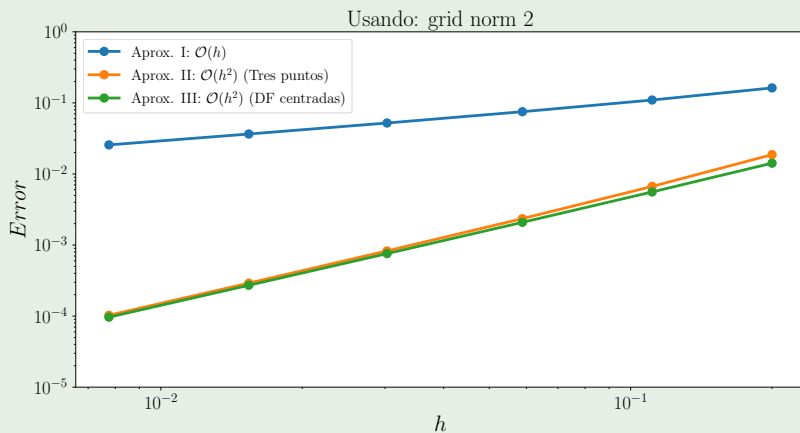
- 1 Realice un estudio de refinamiento de malla para el siguiente número de nodos (internos) **4, 8, 16, 32, 64, 128** y reproduzca las tres gráficas de las siguientes páginas.

Hint: Use una lista para los nodos: `nodes = [4, 8, ...]`; enseguida calcule la h para cada uno de estos nodos usando una *list comprehension*; posteriormente ejecute la función `meshRefining()` pasando como parámetros las funciones que calculan las aproximaciones de la condición Neumman, la lista de nodos y la norma correspondiente para el cálculo del error. Note que la función `meshRefining()` regresa la lista de errores con la cual puede generar las gráficas requeridas.

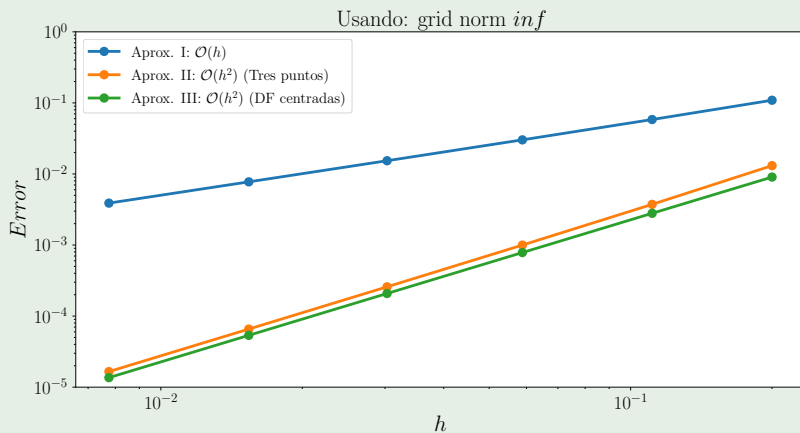
EJERCICIO 4.



EJERCICIO 4.



EJERCICIO 4.



CONTENIDO

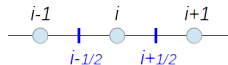
- ① CALIBRACIÓN 1.
Ejercicio 3.
- ② CALIBRACIÓN 2: CONDICIONES TIPO NEUMMAN
Aproximación I.
Aproximación II.
Aproximación III.
Medición el error
Ejercicio 4.
- ③ CALIBRACIÓN 3: CONDUCTIVIDAD VARIABLE
Ejercicio 5.
- ④ REFERENCIAS
- ⑤ CRÉDITOS

MODELO MATEMÁTICO Y NUMÉRICO

Considere la ecuación de Poisson con κ variable y condiciones de frontera de tipo Dirichlet:

$$-\frac{d}{dx} \left(\kappa \frac{du}{dx} \right) = f \quad \text{con} \quad \kappa = \kappa(x) \quad (4)$$

Definimos $g = \kappa \frac{du}{dx}$ por lo tanto $\frac{d}{dx} \left(\kappa \frac{du}{dx} \right) = \frac{dg}{dx}$. Esta derivada se puede aproximar como sigue (véase la figura):

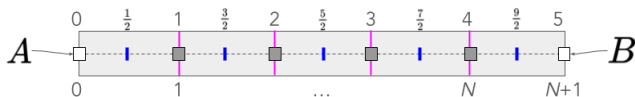


$$\left. \frac{dg}{dx} \right|_i = \frac{g_{i+\frac{1}{2}} - g_{i-\frac{1}{2}}}{h} = \frac{\left[\kappa \frac{du}{dx} \right]_{i+\frac{1}{2}} - \left[\kappa \frac{du}{dx} \right]_{i-\frac{1}{2}}}{h} \quad (5)$$

$$\left[\kappa \frac{du}{dx} \right]_{i+\frac{1}{2}} = \kappa_{i+\frac{1}{2}} \left[\frac{u_{i+1} - u_i}{h} \right] = \frac{1}{h} \left[\kappa_{i+\frac{1}{2}} u_{i+1} - \kappa_{i+\frac{1}{2}} u_i \right] \quad (6)$$

$$\left[\kappa \frac{du}{dx} \right]_{i-\frac{1}{2}} = \kappa_{i-\frac{1}{2}} \left[\frac{u_i - u_{i-1}}{h} \right] = \frac{1}{h} \left[\kappa_{i-\frac{1}{2}} u_i - \kappa_{i-\frac{1}{2}} u_{i-1} \right] \quad (7)$$

MODELO NUMÉRICO



Usando (5), (6) y (7) en (4) obtenemos:

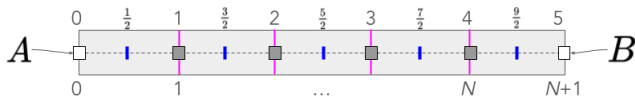
$$-\frac{1}{h^2} \left[\kappa_{i+\frac{1}{2}} u_{i+1} - (\kappa_{i+\frac{1}{2}} + \kappa_{i-\frac{1}{2}}) u_i + \kappa_{i-\frac{1}{2}} u_{i-1} \right] = f_i \quad (8)$$

Condiciones de frontera tipo Dirichlet: $u_0 = A$ y $u_{N+1} = B$

$$i = 1 \implies -\kappa_{1+\frac{1}{2}} u_2 + (\kappa_{1+\frac{1}{2}} + \kappa_{1-\frac{1}{2}}) u_1 = h^2 f_1 + \kappa_{1-\frac{1}{2}} A$$

$$i = N \implies (\kappa_{N+\frac{1}{2}} + \kappa_{N-\frac{1}{2}}) u_N - \kappa_{N-\frac{1}{2}} u_{N-1} = h^2 f_N + \kappa_{N+\frac{1}{2}} B$$

MODELO NUMÉRICO

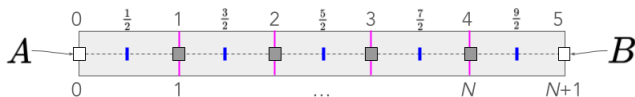


Las conductividades $\kappa_{i+\frac{1}{2}}$ y $\kappa_{i-\frac{1}{2}}$ se pueden aproximar de varias maneras, por ejemplo:

Promedio aritmético: $\kappa_{i+\frac{1}{2}} = \frac{\kappa_{i+1} + \kappa_i}{2}$ y $\kappa_{i-\frac{1}{2}} = \frac{\kappa_{i-1} + \kappa_i}{2}$

Media armónica: $\kappa_{i+\frac{1}{2}} = \frac{2\kappa_{i+1}\kappa_i}{\kappa_{i+1} + \kappa_i}$ y $\kappa_{i-\frac{1}{2}} = \frac{2\kappa_{i-1}\kappa_i}{\kappa_{i-1} + \kappa_i}$

MODELO NUMÉRICO)



La ecuación (8) la podemos escribir como

$$-b_i u_{i+1} + a_i u_i - c_i u_{i-1} = h^2 f_i \quad (9)$$

donde $b_i = \kappa_{i+\frac{1}{2}}$, $c_i = \kappa_{i-\frac{1}{2}}$ y $a_i = b_i + c_i$. Con esta notación, el **sistema lineal** es el siguiente:

$$\Rightarrow \underbrace{\begin{bmatrix} a_1 & -b_1 & 0 & \dots & 0 & 0 \\ -c_2 & a_2 & -b_2 & \dots & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\ 0 & \dots & 0 & -c_{N-1} & a_{N-1} & -b_{N-1} \\ 0 & \dots & 0 & 0 & a_N & -b_N \end{bmatrix}}_{N \times N} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{N-1} \\ u_N \end{bmatrix} = h^2 \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{N-1} \\ f_N \end{bmatrix} + \begin{bmatrix} \kappa_{1-\frac{1}{2}} A \\ 0 \\ \vdots \\ 0 \\ \kappa_{N+\frac{1}{2}} B \end{bmatrix}$$

EJERCICIO 5.

Abra el notebook E05_Calibracion3.ipynb, del repositorio [Mixbaal](#). Implemente lo siguiente:

- 1 En la celda 2: las funciones para calcular las conductividades en los puntos medios, es decir el promedio aritmético y la media armónica:

```
def pAritmetico(a, b):  
    return 0.5 * (a + b)  
  
def mArmonica(a, b):  
    return 2 * a * b / (a + b)
```

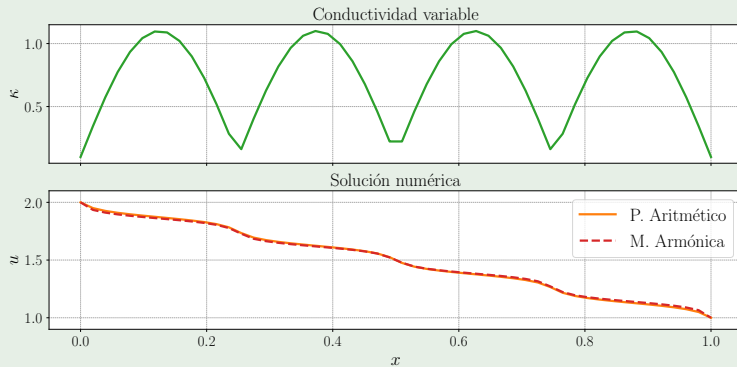
- 2 En la celda 3: una función para construir la matriz, que reciba como parámetros el tamaño del sistema N , la conductividad κ y la función para calcular la conductividad en los puntos medios f :

```
def buildMatrix(N, k, f):
```

- 3 Las celdas 4, 5 y 6 contienen código que hace uso de las funciones anteriores para calcular la solución. Reproduzca los casos que se describen en las siguientes dos páginas:

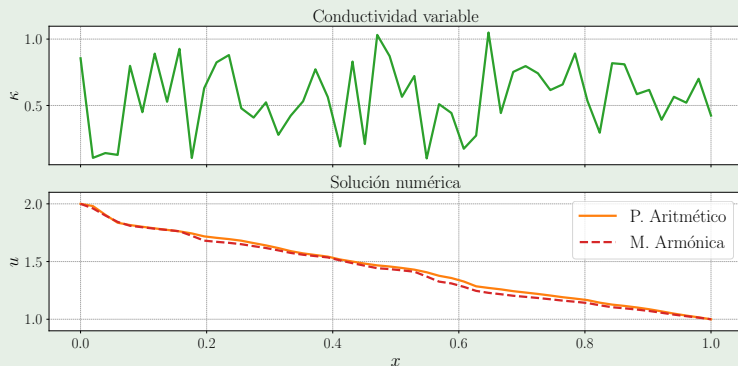
EJERCICIO 5.

③ Caso 1: $L = 1$, $N = 50$, $A = 2.0$, $B = 1.0$, $\kappa = |\sin(4\pi x)| + \delta\kappa$ con $\delta\kappa = 0.1$.



EJERCICIO 5.






③ Caso 2: $L = 1$, $N = 50$, $A = 2.0$, $B = 1.0$, $\kappa = \text{random}(x) + \delta\kappa$ con $\delta\kappa = 0.1$.



Note que en este caso la solución no se reproducirá exactamente igual debido al uso de valores pseudo-aleatorios para la conductividad.

CONTENIDO

- ① CALIBRACIÓN 1.
Ejercicio 3.
- ② CALIBRACIÓN 2: CONDICIONES TIPO NEUMMAN
Aproximación I.
Aproximación II.
Aproximación III.
Medición el error
Ejercicio 4.
- ③ CALIBRACIÓN 3: CONDUCTIVIDAD VARIABLE
Ejercicio 5.
- ④ REFERENCIAS
- ⑤ CRÉDITOS

-  [1] Bergman, T.L. and Incropera, F.P. and DeWitt, D.P. and Lavine, A.S., *Fundamentals of Heat and Mass Transfer*, Wiley, **2011**.
-  [2] R.J. Leveque, *Finite Difference Method for Ordinary and Partial Differential Equations: Steady State and Time-Dependent Problems*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, **2007**.
-  [3] Y. Saad
Iterative Methods for Sparse Linear Systems.
PWS/ITP 1996.
Online: <http://www-users.cs.umn.edu/~saad/books.html>, **2000**
-  [4] Richard Burden and J. Douglas Faires
Numerical Analysis
Cengage Learning; 9 edition (August 9, **2010**)
-  [5] I. Herrera & G. F. Pinder,
Mathematical Modeling in Science and Engineering: An Axiomatic Approach, John Wiley **2012**.

CONTENIDO

- ① CALIBRACIÓN 1.
Ejercicio 3.
- ② CALIBRACIÓN 2: CONDICIONES TIPO NEUMMAN
Aproximación I.
Aproximación II.
Aproximación III.
Medición el error
Ejercicio 4.
- ③ CALIBRACIÓN 3: CONDUCTIVIDAD VARIABLE
Ejercicio 5.
- ④ REFERENCIAS
- ⑤ CRÉDITOS

Dr. Luis M. de la Cruz Salas

Departamento de Recursos Naturales

Instituto de Geofísica

Universidad Nacional Autónoma de México



Trabajo realizado con el apoyo del Programa UNAM-DGAPA-PAPIME PE101019

