

Universidad de Guanajuato

Método de Volumen Finito (Proyecto II)

García Sánchez Marco Antonio¹

¹División de Ingenierías, Campus Irapuato-Salamanca, Universidad de Guanajuato,
Salamanca, Gto.

Docente: Damián Ascencio César Eduardo

August 2019

Índice

1. Introducción	2
2. Ecuaciones Gobernantes	2
3. Discretización	2
3.1. Caso 1: UDS	3
3.1.1. Discretización para las esquinas	3
3.1.2. Discretización para las caras	4
3.2. Caso 2: CDS	4
3.2.1. Discretización para las esquinas	4
3.2.2. Discretización para las caras	5
4. Resultados e independencia de malla	5
4.1. Resultados UDS	5
4.1.1. $\Gamma = 0$	5
4.1.2. $\Gamma = 1$	7
4.2. resultados CDS	7
4.2.1. $\Gamma = 0$	7
4.2.2. $\Gamma = 1$	7
5. Conclusiones	7
Bibliografía	8

Resumen

En el presente trabajo se expone el método de volumen finito mediante dos esquemas para la discretización de la ecuación gobernante en el fenómeno de advección-difusión.

1. Introducción

Muchos problemas físicos pueden ser modelados analizando el balance de dos fenómenos: la advección y la difusión, los cuales se llevan a cabo de forma simultánea.

El primero se relaciona con el transporte de especies debido a la presencia de campos de velocidad, y el segundo se define como la dispersión de las especies involucradas en el proceso a lo largo del dominio físico del problema. El fenómeno de la advección-difusión es muy común en la naturaleza y en aplicaciones tanto industriales como de ingeniería; generalmente se denomina también como problema del transporte.

Si bien, el método del volumen finito consiste en dividir el dominio en pequeñas celdas de volumen finito y aplicar aquí las leyes de conservación de la energía relacionando el campo de nodos centrales con los nodos vecinos. Asimismo, este método debe incluir el efecto de los términos advectivos, los cuales están presentes junto con todo fenómeno convectivo de la naturaleza.

En este proyecto se incluyen los efectos de los términos advectivos y se propone el uso de dos esquemas diferentes para resolver este problema. Utilizaremos el esquema de discretización hacia adelante y el esquema de diferencias centrales para tomar en consideración dichos efectos.

2. Ecuaciones Gobernantes

Un esquema junto con los datos del problema que se atacará se muestran en la Figura 0.

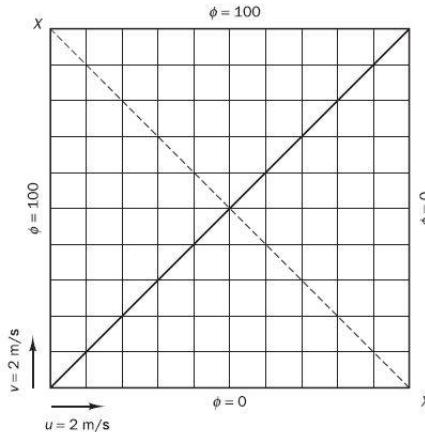


Figura 0. Esquema del problema.

La ecuación gobernante de advección-difusión en [1] está dada por la siguiente expresión:

$$\frac{\partial}{\partial x}(\rho u \varphi) + \frac{\partial}{\partial y}(\rho v \varphi) = \Gamma \frac{\partial^2}{\partial x^2} \varphi + \Gamma \frac{\partial^2}{\partial y^2} \varphi \quad (1)$$

3. Discretización

Discretizando la ecuación gobernante se obtiene

$$\begin{aligned} A_n(\rho V_n)\varphi_n + A_e(\rho U_e)\varphi_e - A_w(\rho U_w)\varphi_w - A_s(\rho V_s)\varphi_s = \\ = A_e\Gamma_e \frac{d\varphi}{dx}|_e - A_w\Gamma_w \frac{d\varphi}{dx}|_w + A_n\Gamma_n \frac{d\varphi}{dy}|_n - A_s\Gamma_s \frac{d\varphi}{dy}|_s \end{aligned} \quad (2)$$

Asumiendo

-Malla homogénea $dx = dy$; $A_n = A_w = A_s = A_e$

-También $F_i = (\varphi U)_i$ y $D_i = \frac{\Gamma_i}{dx}$

A partir de esto se puede obtener una ecuación gobernante discretizada

$$\begin{aligned} F_n\varphi_n + F_e\varphi_e - F_w\varphi_w - F_s\varphi_s = D_e(\varphi_e - \varphi_p) + \\ + D_n(\varphi_n - \varphi_p) - D_w(\varphi_p - \varphi_w) - D_s(\varphi_p - \varphi_s) \end{aligned} \quad (3)$$

3.1. Caso 1: UDS

El esquema de diferencias hacia adelante se presenta como:

$$u > 0; \varphi_e = \varphi_p \text{ y } \varphi_w = \varphi_W$$

$$v > 0; \varphi_n = \varphi_p \text{ y } \varphi_s = \varphi_S$$

Sustituyendo lo anterior en la ecuación 3 y factorizando, encontramos la ecuación para nodos centrales:

$$\begin{aligned} \varphi_p(F_e + F_n + D_e + D_w) + \varphi_w(-F_w - D_w) + \\ + \varphi_s(-F_s - D_s) + \varphi_e(-D_e) + \varphi(-D_n) = 0 \end{aligned} \quad (4)$$

3.1.1. Discretización para las esquinas

Nodo 1; usando nodo fantasma en w y s:

$$\begin{aligned} \varphi_p(F_e + F_n + F_s + D_e + 2D_w + 2D_s + D_n) + \varphi_e(-D_e) + \\ + \varphi_n(-D_n) = 2C_w(F_w + D_w) + 2C_s(F_s + D_s) \end{aligned} \quad (5)$$

Nodo n; usando nodo fantasma en E y S

$$\begin{aligned} \varphi_p(F_e + F_n + F_s + 2D_e + D_w + 2D_s + D_n) + \varphi_w(-F_w - D_w) + \\ + \varphi_n(-D_n) = 2C_e(D_e) + 2C_s(F_s + D_s) \end{aligned} \quad (6)$$

Nodo γ ; usando nodo fantasma en W y N

$$\begin{aligned} \varphi_p(F_e + F_n + F_w + D_e + 2D_w + D_s + 2D_n) + \varphi_e(-D_e) + \\ + \varphi_s(-F_s - D_s) = 2C_w(F_w + D_w) + 2C_n(D_n) \end{aligned} \quad (7)$$

Nodo n^2 ; usando nodo fantasma en E y N

$$\begin{aligned} \varphi_p(F_e + F_n + F_w + 2D_e + D_w + D_s + 2D_w) + \varphi_w(-D_w - F_w) + \\ + \varphi_s(-F_s - D_s) = 2C_n(D_n) + 2C_e(D_e) \end{aligned} \quad (8)$$

3.1.2. Discretización para las caras

Cara Sur: usando nodo fantasma en S

$$\begin{aligned} \varphi_p(F_e + F_n + F_s + D_e + D_w + 2D_s + D_n) + \varphi_w(-D_w \\ - F_w) + \varphi_n(-D_n) + \varphi_e(-D_e) = 2C_s(F_s + D_s) \end{aligned} \quad (9)$$

Cara Norte; usando nodo fantasma en N

$$\begin{aligned} \varphi_p(F_e + F_n + F_s + D_e + D_w + D_s + 2D_n) + \varphi_w(-D_w \\ - F_w) + \varphi_e(-D_e) + \varphi_s(-F_s - D_s) = 2C_n(D_n) \end{aligned} \quad (10)$$

Cara Oeste; usando nodo fantasma en W

$$\begin{aligned} \varphi_p(F_e + F_n + F_w + D_e + 2D_w + D_s + D_n) + \varphi_e(-D_e) \\ + \varphi_s(-F_s - D_s) + \varphi_n(-D_n) = 2C_w(F_w + D_w) \end{aligned} \quad (11)$$

3.2. Caso 2: CDS

El esquema de diferencias centrales, se presenta como:

$$\begin{aligned} \varphi_e &= \frac{\varphi_p + \varphi_E}{2} \quad y \quad \varphi_w = \frac{\varphi_p + \varphi_W}{2} \\ \varphi_n &= \frac{\varphi_p + \varphi_N}{2} \quad y \quad \varphi_s = \frac{\varphi_p + \varphi_S}{2} \end{aligned}$$

Sustituyendo lo anterior en la ecuación 3 y factorizando encontramos la ecuación para nodos centrales:

$$\begin{aligned} \varphi_p(F_e + F_n + D_e + D_w) + \varphi_w(-F_w - D_w) \\ + \varphi_s(-F_s - D_s) + \varphi_e(-D_e) + \varphi_n(-D_n) = 0 \end{aligned} \quad (12)$$

3.2.1. Discretización para las esquinas

Nodo 1; usando nodo fantasma en w y s:

$$\begin{aligned} \varphi_p\left(\frac{1}{2}(F_e + F_n) + D_e + D_w + 2D_s + 2D_n\right) + \varphi_e\left(\frac{1}{2}F_e - D_e\right) \\ + \varphi_n\left(\frac{1}{2}Fn - D_n\right) = 2C_w\left(\frac{1}{2}Fw + D_w\right) + 2C_s\left(\frac{1}{2}Fs + D_s\right) \end{aligned} \quad (13)$$

Nodo n; usando nodo fantasma en E y S

$$\begin{aligned} \varphi_p\left(\frac{1}{2}(F_e + F_n - F_s - Fw) + 2D_e + D_w + 2D_s + D_n\right) + \varphi_w\left(-\frac{1}{2}F_w\right. \\ \left.- D_w\right) + \varphi_n\left(\frac{1}{2}Fn - D_n\right) = 2C_e\left(-\frac{1}{2}Fe + D_e\right) + 2C_s\left(\frac{1}{2}Fs + D_s\right) \end{aligned} \quad (14)$$

Nodo γ ; usando nodo fantasma en W y N

$$\begin{aligned} \varphi_p\left(\frac{1}{2}(F_e - F_s) + D_e + 2D_w + D_s + 2D_n\right) + \varphi_e\left(\frac{1}{2}F_e - D_e\right) \\ + \varphi_s\left(-\frac{1}{2}Fs - D_s\right) = 2C_w\left(\frac{1}{2}F_w + D_w\right) + 2C_n\left(-\frac{1}{2} + D_n\right) \end{aligned} \quad (15)$$

Nodo n^2 ; usando nodo fantasma en E y N

$$\begin{aligned} \varphi_p\left(\frac{1}{2}(F_w - F_s) + 2D_e + D_w + D_s + 2D_n\right) + \varphi_w\left(-D_w - \frac{1}{2}F_w\right) \\ + \varphi_s\left(-\frac{1}{2}Fs - D_s\right) = 2C_n\left(-\frac{1}{2}Fn - D_n\right) + 2C_e\left(-\frac{1}{2}Fe + D_e\right) \end{aligned} \quad (16)$$

3.2.2. Discretización para las caras

Cara Sur: usando nodo fantasma en S

$$\begin{aligned} \varphi_p \left(\frac{1}{2}(F_e + F_n - F_w) + D_e + D_w + 2D_s + D_n \right) + \varphi_w \left(-D_w - \frac{1}{2}F_w \right) \\ + \varphi_n \left(\frac{1}{2}F_n - D_n \right) + \varphi_e \left(\frac{1}{2}F_e - D_e \right) = 2C_s \left(\frac{1}{2}F_s + D_s \right) \end{aligned} \quad (17)$$

Cara Este; usando nodo fantasma en E

$$\begin{aligned} \varphi_p \left(\frac{1}{2}(F_e + F_n - F_s) + D_e + D_w + D_s + 2D_n \right) + \varphi_w \left(-D_w - \frac{1}{2}F_w \right) \\ + \varphi_e \left(\frac{1}{2}F_e - D_e \right) + \varphi_s \left(-\frac{1}{2}F_s - D_s \right) = 2C_n \left(\frac{1}{2}F_w - D_n \right) \end{aligned} \quad (18)$$

Cara Norte; usando fantasma en N

$$\begin{aligned} \varphi_p \left(\frac{1}{2}(F_n - F_s - F_w) + D_e + D_w + D_s + 2D_n \right) + \varphi_e \left(\frac{1}{2}F_e - D_e \right) \\ + \varphi_s \left(-\frac{1}{2}F_s - D_s \right) + \varphi_w \left(\frac{1}{2}F_w - D_w \right) = 2C_w \left(-\frac{1}{2}F_w + D_w \right) \end{aligned} \quad (19)$$

4. Resultados e independencia de malla

Para la solución de las ecuaciones generadas en el método de volumen finito, es posible usar algunos métodos numéricos como el Gauss Seidel. Para la solución del problema propuesto se realizó un código en Phyton, el cual se muestra en Anexos, con el cual se resolverán las ecuaciones encontradas, así mismo se puede tener control sobre algunos datos como número de nodos y condiciones de frontera, así como otras restricciones, es posible observar las variaciones que existen a lo largo de la aleta según las condiciones que se estipulen.

Las propiedades y características del problema para la solución de manera computacional son mostrados en el Cuadro 1

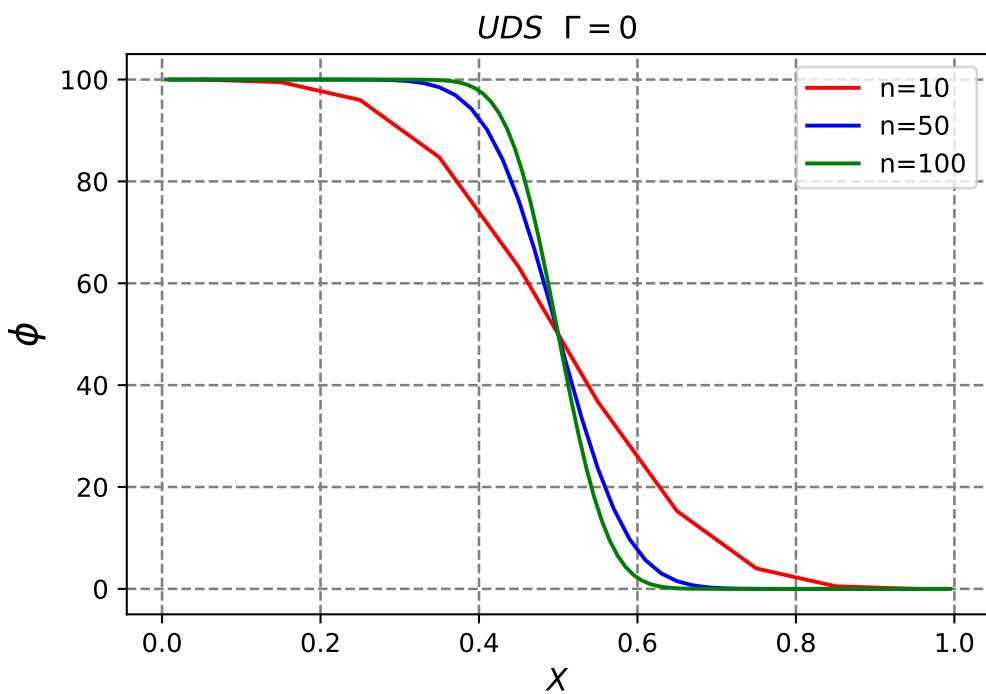
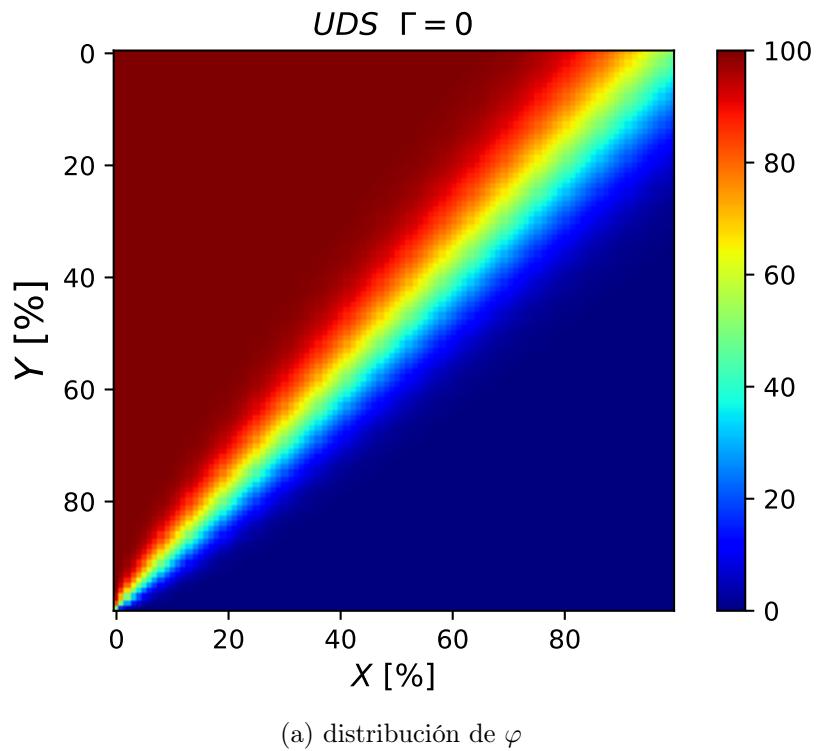
Cuadro 1: Propiedades del problema

Datos	valor	unidades
C_s	0	—
C_n	100	—
C_e	0	—
C_w	100	—
ρ	1	kgm^{-3}
Γ	1,0	—
L	1	m
u	2	ms^{-1}
v	2	ms^{-1}
$E_{propuesto}$	0.001	

4.1. Resultados UDS

4.1.1. $\Gamma = 0$

Resolviendo de manera numérica las ecuaciones discretizadas fue posible la construcción de un esquema en el cual es posible ver la distribución de la propiedad φ .

Figura 1: UDS para $\Gamma = 0$

En la figura 1 se muestra la distribución así como la un análisis de independencia de malla, en 1(a) se puede observar una solución esperada por la simetría del problema, en 1(b) es posible observar la independencia de

malla es buena a partir de 50×50 elementos teniendo un comportamiento de tendencia la solución con 100×100 elementos. Para la solución del problema se optó por usar una densidad de malla de 50×50 elementos.

4.1.2. $\Gamma = 1$

Usando la misma densidad de malla ya mencionada es posible obtener la distribución de φ cuando $\Gamma = 1$ la figura 2 muestra la independencia y distribución de la propiedad para este caso. De 2(b) se puede observar que la malla de 10×10 arroja resultados correctos ya que se ha graficado la diagonal de propiedad no constante, esto quiere decir que los resultados del problema con las condiciones ya mencionadas es una buena aproximación al resultado físico, además que era de esperarse nuevamente por la simetría del problema un comportamiento similar.

4.2. resultados CDS

4.2.1. $\Gamma = 0$

De las ecuaciones discretizadas para este esquema se puede obtener un sistema de ecuaciones singular a resolver, lo que significa que uno de los elementos de la diagonal es 0, este sistema de ecuaciones no fue posible resolverlo, por ende se concluye en esta sección que no existe solución numérica usando el esquema diferencias centrales cuando $\Gamma = 0$

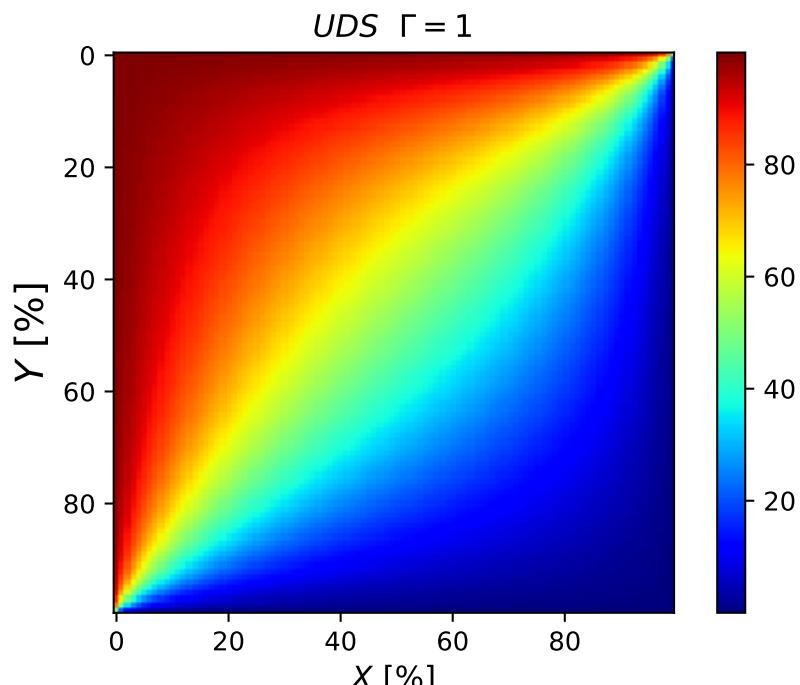
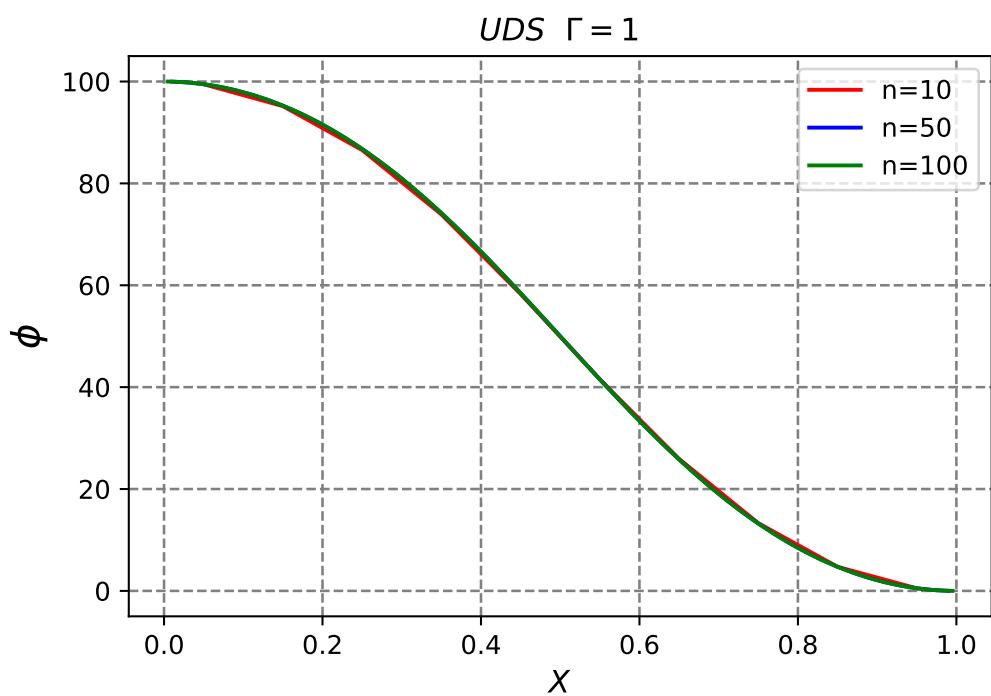
4.2.2. $\Gamma = 1$

Usando las ecuaciones de este caso se pudo obtener la figura 3 donde se muestra la independencia de malla así como la distribución de la propiedad. Como se puede constatar con las figuras 2 y 3, existe un comportamiento muy similar con las mallas cuando $\Gamma = 1$ lo cual es de esperarse ya que los dos esquemas deben proporcionar la misma solución con una malla adecuada.

5. Conclusiones

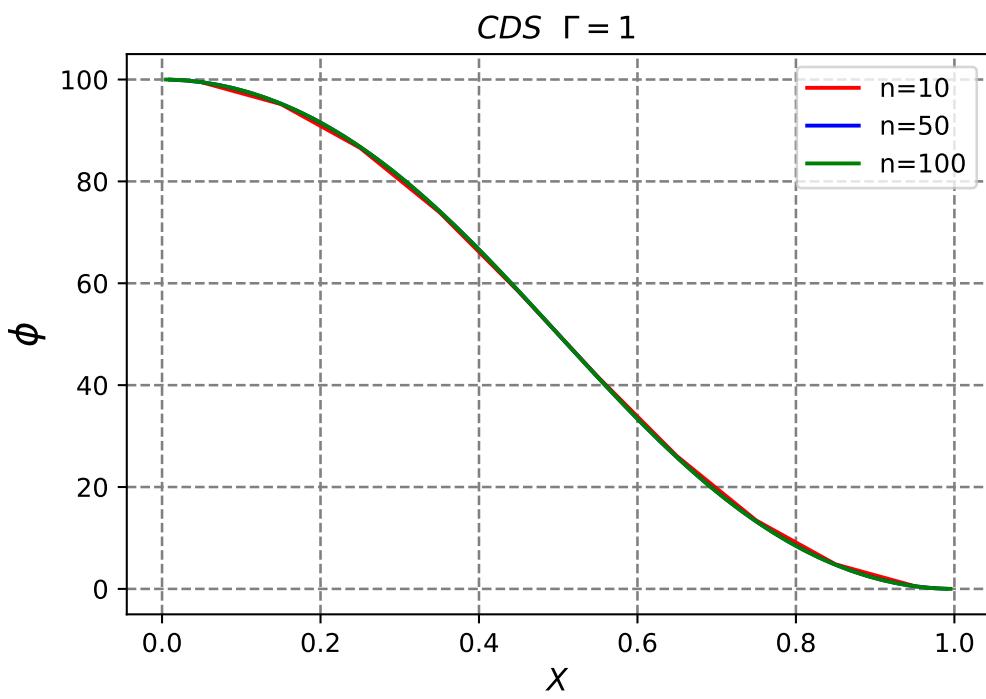
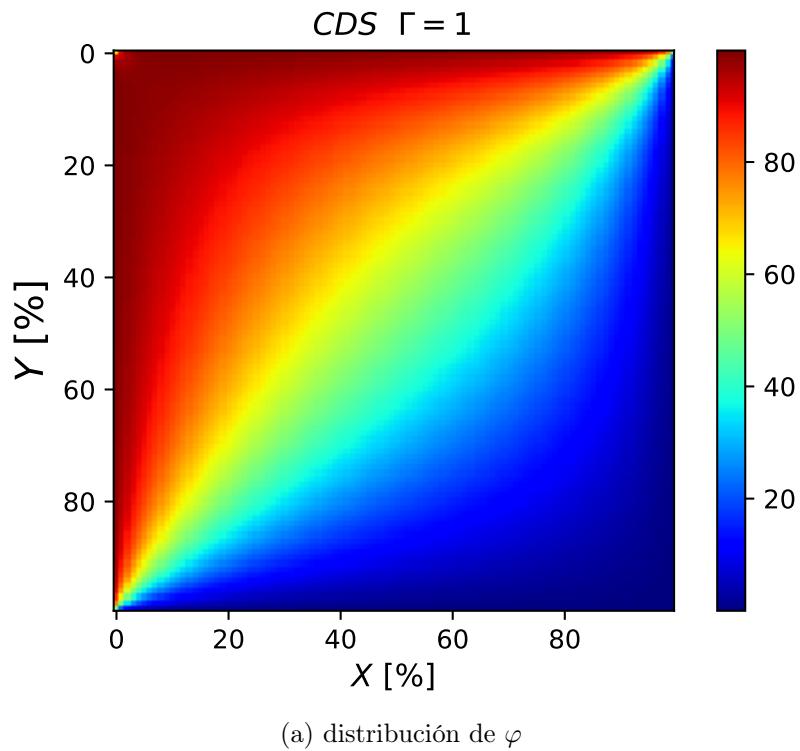
Como puede observarse en los resultados obtenidos la diferencia de un método a otro radica en su exactitud mientras en diferenciales centrales es de suma importancia conocer lo que pasa en las dos direcciones del nodo a analizarse y en diferencias hacia adelante solo es de interés las condiciones del nodo de interés y lo que se approxima delante de este.

Para el método de diferenciales hacia adelante se tiene el inconveniente que muestra resultados de una falsa difusión pero este problema se puede corregir aumentando el número de nodos y de este manera obtener número de Peclet relativamente bajo para de esta manera la solución del sistema sea más estable.

(a) distribución de φ 

(b) independencia de malla

Figura 2: UDS para $\Gamma = 1$

Figura 3: CDS para $\Gamma = 1$

Referencias

- [1] H. Versteeg and W. Malalasekera, “An introduction to computational fluid dynamics,” *Finite Volume Method, Essex, Longman Scientific & Technical*, 1995.

Appendix I: Solution Codes

```

1 print( '\033[H\033[J')
2 import numpy as np
3 import matplotlib.pyplot as plt
4 from Problema2 import Problema2
5 from scipy import interpolate
6
7
8 n=10
9 gamma=1
10 L=1
11 dx=dy=L/n
12
13 esquema='CDS'
14
15 if esquema=='UDS':
16     alpha=1
17     betha=0
18 if esquema=='CDS':
19     alpha=0
20     betha=1
21
22 Dcons=np.zeros(((n)),float)
23 Dphi=np.zeros(((n)),float)
24 x=np.zeros(((n)),float)
25
26
27 sol=Problema2(n,dx,np,gamma,alpha,betha)
28 for k in range(0,n):
29     Dcons[k]=sol[k*(n)+k]
30     Dphi[k]=sol[-k*n-n+k]
31     x[k]=dx/2+dx*k
32
33 n=50
34 dx=dy=L/n
35
36 Dcons2=np.zeros(((n)),float)
37 Dphi2=np.zeros(((n)),float)
38 x2=np.zeros(((n)),float)
39
40 sol2=Problema2(n,dx,np,gamma,alpha,betha)
41 for k in range(0,n):
42     Dcons2[k]=sol2[k*(n)+k]
43     Dphi2[k]=sol2[-k*n-n+k]
44     x2[k]=dx/2+dx*k
45
46 n=100
47 dx=dy=L/n
48
49 Dcons3=np.zeros(((n)),float)
50 Dphi3=np.zeros(((n)),float)
51 x3=np.zeros(((n)),float)
52
53 sol3=Problema2(n,dx,np,gamma,alpha,betha)
54 for k in range(0,n):
55     Dcons3[k]=sol3[k*(n)+k]
56     Dphi3[k]=sol3[-k*n-n+k]
57     x3[k]=dx/2+dx*k
58 #

```

```

59 #
60 A=np.zeros(((n,n)),float)
61 arr = sol3.reshape((n,n))
62 for m in range (0,n):
63     A[m]=arr [n-1-m]
64
65 plt.figure (1)
66 plt.plot(x,Dphi, label = "n=10", color = 'red')
67 plt.plot(x2,Dphi2, label = "n=50", color = 'blue')
68 plt.plot(x3,Dphi3, label = "n=100", color = 'green')
69 plt.legend(loc="upper right")
70 plt.hold(True)
71 plt.grid(True)
72 plt.grid(color = '0.5', linestyle = '—', linewidth = 1)
73 plt.xlabel(r"$X$ ", fontsize = 12, color = 'black')
74 plt.ylabel(r"$\phi$ ", fontsize = 15, color = 'black')
75 plt.title('CDS \ \ Gamma=1$', fontsize = 12, color = 'black', verticalalignment = 'baseline',
    horizontalalignment = 'center')
76 plt.savefig('CDS_Gamma=1.pdf')
77 plt.show()
78
79
80 xm=np.arange(dx/2,L,dx)
81 ym=np.arange(dx/2,L,dx)
82 xx,yy=np.meshgrid(xm,ym)
83 f=interpolate.interp2d(xm,ym,sol)
84 im = plt.imshow(A,cmap='jet')
85 plt.colorbar()
86 plt.hsv
87 plt.xlabel(r"$X\ [\%]" , fontsize = 12, color = 'black')
88 plt.ylabel(r"$Y\ [\%]" , fontsize = 15, color = 'black')
89 plt.title('CDS \ \ Gamma=1$', fontsize = 12, color = 'black', verticalalignment = 'baseline',
    horizontalalignment = 'center')
90 plt.savefig('CDS_Gamma=1_Dis.pdf')

1
2 def Problema2(n,dx,np,gamma,alpha,betha):
3     #nodosX=nodosY
4     nt=n**2 #nodos totales del sistema
5     error=.001 #error propuesto
6     A=np.zeros(((nt,nt)),float)
7     X=np.zeros(((nt)),float)
8     B=np.zeros(((nt)),float)
9     Xc=np.zeros(((nt)),float)
10    E=np.zeros(((nt)),float)
11    Ln=np.zeros(((nt)),float)
12    A2=np.zeros(((n,n)),float)      #matriz de visualizacion de malla
13
14    rho=1
15    u=v=2
16    Fe=Fw=rho*u
17    Fn=Fs=rho*v
18    De=Dw=Ds=Dn=gamma/dx
19    print(Dn)
20
21    Cs=0
22    Ce=0
23    Cn=100
24    Cw=100
25

```

```

26
27     for i in range(0,nt):
28         #nodos de las esquinas
29         if i==0:                      #nodo inferior izquierdo
30             A[i,i]=alpha*(Fn+Fe+Fw+Fs+De+2*Dw+Dn+2*Ds)+beta*(.5*(Fn+Fe-Fs-Fw+Fs+Fw)+De+Dn+2*Ds
31 +2*Dw)
32             A[i,n]=alpha*(-Dn)+beta*(.5*Fn-Dn)
33             A[i,i+1]=alpha*(-De)+beta*(.5*Fe-De)
34             B[i]=alpha*((Fw+Dw)*2*Cw+(Fs+Ds)*2*Cs)+beta*(2*Cs*(.5*Fs+Ds)+2*Cw*(.5*Fw+Dw))
35         if i==(n-1):                  #nodo inferior derecha
36             A[i,i]=alpha*(Fn+Fe+Fs+2*De+Dw+2*Ds+Dn)+beta*(.5*(Fn+Fe-Fs-Fw+Fs-Fe)+2*De+Dn+Dw+2*
37 Ds)
38             A[i,2*n-1]=alpha*(-Dn)+beta*(.5*Fn-Dn)
39             A[i,i-1]=alpha*(-Fw-Dw)+beta*(-.5*Fw-Dw)
40             B[i]=alpha*((De)*2*Ce+(Fs+Ds)*2*Cs)+beta*(2*Cs*(.5*Fs+Ds)+2*Ce*(-.5*Fe+De))
41
42         if i==((n-1)*n):            #nodo superior izquierda
43             A[i,i]=alpha*(Fe+Fn+Fw+De+2*Dw+2*Dn+Ds)+beta*(.5*(Fn+Fe-Fw-Fs-Fn+Fw)+De+2*Dn+Ds+2*
44 Dw)
45             A[i,i+1]=alpha*(-De)+beta*(.5*Fe-De)
46             A[i,i-n]=alpha*(-Fs-Ds)+beta*(-.5*Fs-Ds)
47             B[i]=alpha*((Fw+Dw)*2*Cw+(Dn)*2*Cn)+beta*(2*Cs*(-.5*Fe+Dn)+2*Cw*(.5*Fw+Dw))
48
49         if i===(nt-1):              #nodo superior derecha
50             A[i,i]=alpha*(Fe+Fn+2*De+Dw+2*Dn+Ds)+beta*(.5*(Fn+Fe-Fs-Fw-Fn-Fe)+2*De+2*Dn+Ds+Dw)
51             A[i,i-1]=alpha*(-Dw-Fw)+beta*(-.5*Fw-Dw)
52             A[i,i-n]=alpha*(-Ds-Fs)+beta*(-.5*Fs-Ds)
53             B[i]=alpha*((Dn)*2*Cn+(De)*2*Ce)+beta*(2*Cn*(-.5*Fn+Dw)+2*Ce*(-.5*Fe+De))
54
55         #nodos de las fronteras
56         if i>0 and i<n-1:        #frontera sur
57             A[i,i]=alpha*(Fn+Fe+Fs+De+Dw+Dn+2*Ds)+beta*(.5*(Fn+Fe-Fs-Fw+Fs)+De+Dn+Dw+2*Ds)
58             A[i,i+1]=alpha*(-De)+beta*(.5*Fe-De)
59             A[i,i+(n)]=alpha*(-Dn)+beta*(.5*Fn-Dn)
60             A[i,i-1]=alpha*(-Fw-Dw)+beta*(-.5*Fw-Dw)
61             B[i]=alpha*((Fs+Ds)*2*Cs)+beta*(2*Cs*(.5*Fs+Ds))
62
63         #nodos centrales...
64         for j in range(1,(n-1)):
65             A[j+n*i,j+n*i]=alpha*(Fn+Fe+De+Dw+Dn+Ds)+beta*(.5*(Fn+Fe-Fs-Fw)+De+Dn+Ds+Dw)
66             A[j+n*i,j+n*i+1]=alpha*(-De)+beta*(.5*Fe-De)
67             A[j+n*i,j+n*i-1]=alpha*(-Fw-Dw)+beta*(-.5*Fw-Dw)
68             A[j+n*i,j+n*i-n]=alpha*(-Fs-Ds)+beta*(-.5*Fs-Ds)
69             A[j+n*i,j+n*i+n]=alpha*(-Dn)+beta*(.5*Fn-Dn)
70
71         if i<n-1 and i>0:        #frontera oeste
72             A[i*n,i*n]=alpha*(Fn+Fe+Fw+De+2*Dw+Dn+Ds)+beta*(.5*(Fn+Fe-Fs-Fw+Fw)+De+Dn+Ds+2*Dw)
73             A[i*n,i*n+n]=alpha*(-Dn)+beta*(.5*Fn-Dw)
74             A[i*n,i*n+1]=alpha*(-De)+beta*(.5*Fe-De)
75             A[i*n,i*n-n]=alpha*(-Fs-Ds)+beta*(-.5*Fs-Ds)
76             B[i*n]=alpha*(2*Cw*(Fw+Dw))+beta*(2*Cw*(.5*Fw+Dw))
77
78         if i<n-1 and i>0:        #frontera este
79             A[i*n+n-1,i*n+n-1]=alpha*(Fn+Fe+2*De+Dw+Dn+Ds)+beta*(.5*(Fn+Fe-Fs-Fw-Fe)+Dn+Ds+Dw
80 +2*De)
81             A[i*n+n-1,i*n+2*n-1]=alpha*(-Dn)+beta*(.5*Fn-Dn)

```

```
82
83     if i>(nt-n) and i<nt-1:    #frontera norte
84         A[i,i]=alpha*(Fe+Fn+De+Dw+2*Dn+Ds)+beta*(.5*(Fn+Fe-Fs-Fw-Fn)+Dn+Ds+Dw+2*Dn)
85         A[i,i-1]=alpha*(-Fw-Dw)+beta*(-.5*Fw-Dw)
86         A[i,i+1]=alpha*(-De)+beta*(.5*Fe-De)
87         A[i,i-n]=alpha*(-Ds-Fs)+beta*(-.5*Fs-Ds)
88         B[i]=alpha*(2*Cn*(Dn))+beta*(2*Cn*(-.5*Fn+Dn))
89
90 #X=gauss_seidel(A,B,Xc,E,n,error,np)
91 AA=np.linalg.inv(A)
92 sol=np.dot(AA,B)
93 print(A)
94 print(B)
95 return(sol)
```