Assignment 7

Shashank R – 16ME269

Python code for solving ODE

2nd order approximation – CDS

1st order approximation – CDS

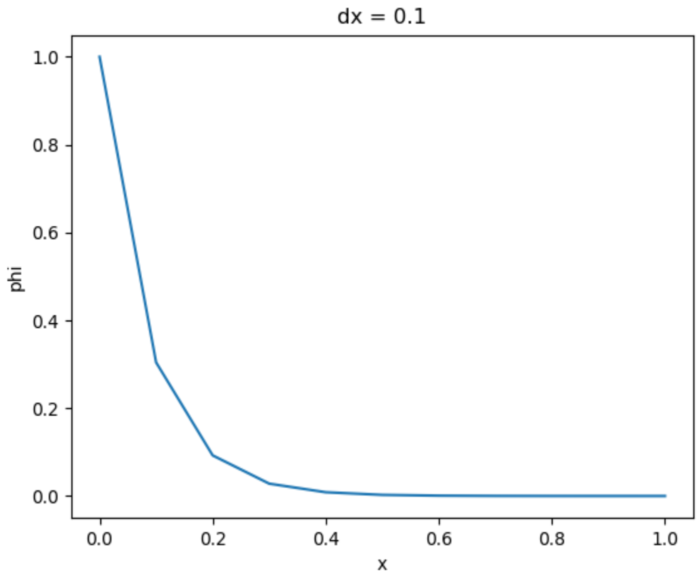
```
import numpy as np

import matplotlib.pyplot as plt

import time


x0 = 0

xn = 1


dx = 0.1

#dx = 0.01


n = 10 #Choose when dx = 0.1

#n = 100 #Choose when dx = 0.01

x = np.linspace(x0,xn,n+1)

a = np.zeros(n+1)

b = np.zeros(n+1)

c = np.zeros(n+1)

d = np.zeros(n+1)

phi = np.zeros(n+1)


#boundary conditions

phi[0] = 1.0

phi[n] = 0.0
```

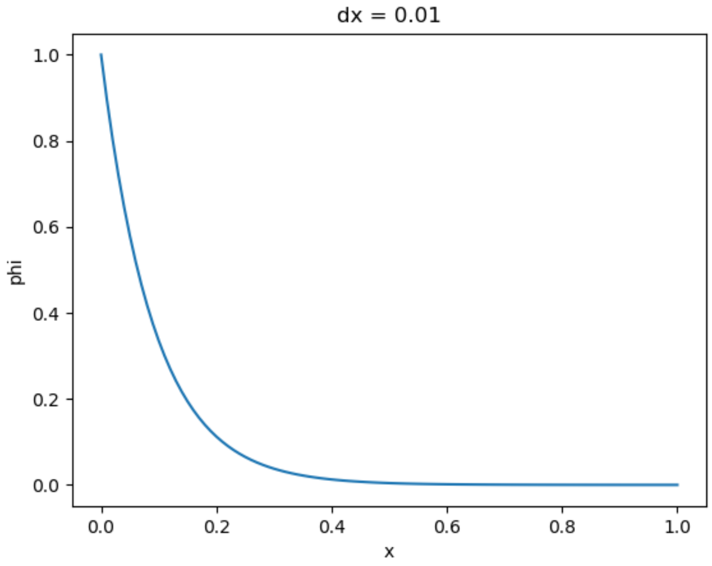
```
#Calculating coefficients for the first and last node
b[1] = -0.2/(dx*dx) - 1
c[1] = 0.1/(dx*dx) + 1/(2*dx)
d[1] = phi[0]*(1/(2*dx)-0.1/(dx*dx))
a[n-1] = 0.1/(dx*dx) - 1/(2*dx)
b[n-1] = -0.2/(dx*dx) - 1
d[n-1] = phi[n]*(-1/(2*dx)-0.1/(dx*dx))

#Calculating coefficients for the rest
for i in range(2,n-1):
     a[i] = 0.1/(dx*dx) - 1/(2*dx)
    b[i] = -0.2/(dx*dx) - 1
    c[i] = 0.1/(dx*dx) + 1/(2*dx)

#Thomas algorithm
for i in range(2,n):
    a[i] = a[i]/b[i-1]
    b[i] = b[i] - a[i]*c[i-1]
    d[i] = d[i] - a[i]*d[i-1]

phi[n-1] = d[n-1] / b[n-1]
for i in range(n-2,0,-1):
    phi[i]= (d[i] - c[i]*phi[i+1])/b[i]

plt.plot(x, phi)
plt.title("dx = 0.1")
plt.ylabel('phi')
plt.xlabel('x')
plt.show()
```

```
print('CPU time:',time.process_time(),'s')
```

Output

CPU time: 1.078125 s





2nd order approximation – CDS

1st order approximation – FDS

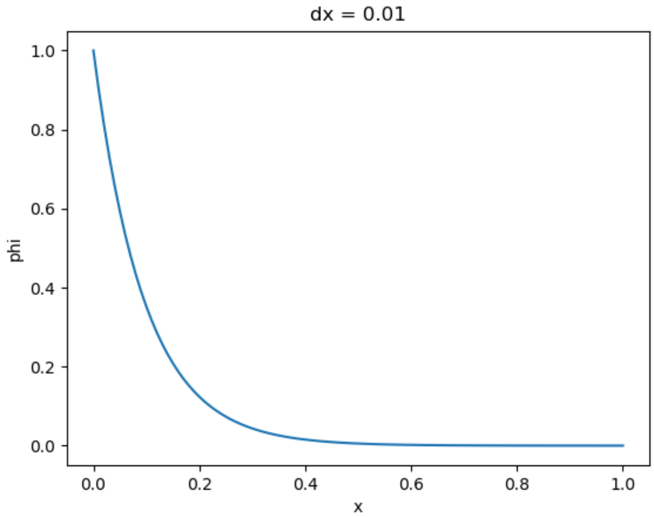
```
import numpy as np

import matplotlib.pyplot as plt

import time


x0 = 0

xn = 1

dx = 0.01

n = 100


x = np.linspace(x0,xn,n+1)

a = np.zeros(n+1)

b = np.zeros(n+1)

c = np.zeros(n+1)

d = np.zeros(n+1)
```

```
phi = np.zeros(n+1)

#boundary conditions
phi[0] = 1.0
phi[n] = 0.0

#Calculating coefficients for the first and last node
b[1] = -0.2/(dx*dx) - 1- 1/dx
c[1] = 0.1/(dx*dx) + 1/(dx)
d[1] = phi[0]*(-0.1/(dx*dx))
a[n-1] = 0.1/(dx*dx)
b[n-1] = -0.2/(dx*dx) - 1 - 1/dx
d[n-1] = phi[n]*(-1/(dx)-0.1/(dx*dx))

#Calculating coefficients for the rest
for i in range(2,n-1):
    a[i] = 0.1/(dx*dx)
    b[i] = -0.2/(dx*dx) - 1 - 1/dx
    c[i] = 0.1/(dx*dx) + 1/(dx)

#Thomas algorithm
for i in range(2,n):
    a[i] = a[i]/b[i-1]
    b[i] = b[i] - a[i]*c[i-1]
    d[i] = d[i] - a[i]*d[i-1]

phi[n-1] = d[n-1] / b[n-1]
for i in range(n-2,0,-1):
    phi[i]= (d[i] - c[i]*phi[i+1])/b[i]
```

```
plt.plot(x, phi)

plt.title("dx = 0.1")

plt.ylabel('phi')

plt.xlabel('x')

plt.show()

print('CPU time:',time.process_time(),'s')
```

Output

CPU time: 1.046875 s



2^{nd} order approximation – CDS

1^{st} order approximation – BDS

```
import numpy as np

import matplotlib.pyplot as plt

import time


x0 = 0

xn = 1

dx = 0.01

n = 100
```

```
x = np.linspace(x0,xn,n+1)
a = np.zeros(n+1)
b = np.zeros(n+1)
c = np.zeros(n+1)
d = np.zeros(n+1)
phi = np.zeros(n+1)

#boundary conditions
phi[0] = 1.0
phi[n] = 0.0

#Calculating coefficients for the first and last node
b[1] = -0.2/(dx*dx) - 1 + 1/dx
c[1] = 0.1/(dx*dx)
d[1] = phi[0]*(1/dx -0.1/(dx*dx))
a[n-1] = 0.1/(dx*dx) - 1/dx
b[n-1] = -0.2/(dx*dx) - 1 + 1/dx
d[n-1] = phi[n]*(-0.1/(dx*dx))

#Calculating coefficients for the rest
for i in range(2,n-1):
         a[i] = 0.1/(dx*dx) -  1/dx
        b[i] = -0.2/(dx*dx) - 1 + 1/dx
        c[i] = 0.1/(dx*dx)

#Thomas algorithm
for i in range(2,n):
        a[i] = a[i]/b[i-1]
        b[i] = b[i] - a[i]*c[i-1]
        d[i] = d[i] - a[i]*d[i-1]
```

```
phi[n-1] = d[n-1] / b[n-1]

for i in range(n-2,0,-1):

        phi[i]= (d[i] - c[i]*phi[i+1])/b[i]


plt.plot(x, phi)

plt.title("dx = 0.1")

plt.ylabel('phi')

plt.xlabel('x')

plt.show()

print('CPU time:',time.process_time(),'s')
```

Output

CPU time: 1.15625 s