Applied Computational Methods in Mechanical Sciences

(ME466)

Assignment 8

Himanshu Kumar

16ME234

October 22, 2019

**Problem Statement:**

Obtain the variation of vs, given the governing differential equation as:

Using central finite difference scheme for the given first order and second order derivatives. Use non- uniform grid using geometric progression:

1. For,.
2. For,.

where represents the number of grid points and is the ratio for geometric progression.

The boundary conditions are: and .

Value of constants used:

**Python Code:**

import time

import matplotlib.pyplot as plt

def grid\_1d\_gp(r,n):

x = [0 for i in range(n+1)]

rng=(0,1)

#gp sum

sum=0

for i in range(n):

sum = sum + pow(r,i)

x0 = (rng[1]-rng[0])/sum

x[0] = rng[0]

for i in range(n):

x[i+1] = x[i] + x0\*pow(r,i)

return(x)

def matrix\_form(x,bc):

p=1

u=1

t=0.02

n = len(x)-2

print(n)

a = [[0 for i in range(n)] for j in range(n)]

#rhs vector

b = [0 for i in range(n)]

for i in range(1,n+1):

gamma = ( (-p\*u)/(x[i+1]-x[i-1]) ) - ( (2\*t)/((x[i]-x[i-1])\*(x[i+1]-x[i-1])) )

alpha = ( (2\*t)/((x[i+1]-x[i-1])\*(x[i+1]-x[i])) ) + ( (2\*t)/((x[i+1]-x[i-1])\*(x[i]-x[i-1])) )

beta = ( (p\*u)/(x[i+1]-x[i-1]) ) - ( (2\*t)/((x[i+1]-x[i])\*(x[i+1]-x[i-1])) )

j=i-1

if(j is 0):

a[0][0]=alpha

a[0][1] =beta

b[0] = 0 - (gamma\*bc[0])

elif(j is n-1):

a[n-1][n-1]=alpha

a[n-1][n-2]=gamma

b[n-1] = 0 - (beta\*bc[1])

else:

a[j][j-1] = gamma

a[j][j] = alpha

a[j][j+1] = beta

return(a,b)

def gauss\_elm(A,B):

n= len(B)

# step 1: Gaussian elimination.

i=0

while i < n:

# pivots

pivot = A[i][i]

j=i+1

while j<n:

r = A[j][i]/pivot

# row opreation

k=i

while k<n:

A[j][k] = A[j][k] - A[i][k]\*r

k=k+1

B[j]=B[j]-B[i]\*r

j=j+1

i=i+1

#Back Substitution from nth row

x= [0 for i in range(n)]

i = n-1

x[i] = B[i]/A[i][i]

i=i-1

while i>=0:

sum = 0

k=i+1

while k<n:

sum = sum + A[i][k]\*x[k]

k=k+1

x[i]=(B[i]-sum)/A[i][i]

i=i-1

return(x)

def solver(r,n):

bc= (0,1)

x = grid\_1d\_gp(r,n)

a,b = matrix\_form(x,bc)

phi = gauss\_elm(a,b)

mod\_phi = [0 for i in range(n+1)]

for i in range(len(phi)):

mod\_phi[i+1] = phi[i]

mod\_phi[0] = bc[0]

mod\_phi[n]= bc[1]

print ("\n CPU time: ", time.process\_time(),'s')

return(mod\_phi,x)

def plotter(phi,x):

#plotting

plt.plot(x,phi,'r.')

plt.xlabel('x')

plt.ylabel('phi')

plt.title('phi vs. x')

plt.show()

#main

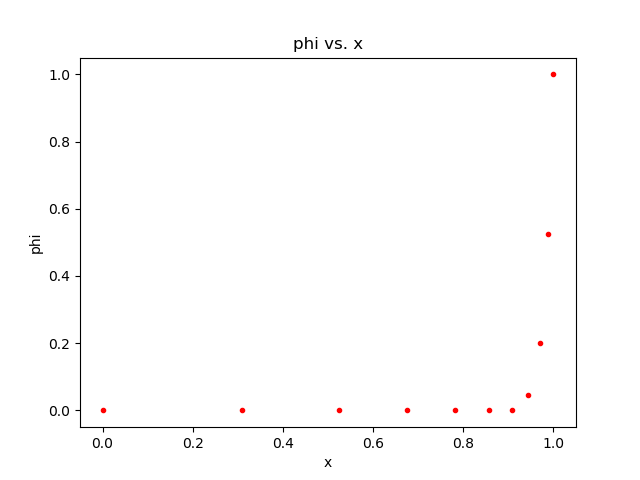
p,x = solver(0.7,10)

plotter(p,x)

**Results:**

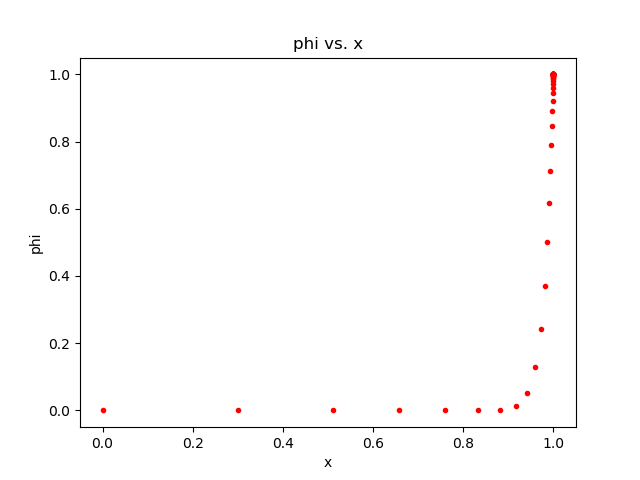
1. **With r=0.7 and :**

CPU time: 0.6875 s



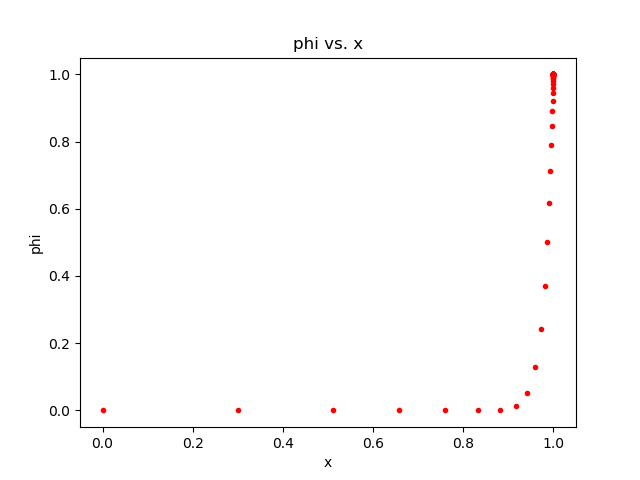
1. **With r=0.7 and :**

CPU time: 0.65625 s



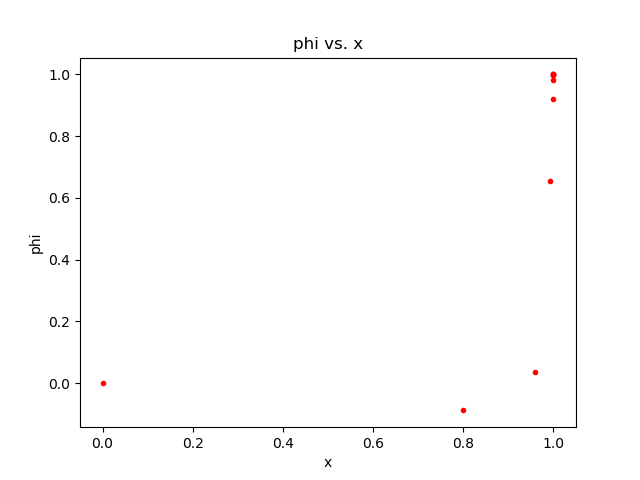
1. **With r=0.7 and :**

CPU time: 0.6875 s



1. **With r=0.2 and :**

CPU time: 0.375 s



1. **With r=0.2 and :**

**Truncation error, too fine grid at x~1.0**

1. **With r=0.2 and :**

**Truncation error, too fine grid at x~1.0**

**Additional Result:**

**With r=0.95 and:**

