***LAB EXPERIMENTS FOR XDIFFERENTIAL AND DIFFERENCE EQUATIONS***

***AAKASH-AGGARWAL***

***18BLC1060***

**Python 3.7**

**Compiler: Anaconda Navigator/Spyder version 4.0.1**

1.Computing the harmonics  of  a continuous real-valued function

CODE:

from array import array

import numpy as np

import math

from sympy import \*

import matplotlib.pyplot as plt

x=symbols('x')

p=float(input('Enter the period length(in integer or float type only):'))

print('p',p)

l=p/2

n = int(input("Enter number of elements in vectors : "))

# Below line read inputs from user using map() function

X = list(map(float,input("\nEnter the X-vector : ").strip().split()))[:n]

Y= list(map(float,input("\nEnter the Y-vector : ").strip().split()))[:n]

#print("\nList is - ", X)

#print(Y)

N=len(X)

print(N)

r=int(input('Enter the number of terms in series:'))

a\_0=(2/N)\*sum(Y);

a=[]

b=[]

H=[]

add1=[]

add2=[]

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

    sum1=0

    sum2=0

    for j in range(N):

        sum1=sum1+(Y[j]\*math.cos(i\*math.pi\*X[j]/l))

        sum2=sum2+(Y[j]\*math.sin(i\*math.pi\*X[j]/l))

    add1.append(sum1)

    add2.append(sum2)

#print('add1',add1)

#print('add1',add2)

for i in range(r):

    a.append((2/N)\*add1[i])

    b.append((2/N)\*add2[i])

#print('a',a)

#print('b',b)

for i in range(r):

    H.append(a[i]\*cos((i+1)\*math.pi\*x/l) + b[i]\*sin((i+1)\*math.pi\*x/l))

print('H',H)

HS=(a\_0)/2+sum(H);

#print('Harmonic series is given by',HS)

HS=simplify(HS)

print('Harmonic series is given by',HS)

u = np.arange(0, p+1,0.1)

v=[]

for i in range(len(u)):

    v.append(HS.subs(x,u[i]))

plt.plot(u,v)

plt.plot(X,Y,'ro')

plt.xlim([0, p])

plt.show()

OUTPUT:

Enter the period length:6

p 6.0

Enter number of elements : 6

Enter the X-vector : 0 1 2 3 4 5

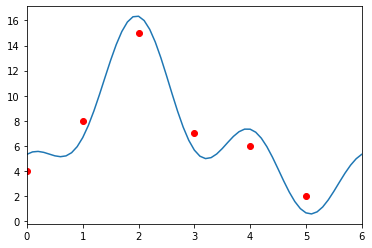
Enter the Y-vector : 4 8 15 7 6 2

6

Enter the number of terms in series:3

H [4.33012701892219\*sin(1.0471975511966\*x) - 2.83333333333333\*cos(1.0471975511966\*x), -0.866025403784437\*sin(2.0943951023932\*x) - 1.5\*cos(2.0943951023932\*x), 5.71914493359824e-16\*sin(3.14159265358979\*x) + 2.66666666666667\*cos(3.14159265358979\*x)]

Harmonic series is given by 4.33012701892219\*sin(1.0471975511966\*x) - 0.866025403784437\*sin(2.0943951023932\*x) + 5.71914493359824e-16\*sin(3.14159265358979\*x) - 2.83333333333333\*cos(1.0471975511966\*x) - 1.5\*cos(2.0943951023932\*x) + 2.66666666666667\*cos(3.14159265358979\*x) + 7.0



2. Power series of a ODE ( ordinary point)

CODE:

from sympy import \*

import sympy as sym

from sympy.solvers.solveset import linsolve

x,r,d\_0,c\_0,c\_1,c\_2,c\_3,c\_4,c\_5=symbols('x r d\_0 c\_0 c\_1 c\_2 c\_3 c\_4 c\_5')

print('Input the coefficients of DE as constants or function of x')

p1=input('Coefficient of D2y:')

p2=input('Coefficient of Dy:')

p3=input('Coefficient of y:')

c=[c\_0,c\_1,c\_2,c\_3,c\_4,c\_5]

p1=sympify(p1)

p2=sympify(p2)

p3=sympify(p3)

y=0

for i in range(0,6):

    y=y+c[i]\*x\*\*(i)

print('y=',y)

y=sympify(y)

dy=diff(y,x)

d2y=diff(dy,x)

ode=p1\*d2y+p2\*dy+p3\*y

ode=expand(ode)

print('ode',ode)

ps=collect(ode,x)

ps=simplify(ps)

print('ps',ps)

d=[]

for i in range (6+1):

    d.append(ps.coeff(x,i))

print('d',d)

fin\_list = solve((d[0],d[1],d[2],d[3]),(c\_2,c\_3,c\_4,c\_5))

c = fin\_list

z=y.subs(c\_2,c[c\_2])

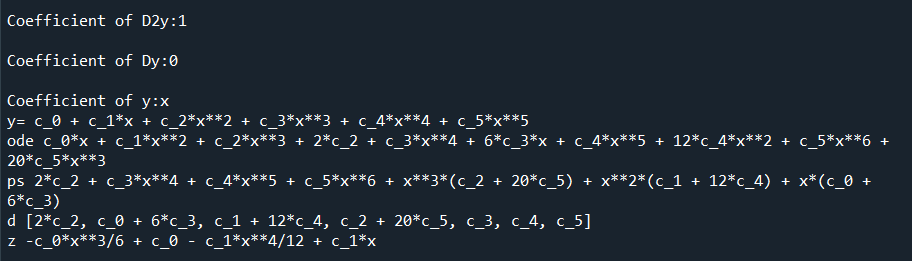
z=z.subs(c\_3,c[c\_3])

z=z.subs(c\_4,c[c\_4])

z=z.subs(c\_5,c[c\_5])

print('z',z)

OUTPUT:



3. Power series of an ODE about a regular singular point

**CODE:**

from sympy import \*

import sympy as sym

x,r,d\_0,c\_0,c\_1,c\_2,c\_3,c\_4,c\_5=symbols('x r d\_0 c\_0 c\_1 c\_2 c\_3 c\_4 c\_5')

print('Input the coefficients of DE as constants or function of x')

p1x=input('Coefficient of D2y:')

p1x=sympify(p1x)

p2x=input('Coefficient of Dy:')

p2x=sympify(p2x)

p3x=input('Coefficient of y:')

p3x=sympify(p3x)

c=[c\_0,c\_1,c\_2,c\_3,c\_4,c\_5]

y=0

#p=0

for i in range(0,6):

    y=y+c[i]\*x\*\*(i)

print('y=',y)

Px=p2x/p1x

print('Px=',Px)

Qx=p3x/p1x

print('Qx=',Qx)

px=simplify(x\*Px)

print('px=',px)

qx=simplify((x\*\*2)\*Qx)

print('qx=',qx)

a\_0=px.subs(x,0)

print('a\_0',a\_0)

b\_0=qx.subs(x,0)

print('b\_0',b\_0)

expr=r\*(r-1)+a\_0\*r+b\_0

sol=solve(expr,r)

print('sol',sol)

print(sol[0])

d=[]

y\_y=[]

for i in range(2):

    y1=y\*x\*\*sol[i]

    y1=expand(y1)

    print('y1',y1)

    y\_y.append(y1)

    dy1=diff(y1,x)

    d2y1=diff(dy1,x)

    ode1=p1x\*d2y1+p2x\*dy1+p3x\*y1

    ode1=ode1/x\*\*sol[i];

    ode1=expand(ode1)

    #print('ode1',ode1)

    ps1=collect(ode1,x)

    ps1=simplify(ps1)

    #print('ps1',ps1)

    d1=[]

    for i in range (6+1):

        d1.append(ps1.coeff(x,i))

    #print('d1',d1)

    d.append(d1)

print(' ')

print('d',d)

fin\_list = solve((d[0][0],d[0][1],d[0][2],d[0][3],d[0][4]),(c\_1,c\_2,c\_3,c\_4,c\_5))

c1 = fin\_list

z1= y\_y[0].subs(c\_1,c1[c\_1])

z1=z1.subs(c\_2,c1[c\_2])

z1=z1.subs(c\_3,c1[c\_3])

z1=z1.subs(c\_4,c1[c\_4])

z1=z1.subs(c\_5,c1[c\_5])

print(' ')

print('The particular solution of the given ODE around x=0 is given by: ')

print('z1',z1)

print(' ')

fin\_list = solve((d[1][0],d[1][1],d[1][2],d[1][3],d[1][4]),(c\_1,c\_2,c\_3,c\_4,c\_5))

c2 = fin\_list

z2=y\_y[1].subs(c\_1,c2[c\_1])

z2=z2.subs(c\_2,c2[c\_2])

z2=z2.subs(c\_3,c2[c\_3])

z2=z2.subs(c\_4,c2[c\_4])

z2=z2.subs(c\_5,c2[c\_5])

print('The particular solution of the given ODE around x=0 is given by: ')

print('z2',z2)

**OUTPUT:**

