```
# Ecuaciones Diferenciales Parciales
# Hiperbólica. Método explicito
import numpy as np
import sympy as sym
import matg1013 as mat

# INGRESO
x0 = 0
xn =1
1  # Longitud de cuerda
y0 = 0
yn = 1 # Puntos de amarre
t0 = 0
c = 1
po=0.9
Ta=po
tb=po
tc=lambda x,y: po*np.cos(2*np.pi*x)

# discretiza

tramosx = 10
tramost = 20

dx = (xn-x0)/tramosx
dt = dx/c

# PROCEDIMIENTO

xi = np.arange(x0,xn+dx,dx)
tj = np.arange(0,tramost*dt+dt,dt)
n = len(xi)
m = len(tj)
```

```
u = np.zeros(shape=(n,m),dtype=float)
u[:,0] = tc(xi,tj[0])
u[0,:] = po
u[n-1,:] = po

# Aplicando condición inicial

j = 0
for i in range(1,n-1,1):
    u[i,j+1] = (u[i+1,j]+u[i-1,j])/2

# Para los otros puntos
for j in range(1,m-1,1):
    for i in range(1,n-1,1):
        u[i,j+1] = u[i+1,j]-u[i,j-1]+u[i-1,j]

# SALIDA
np.set_printoptions(precision=2)
print('xi =')
print(xi)
print('tj =')
print(tj)
print('matriz u =')
print(u)

# **** GRÁFICO CON ANIMACION ***********
import matplotlib.animation as animation
import matplotlib.pyplot as plt

# Inicializa parametros de trama/foto
retardo = 300   # milisegundos entre tramas
tramas = m
maximoy = np.max(np.abs(u))
figura, ejes = plt.subplots()
```

```
plt.xlim([x0,xn])
plt.ylim([-maximoy+0.5,maximoy+1])

# lineas de función y polinomio en gráfico
linea_poli, = ejes.plot(xi,u[:,0], '-')
plt.axhline(0, color='k')  # Eje en x=0
plt.title('EDP hiperbólica')
# plt.legend()
# txt_x = (x0+xn)/2
txt_y = maximoy*(1-0.1)
texto =
ejes.text(x0,txt_y,'tiempo:',horizontalalignment='left')
plt.xlabel('x')
plt.ylabel('y')
plt.grid()

# Nueva Trama
def unatrama(i,xi,u):
    # actualiza cada linea
    y = u[:,i]
    xi2=np.linspace(np.min(xi),np.max(xi),101)
    polinomio=mat.traza3natural(xi,y)
    x=sym.Symbol("x")
    puntosy=[]
    puntosx=[]
    for a in range(len(polinomio)):
        poli=polinomio[a]
        pn=sym.lambdify(x,poli)
        xi3=np.linspace(xi[a],xi[a+1],51)
        puntosy+=(list(pn(xi3)))
        puntosx+=(list(xi3))

    linea_poli.set_ydata(np.asarray(puntosy))
    linea_poli.set_xdata(np.asarray(puntosx))
```

```
    linea_poli.set_label('tiempo linea: '+str(i))
    texto.set_text('tiempo['+str(i)+']')
    # color de la línea
    if (i<=9):
        lineacolor = 'C'+str(i)
    else:
        numcolor = i%10
        lineacolor = 'C'+str(numcolor)
    linea_poli.set_color(lineacolor)
    return linea_poli, texto

# Limpia Trama anterior
def limpiatrama():
    linea_poli.set_ydata(np.ma.array(xi, mask=True))
    linea_poli.set_label('')
    texto.set_text('')
    return linea_poli, texto

# Trama contador


i = np.arange(0,tramas,1)
ani = animation.FuncAnimation(figura,
                              unatrama,
                              i ,
                              fargs=(xi,u),
                              init_func=limpiatrama,
                              interval=retardo,
                              blit=True)
# Graba Archivo video y GIFAnimado :
# ani.save('EDP_hiperbólica.mp4')
ani.save('EDP_PresionTuboMusicalAbierto.gif',
writer='imagemagick')
plt.title("****Gráfica Trazador Cúbico****")
```

```
plt.draw()
plt.show()
```