TP 05

Tracé de fonctions

Sources :

Proposition de corrigé

Activité 1 : Tracé de fonctions simples

En préambule :

```
import matplotlib.pyplot as plt
from numpy import exp, linspace, pi, sin, cos
from math import floor
```

Q 1 : x est une liste python, donc de type `list`.

Q 2 : Python représente une fonction comme une ligne brisée. On indique les coordonnées des extrémités des segments en passant en argument la liste des abscisses et celle des ordonnées à la fonction plot.

Q 3 : Le tracé s'arrete au point d'abscisse 9,5. C'est bien le dernier élément de x.

Q 4 :

```
def ex_sin(nom_de_fichier):
    """Trace la courbe du sinus sur [0,10] et l'enregistre dans nom_de_fichier"""
    x = linspace(0,10,200)
    y = [sin(t) for t in x]

    plt.clf()
    plt.plot(x,y,label='sin(x)')
    plt.xlabel('x')
    plt.legend(loc=0)
    plt.title('Tracé du sinus sur [0,10]')
    plt.savefig(nom_de_fichier)
```

Q 5 :

```
def transitoire(A,nom_de_fichier):
    """Trace les graphes de t->A(1-exp(t/tau)) pour tau = 2,4,6,8 sur [0,10].
       Les sauvegarde dans nom_de_fichier."""
    x = linspace(0,10,200)
    tau = [0.5,1,2,4,8]
    style = ['g-','b-','r-','b--','r--']
    plt.clf()
    for k in range(5):
       y = [A*(1-exp(-t/tau[k])) for t in x]
       plt.plot(x,y,style[k],label='$\\tau='+str(tau[k])+'$')
    plt.xlabel('$t$')
    plt.ylabel('$'+str(A)+'\\times\\exp(-t / \\tau)$')
```

```
    plt.title('Régime transitoire, A={}'.format(A))
    plt.axis([0,10,0,A])
    plt.legend(loc=0)
    plt.savefig(nom_de_fichier)
    return None
```

Activité 2 : Synthèse de Fourier

Q 6 :

```
def creneau(t):
    """0 si t entier
       1 si floor(t) pair, -1 sinon"""
    x = t% 2
    if 0 < x < 1 :
        return 1
    elif 1 < x < 2 :
        return -1
    else:
        return 0
```

Q 7 :

```
def sp_creneau(n,t):
    """Renvoie la somme partielle C_n de la série de Fourier associée au créneau"""
    S = 0
    for p in range(n+1) :
        S = S + sin((2*p+1)*pi*t)/(2*p+1)
    return S*4/pi
```

Q 8 :

```
def fourier_creneau(nom_de_fichier):
    """Trace la fonction creneau et quelques approximations de Fourier"""
    x = linspace(0,4,4000)
    C = [creneau(t) for t in x]
    plt.clf()
    plt.plot(x,C,'b-',label="$C$")
    n = [0,3,5,100]
    style = ['-r','-c','-g','-m']
    for k in range(4):
        Cn = [sp_creneau(n[k],t) for t in x]
        plt.plot(x,Cn,style[k],label="$C_{"+str(n[k])+"}$")
    plt.xlabel("$t$")
    plt.ylabel("$y$")
    plt.title("Approximation de Fourier du signal créneau")
    plt.legend(loc=0)
    plt.savefig(nom_de_fichier)
    return None
```

Q 9 :

```
def triangle(t):
    """Renvoie le signal triangle"""
    x = t % 2
    if 0 <= x <= 1 :
        return 1-2*x
    else :
        return 2*x - 3
```

Q 10 :

```
def sp_triangle(n,t):
    """Renvoie la somme partielle C_n de la série de Fourier associée au créneau"""
    S = 0
    for p in range(n+1) :
        S = S + cos((2*p+1)*pi*t)/(2*p+1)**2
    return S*8/pi**2
```

Q 11 :

```
def fourier_triangle(nom_de_fichier):
    """Trace la fonction creneau et quelques approximations de Fourier"""
    x = linspace(0,4,1000)
    T = [triangle(t) for t in x]
    plt.clf()
    plt.plot(x,T,'b-',label="$T$")
    n = [0,3,5,100]
    style = ['-r','-c','-g','-m']
    for k in range(4):
        Tn = [sp_triangle(n[k],t) for t in x]
        plt.plot(x,Tn,style[k],label="$T_{"+str(n[k])+"}$")
    plt.xlabel("$t$")
    plt.ylabel("$y$")
    plt.title("Approximation de Fourier du signal triangulaire")
    plt.legend(loc=0)
    plt.savefig(nom_de_fichier)
    return None
```