Ciências
ULisboa

Modelação Numérica 2022, Exame 2 (7 de Junho)

1. O comando **`X=np.fft.fft(x)`** calcula a transformada discreta de Fourier Admita que a série **`x`** é uma série temporal com $N = 1000$ elementos, com período de amostragem $\Delta t = 0.2\ s$.
   (a) Caracterize **`X`**.
   `X` é um array complex com 1000 elementos
   (b) Explique o conceito de frequência de Nyquist e calcule o seu valor no caso vertente.
   A frequência de Nyquist é a frequência máxima do espectro. De acordo com o teorema da amostragem é igual a $1/(2\Delta t)$
   (c) Explique por que razão só costumamos olhar para metade dos valores de **`X`**.
   O espectro complexo é representado por duas séries reais: amplitude e fase. No caso de uma série real o espectro de amplitude é par (simétrico em relação à origem) e só representamos a metade positiva
   (d) Escreva um fragmento python que represente graficamente **`x`** e **`X`**, utilizando um número suficiente de sub-gráficos.

```
…
N=len(x)
time=np.arange(0,N*dt,dt)
fNyq=1/(2*dt)
N2=N//2+1
df=fNyq
freq=np.arange(0,fNyq+df/2,df)
fig,ax=plt.suplots(nrows=3)
ax[0].plot(time,x)
ax[1].plot(freq,np.abs(X[0:N2])
ax[2].plot(freq,np.arctan2(imag(X[0:N2]),real(X[0:N2))
…
```

2. A dispersão de uma nuvem poluente num fluido com velocidade constante pode representar-se pela equação de advecção-difusão (a uma dimensão):

$$\frac{\partial C}{\partial t} = -u\frac{\partial C}{\partial x} + K\left(\frac{\partial^2 C}{\partial x^2}\right)$$

   (a) Discretize a equação num método explícito utilizando aproximações de segunda ordem no espaço e primeira ordem no tempo.
   (b) Repita (a) com um método implícito.
   (c) Escreva um código python que, para o caso (a): (1) inicialize as variáveis necessárias; (2) calcule a evolução temporal de C, admitindo que a sua fronteira é constante.

```
…
Nx=len(C)
CP=np.copy(C)
for it in range(nt):
```

```
    for i in range(1,Nx-1):
       CP[i]=C[i]-u*dt/dx*(C[i+1]-C[i-1])\
                      +k*dt/dx**2*(C[i-1]-2*C[i]+C[i+1])
    C=np.copy(CP)
```
…

Acrescente ao código anterior uma condição fronteira cíclica.

…

```
    Nx=len(C)
    CP=np.copy(C)
    for it in range(nt):
    for i in range(Nx):
        ip1=i+1
        im1=i-1
        if ip1>Nx-1
            ip1=0
        if im1<0
            im1=Nx-1
       CP[i]=C[i]-u*dt/dx*(C[ip1]-C[im1])\
                    +k*dt/dx**2*(C[im1]-2*C[i]+C[ip1])
    C=np.copy(CP)
```

3. Explique sucintamente o método de otimização baseado no conceito de função de custo. Exemplifique escrevendo uma função de custo (em python) que poderia ser utilizada para localizar uma fonte sonora a partir de observações de tempos de chegada em N estações, num meio com velocidade do som constante.

Este método utiliza um algoritmo para localizar iterativamente a combinação de parâmetros que dá origem a um mínimo da função de custo. O objetivo é encontrar o mínimo absoluto, mas o método pode terminar num mínimo local.

Exemplo: dadas as posições das nE estações (xE,yE,zE) e os tempos de chegada dos sinais (tE). Pretende-se conhecer a posição da fonte (xS,yS,zS).

```
import numpy as np
def cost(xS,yS,zS,xE,yE,zE,tE):
#xE,yE,zE,tE podem ser definidos no programa main
    c=340 #velocidade do som
    custo=np.mean((np.sqrt((xE-xS)**2+(yE-yS)**2+(zE-zS)**2)/c-tE)**2)
    return custo
```