Devoir Surveillé 03
Le 02 février 2018– 2h00

Informatique

PSI★

1 Bases de données

Correction **Question 1**

$$\pi_{\text{num,sta\_commune}}(\text{RADOME})$$

```
SELECT num_sta, commune FROM RADOME
```

Question 2

$$\pi_{\text{num,sta\_commune}}\left(\sigma_{\text{departement}=83}(\text{RADOME})\right)$$

```
SELECT num_sta, commune FROM RADOME WHERE département = 83
```

Question 3

$$\pi_{\text{commune}}\left(\sigma_{\text{RADOME.num\_sta=SYNOP.num\_sta}}(\text{RADOME}\times\sigma_{\text{ff}>20}(\text{SYNOP}))\right)$$

```
SELECT RADOME.commune
    FROM RADOME JOIN SYNOP ON RADOME.num_sta=SYNOP.num_sta
    WHERE SYNOP.ff>20
```

Question 4

$$\pi_{\gamma\text{MAX(pmer)}}(\text{SYNOP})$$

```
SELECT MAX(pmer) FROM SYNOP
```

2 Mise en situation

2.1 Équation gouvernant la température

Correction **Question 5**

$$\rho c_p \frac{\partial T(x,t)}{\partial t} = \lambda \frac{\partial^2 T(x,t)}{\partial x^2}$$

2.2 Conditions aux limites

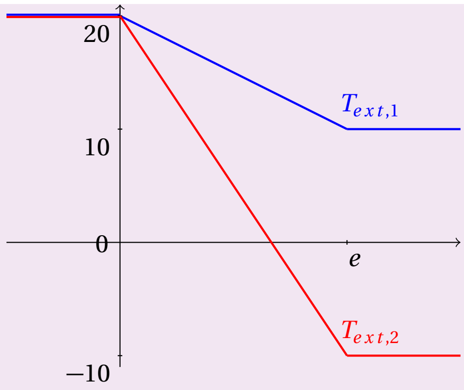
Correction **Question 6**

- **cas 1 :** $T(0,t)=T_{\text{int}}$ et $T(e,t)=T_{\text{ext}}$;
- **cas 2 :** $T(0,t)=T_{\text{int}}$ et $\dfrac{\partial T}{\partial t}(e,t)=0$.

Question 7 En régime permanent, l'équation différentielle devient : $\dfrac{\partial^2 T(x,t)}{\partial x^2}=0$. On a donc : $T(x)=k_1 x+k_2$. Par suite : $T(0)=T_{\text{int}}=k_2$ et $T(e)=T_{\text{ext}}=k_1 e+k_2$. On a donc : $k_1=\dfrac{T_{\text{ext}}-T_{\text{int}}}{e}$. Au final : $T(x)=\dfrac{T_{\text{ext}}-T_{\text{int}}}{e}x+T_{\text{int}}$.

- **conditions 1 :** lorsque $t_1<0$, $T_{\text{int}}=20°\text{C}$ et $T_{\text{ext},1}=10°\text{C}$. En conséquence, $T(x)=\dfrac{T_{\text{ext},1}-T_{\text{int}}}{e}x+T_{\text{int}}$.
- **conditions 2 :** lorsque $t_2>0$, $T_{\text{int}}=20°\text{C}$ et $T_{\text{ext},2}=-10°\text{C}$. En conséquence, $T(x)=\dfrac{T_{\text{ext},2}-T_{\text{int}}}{e}x+T_{\text{int}}$.

Question 8



3 Résolution numérique : méthode des différences finies

Correction **Question 9** On a $\alpha = \dfrac{\rho c_p}{\lambda}$.

Question 10 On a, pour t<0, $T(0,0) = b = T_{\text{int}}$ et $T(e,0) = ae + T_{\text{int}} = T_{\text{ext},1}$. On a donc : $a = \dfrac{T_{\text{ext},1} - T_{\text{int}}}{e}$. Au final :

$$T(x,0) = \frac{T_{\text{ext},1} - T_{\text{int}}}{e} x + T_{\text{int}} \quad \text{pour } x \in [0,e].$$

3.1 Discrétisation dans l'espace et dans le temps

Correction **Question 11** On a : $\Delta x = \dfrac{e}{N+1}$.

Question 12 On a : $x_i = i\Delta x$.

3.2 Méthode utilisant un schéma explicite

Correction **Question 13** On additionne les deux lignes, on a directement :

$$T(x+\Delta x,t) + T(x-\Delta x,t) = 2T(x,t) + \frac{\partial^2 T(x,t)}{\partial x^2}(\Delta x)^2 + o((\Delta x)^3)$$

puis on isole :

$$\frac{\partial^2 T(x,t)}{\partial x^2} = \frac{T(x+\Delta x,t) - 2T(x,t) + T(x-\Delta x,t)}{\Delta x^2} + o(\Delta x)$$

Question 14

$$\left[\frac{\partial^2 T(x,t)}{\partial x^2}\right]_{x_i,t_k} = \frac{T_{i+1}^k - 2T_i^k + T_{i-1}^k}{\Delta x^2}$$

Question 15 En injectant les approximations dans l'EDP, nous obtenons $T_i^{k+1} - T_i^k = \dfrac{\Delta t}{\alpha \Delta x^2}\left(T_{i+1}^k - 2T_i^k + T_{i-1}^k\right)$. Ainsi, $r = \dfrac{\Delta t}{\alpha \Delta x^2}$

Question 16 Pas pour $i = 0$ et $i = N+1$. Pour ces valeurs nous avons : $T_0^k = 20^o C$ et $T_{N+1}^k = -10^o C$.

Question 17 On a, pour $i \in]0;N+1[$:

i	$T_i^{k+1} = r\,T_{i-1}^k + (1-2r)\,T_i^k + r\,T_{i+1}^k$
$i=1$	$T_1^{k+1} = r\,T_0^k + (1-2r)\,T_1^k + r\,T_2^k$
$i=2$	$T_2^{k+1} = r\,T_1^k + (1-2r)\,T_2^k + r\,T_3^k$
$i=3$	$T_3^{k+1} = r\,T_2^k + (1-2r)\,T_3^k + r\,T_4^k$
$i=N-1$	$T_{N-1}^{k+1} = r\,T_{N-2}^k + (1-2r)\,T_{N-1}^k + r\,T_N^k$
$i=N$	$T_N^{k+1} = r\,T_{N-1}^k + (1-2r)\,T_N^k + r\,T_{N+1}^k$

On a donc :

$$T^{k+1} = M \cdot T^k + r\,V$$

avec :

$$M = \begin{pmatrix} 1-2r & r & 0 & 0 & 0 & \dots & 0 \\ r & 1-2r & r & 0 & 0 & \dots & 0 \\ 0 & r & 1-2r & r & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \dots & \dots \\ 0 & 0 & 0 & 0 & 0 & r & 1-2r \end{pmatrix} \quad \text{et} \quad V = \begin{pmatrix} T_0^k = T_{\text{int}} \\ 0 \\ \vdots \\ 0 \\ T_N^k = T_{\text{ext},2} \end{pmatrix}.$$

Question 18 À l'instant $t=0$, le flux de température dans le mur est connu. La température extérieure passe alors à $-10^o C$. On utilise alors l'équation de récurrence.

(Lors du codage de la méthode explicite, il faut faire attention à la divergence du schéma. Pour cela, il faut prendre des intervalles de temps « petits ».)

Question 19

```
def euler_explicite(M,T_0,r,V,k):
    if k==0:
        return T_0
    else:
        T=euler_explicite(M,T_0,r,V,k-1)
        return numpy.dot(M,T)+r*V
```

3.3 Méthode utilisant un schéma implicite

Correction **Question 20** On peut utiliser l'algorithme du pivot de Gauss. Sa complexité est en $\mathcal{O}(n^3)$.

Question 21 Proposition de corrigé qui ne correspond pas exactement aux préconisations de l'énoncé :

```
def CalcTkp1(M,D):
    N=D.shape[0]
    Cp=[M[0,1]/M[0,0]]
    Dp=[D[0,0]/M[0,0]]
    U=numpy.zeros((N,1))
    for i in range(1,N-1):
        Cp.append(M[i,i+1]/(M[i,i]-M[i,i-1]*Cp[i-1]))
        Dp.append((D[i,0]-M[i,i-1]*Dp[-1])/(M[i,i]-M[i,i-1]*Cp[i-1]))
    Dp.append((d[N-1,0]-M[N-1,N-2]*Dp[-1])/(M[N-1,N-1]-M[N-1,N-2]*Cp[-1]))
    U[N-1,0]=Dp[-1]
    for j in range(2,N):
        U[N-j,0]=Dp[N-j]-Cp[N-j]*U[N-j+1,0]
    return U
```

Question 22 Cet algorithme contient deux boucles `for` indépendantes. Dans chacune de ces boucles, le nombre d'opérations est une constante. En dehors de ces boucles, il n'y a aussi que des opérations en temps constant. Ainsi la complexité de l'algorithme est linéaire (à comparer à une complexité cubique pour l'algorithme de Gauss).

4 Résolution de l'équation différentielle implicite

Correction **Question 23**

```
from math import sqrt

def calc_norme(v):
    res = 0
    for i in range(len(v)):
        res = res+v[i]**2
    return sqrt(res)
```

Question 24

```
def solution(M,T,V) :
    """
    Entrées :
        * M, numpy.array (N+1 x N+1) : matrice àinverser
        * T, numpy.array (N+1 x 1) : température àchaque abscisse du mur àl'instant k
        * V, numpy.array (N+1 x 1) : température *imposée* àchaque abscisse du mur àl'instant final
    Sortie :
        * TT, numpy.array (N+1 x 1) : température àchaque abscisse du mur àl'instant k+1
    """
    # On considère que r a été définie auparavant comme variable globale.
    D = T+r*V
    return CalcTkp1(M,D)
```

Question 25

```
ItMax = 2000
T_tous_k = numpy.zeros([N,ItMax])
```

Question 26 On a : $T(x_i, k=0) = \dfrac{T_{\text{ext},1} - T_{\text{int}}}{e} x_i + T_{\text{int}} = \dfrac{T_{\text{ext},1} - T_{\text{int}}}{e} \cdot i \cdot \dfrac{e}{N+1} + T_{\text{int}} = \dfrac{T_{\text{ext},1} - T_{\text{int}}}{N+1} \cdot i + T_{\text{int}}$.

```
def T0(Ti,Tf,N):
    tt = numpy.zeros((N,1))
    for i in range(N):
        tt[i] = ((Tf-Ti)/(N-1))*i+Ti #si il y a N points on a divisé l'intervalle en N-1
    return tt
```

Question 27

```
t0 = T0(Tint,Text1,N)
for i in range(N):
    T_tous_k[i][0] = t0[i]
```

Question 28

```
M = numpy.zeros((N,N))
M[0][0]=1+2*r
M[0][1]=-r

M[N-1][N-1]=1+2*r
M[N-1][N-2]=-r
for i in range(1,N-1):
    M[i][i]=1+2*r
    M[i][i+1]=-r
    M[i][i-1]=-r

V = numpy.zeros((N,1))
V[0][0]=Tint
V[N-1][0]=Text2
```

Question 29

```
T1=solution(M,T0,V)
```

Question 30 Bloc d'instructions à intégrer dans la fonction `euler_implicite(M,T0,V)` :

```
v=numpy.zeros((T_tous_k.shape[0],1))
for i in range(T_tous_k.shape[0]):
    v[i,0]=T_tous_k[i,1]-T_tous_k[i,0]
k=1
```

```
T=T0(Tint,Text,N)
while calc_norme(v)>10**(-2) and k!=ItMax:
    k=k+1
    T=solution(M,T,V)
    for i in range(T_tous_k.shape[0]):
        T_tous_k[i,k]=T[i,0]
        v[i,0]=T_tous_k[i,k]-T_tous_k[i,k-1]
```

5 Analyse des résultats

Correction Question 31

```
k=[0,240,480,720,960,1200,1440]
les_X=[i*0.4/(N+1) for i in range(N+2)]
for i in k:
    plt.plot(les_X,T_tous_k[:,i])
plt.show()
```

Question 32 Le régime permanent est atteint au bout de 12 heures. Avec un pas temporel de 30 secondes, il y a donc 12×120 soit 2400 itérations. Si on considère que $Itmax = 2400$, on aura donc 100×2400 valeurs. Pour l'estimation de la taille du fichier on fait ici l'abstraction du codage des espaces et des retours à la ligne. La taille du fichier est donc : $100 \times 2400 \times 10 \times 1 \simeq 2,4$ Mo.

DS 03