Concours Blanc 2016

Transfert thermique dans un mur en régime transitoire – *D'après Concours CCP – PC 2015*

Informatique

Corrigé

1 Mise en situation

1.1 Équation gouvernant la température

Question 1 *En utilisant les hypothèses dimensionnelles, donner l'équation de la chaleur simplifiée.*

Correction

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

1.2 Conditions aux limites

Question 2 *Traduire chacune de ces conditions aux limites sur la fonction $T(x,t)$ et/ou sa dérivée.*

Correction

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

Dans la suite, seul le premier cas sera étudié.

Question 3 *Résoudre l'équation de la chaleur simplifiée* ***en régime permanent*** *dans les conditions suivantes :*

- ***conditions 1 :*** *pour un instant particulier négatif $t_1 < 0$;*
- ***conditions 2 :*** *pour un instant particulier positif $t_2 > 0$, très longtemps après la variation de température extérieure quand le régime permanent est de nouveau établi dans le mur.*

Correction

En régime permanent, l'équation différentielle devient : $\frac{\partial^2 T(x,t)}{\partial x^2} = 0$. 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 : $k_1 = \frac{T_{\text{ext}} - T_{\text{int}}}{e}$. Au final : $T(x) = \frac{T_{\text{ext}} - T_{\text{int}}}{e} x + T_{\text{int}}$.

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

Question 4 *Quelle est la nature des profils $T(x)$ obtenus (en régime permanent) à ces deux instants ? Tracer à la main les deux profils sur un même graphique sur la copie.*

2 Résolution numérique : méthode des différences finies

Question 5 *Quelle est l'expression de α en fonction des paramètres physiques du mur ?*

Correction On a $\alpha = \frac{\rho c_p}{\lambda}$.

Question 6 *Exprimer a et b en fonction de T_{int},$T_{ext,1}$ et e.*

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

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

2.1 Discrétisation dans l'espace et dans le temps

Question 7 *Donner l'expression de Δx en fonction de N et de l'épaisseur du mur e.*

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

Question 8 *Donner l'abscisse x_i du i^e point en fonction de i et Δx, sachant que $x_0 = 0$ et $x_{N+1} = e$.*

Correction On a : $x_i = i\Delta x$.

2.2 Méthode utilisant un schéma explicite

Question 9 *En déduire une expression approchée à l'ordre 1 de $\left[\dfrac{\partial^2 T(x,t)}{\partial x^2}\right]_{x,t}$ (dérivée partielle spatiale seconde de T évaluée au point x à l'instant t) en fonction de $T(x+\Delta x, t)$, $T(x-\Delta x, t)$ et $T(x,t)$ et Δx.*

Correction 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 :

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

On note T_i^k la température $T(x_i, t_k)$, évaluée au point d'abscisse x_i à l'instant t_k. De même, on note $T_{i+1}^k = T(x_i+\Delta x, t_k)$ et $T_{i-1}^k = T(x_i-\Delta x, t_k)$.

Question 10 *Déduire de la question précédente que $\left[\dfrac{\partial^2 T(x,t)}{\partial x^2}\right]_{x_i,t_k} = \dfrac{T_{i+1}^k - 2T_i^k + T_{i-1}^k}{\Delta x^2}$ (dérivée partielle seconde de T évaluée en x_i à l'instant t_k).*

Correction

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

Question 11 *En utilisant les questions précédentes, montrer que l'équation différentielle peut se mettre sous la forme :*

$$T_i^{k+1} = r\,T_{i-1}^k + (1-2r)\,T_i^k + r\,T_{i+1}^k.$$

r sera explicité en fonction de Δx, Δt et α.

Correction $r = \dfrac{\Delta t}{\alpha \Delta x^2}$

Question 12 *L'équation est-elle valable dans tout le domaine, c'est-à-dire pour toute valeur de i, $0 \le i \le N+1$? Que valent T_0^k et T_{N+1}^k ?*

Correction Oui...
On a : $T_0^k = 20^o C$ et $T_{N+1}^k = -10^o C$.

Question 13 *Montrer que pour tout instant k, le problème peut se mettre sous la forme matricielle suivante :*

$$T^{k+1} = M \cdot T^k + r V \quad \text{avec} \quad T^k = \begin{pmatrix} T_1^k \\ T_2^k \\ \dots \\ T_{N-1}^k \\ T_N^k \end{pmatrix}.$$

avec M une matrice carrée $N \times N$, V un vecteur de taille N que l'on explicitera.

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

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

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 14 *Expliciter succinctement comment déterminer la température dans le mur à chaque instant.*

Correction 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 15 *On donne* `T0` *le vecteur température à l'instant $k=0$. Écrire la fonction* `euler_explicite(M,T0,V,k)` *retournant le vecteur de température T^k à l'instant k. Cette fonction sera définie de manière* ***récursive****.*

Correction

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

2.3 Méthode utilisant un schéma implicite

Question 16 *Pour obtenir T^{k+1} en fonction de T^k, il est nécessaire d'inverser le système matriciel à chaque pas de temps. Donner le nom d'un algorithme permettant de faire cela et donner sa complexité.*

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

Question 17 *En utilisant l'algorithme de Thomas, écrire une fonction* `CalcTkp1(M,D)` *qui retourne le vecteur U, solution du système matriciel, à partir de la matrice M et du vecteur D.*

On pourra commencer par créer des vecteurs contenant des zéros : A, B, C, Cp ,Dp (pour C' et D') et U. Puis on pourra définir A, B et C à partir de la matrice M et ensuite calculer Cp et Dp puis U.

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

```
def CalcTkp1(M,d):
    N=d.shape[0]
```

```
    c=[M[0,1]/M[0,0]]
    d_prime=[d[0,0]/M[0,0]]
    u=np.zeros((N,1))
    for i in range(1,N-1):
        c.append(M[i,i+1]/(M[i,i]-M[i,i-1]*c[i-1]))
        d_prime.append((d[i,0]-M[i,i-1]*d_prime[-1])/(M[i,i]-M[i,i-1]*c[i-1]))
    d_prime.append((d[N-1,0]-M[N-1,N-2]*d_prime[-1])/(M[N-1,N-1]-M[N-1,N-2]*c[-1]))
    u[N-1,0]=d_prime[-1]
    for j in range(2,N):
        u[N-j,0]=d_prime[N-j]-c[N-j]*u[N-j+1,0]
    return u
```

Question 18 *Donner la complexité de l'algorithme et comparer à celui de la question 16. On prendra en compte uniquement les tests, les affectations (même de tableaux) et les opérations élémentaires (+, -, *, /) qui compte chacune pour « un ».*

Correction La complexité de l'algorithme est linéaire (à comparer à une complexité cubique pour l'algorithme de Gauss).

3 Résolution de l'équation différentielle implicite

Question 19 *Écrire une fonction* `calc_norme` *qui calcule la norme d'un vecteur.*

Correction

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

Question 20 *Écrire la fonction* `solution(M,T,V)`*, d'arguments une matrice M tridiagonale et deux vecteurs T et V et retournant le vecteur U tels que $MU = T + rV$.* On utilisera la fonction `CalcTkp1`.

Correction

```
def solution(M,T,V) :
    """
    Entrées :
        * M, np.array (N+1 x N+1) : matrice àinverser
        * T, np.array (N+1 x 1) : température àchaque abscisse du mur àl'instant k
        * V, np.array (N+1 x 1) : température *imposée* àchaque abscisse du mur àl'instant final
    Sortie :
        * TT, np.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 21 *Affecter la valeur 2000 à* `ItMax`*. Créer la matrice* `T_tous_k` *de dimensions $N \times ItMax$ en la remplissant de zéros.*

Correction

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

Question 22 *D'après les questions précédentes la température T dans le mur vérifie la condition initiale $T(x,0) = ax + b$ où $a = \dfrac{T_{ext,1} - T_{int}}{e}$ et $b = T_{int}$.*
Écrire la fonction `T0(Tint,Text,N)` *d'arguments la température intérieure T_{int}, la température extérieure T_{ext}*

et l'entier N et retournant le vecteur T^0, de taille N, contenant les températures en chaque point du mur lorsque $t=0$.

Correction 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 = np.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 23 *Écrire la suite d'instructions qui affecte à la première colonne de* `T_tous_k` *le vecteur des valeurs initiales T^0.* On utilisera la fonction `T0`.

Correction

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

Question 24 *Écrire les instructions permettant de définir M et le vecteur V qui interviennent dans l'équation* **??**.

Correction

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

Question 25 *Donner la suite d'instructions permettant de calculer le profil de la température à l'instant $k=1$ ($t=\Delta t$). C'est à dire le vecteur T^1* . On utilisera la fonction `solution`.

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

Question 26 *Écrire la fonction* `euler_implicite(M,T0,V)` *d'argument une matrice M et deux vecteurs $T0$ et V et renvoyant la matrice* `T_tous_k`.
Cette boucle sera interrompue lorsque la norme du vecteur T^k-T^{k-1} deviendra inférieure à 10^{-2} ou lorsque le nombre d'itérations atteindra la valeur ItMax (prévoir deux cas). Utiliser pour cela les fonctions `calc_norme` *et* `solution` *définies précédemment.*

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

```
v=np.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]
```

4 Analyse des résultats

Question 27 *Le pas de discrétisation temporel est de 30 secondes. Les résultats de la simulation sont stockés dans la matrice* `T_tous_k` *définie précédemment. Écrire les instructions permettant de tracer le réseau de courbes précédentes.*

Correction

```
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 28 *Quelle sera la taille du fichier texte généré ?*

Correction Si on considère que $Itmax = 2000$, on aura donc 100×2000 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 : $100 \times 2000 \times 10 \times 1 \simeq 2\,\text{Mo}$.