Résolution numérique de l'équation de la diffusion

I Méthode des différences finies avec pas de temps explicite

Sous des hypothèses étudiées en cours, l'évolution de la température $T(M,t)$ dans un matériau est régie par l'équation aux dérivées partielles

$$\Delta T(M,t) - \frac{1}{D}\frac{\partial T(M,t)}{\partial t} = -\frac{p(M,t)}{\lambda} \quad \text{ou encore} \quad \frac{\partial T}{\partial t}(M,t) = D\,\Delta T(M,t) + \frac{D}{\lambda}\,p(M,t) \tag{1}$$

où λ désigne la conductivité thermique, D le coefficient de diffusion thermique et où le terme source $p(M,t)$ représente une puissance volumique. Cette edp est en général assortie de conditions de bord et de conditions initiales qui assurent l'unicité de sa solution. Malheureusement, exprimer cette solution au moyen de fonctions mathématiques usuelles s'avère souvent difficile, voire hors de notre portée. On recourt donc à une *résolution numérique* fournissant des *valeurs approchées* du champ de température. Conformément au programme, nous nous limitons à des situations unidimensionnelles pour lesquelles $T(M,t) = T(x,t)$ (figure 1). Il s'agit typiquement de déterminer la température le long d'une barre de longueur L ou dans un mur.

Au lieu de déterminer le champ T pour tout $x \in [0,L]$, on le calcule seulement en N points équidistants d'abscisses x_i, séparés de $\delta x = L/(N-1)$. À un instant donné t, le champ de température est ainsi représenté par un ensemble de N valeurs $\{T_i(t) = T(x_i,t)\}$ avec $x_i = i\delta x$ et $i \in \{0,1,...,N-1\}$.

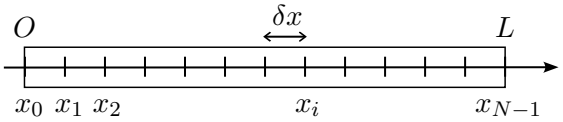


FIGURE 1 – Discrétisation spatiale pour un problème unidimensionnel

Par des développements de Taylor [1], on montre que la dérivée seconde par rapport à x s'obtient approximativement par

$$\frac{\partial^2 T(x_i,t)}{\partial x^2} \simeq \frac{T_{i+1}(t) + T_{i-1}(t) - 2\,T_i(t)}{\delta x^2} \quad \text{pour} \quad i \in \{1, N-2\} \quad . \tag{2}$$

Cette relation revient à approximer une dérivée, mathématiquement définie comme une limite, par un quotient faisant intervenir des différences entre points voisins. Pour cette raison, la technique présentée ici se somme *méthode des différences finies*.

Après cette discrétisation spatiale, on discrétise aussi le temps en ne calculant le champ de température qu'aux instants $t_0 = 0$, $t_1 = \delta t$, $t_2 = 2\delta t$, etc. Notons $t_k = k\delta t$ et $T_i^k = T_i(t_k)$. L'entier placé en indice se réfère à la position alors que celui placé en exposant se réfère au temps. Les valeurs $\{T_i^0, i \in [\![0, N-1]\!]\}$ décrivent la situation initiale et sont supposées connues. À partir de ces données, nous allons calculer successivement $\{T_i^1, i \in [\![0, N-1]\!]\}$, puis $\{T_i^2, i \in [\![0, N-1]\!]\}$, etc., c'est à dire le champ de température aux dates successives. On appelle *pas de temps* l'opération [2] qui permet le passage des $\{T_i^k\}$ aux $\{T_i^{k+1}\}$, c'est à dire l'obtention des températures à l'instant t_{k+1} à partir de celles à l'instant t_k.

Pour effectuer un pas de temps, on applique la méthode d'Euler concernant les équations différentielles de premier ordre, que vous avez étudiée en PCSI. Elle consiste à approximer la dérivée de la fonction à déterminer par un taux d'accroissement. Ici, cela consiste à écrire

$$\frac{\partial T}{\partial t}(x_i,t_k) \simeq \frac{T_i^{k+1} - T_i^k}{\delta t} \quad .$$

1. Le professeur conduit ce calcul au tableau
2. Il y a là un léger abus de langage puisque δt est lui aussi appelé « pas de temps ».

En tenant compte de la discrétisation spatiale et de la discrétisation temporelle, l'équation de la chaleur s'écrit sous la forme

$$\frac{T_i^{k+1} - T_i^k}{\delta t} \simeq D\frac{T_{i+1}(t) + T_{i-1}(t) - 2\,T_i(t)}{\delta x^2} + \frac{D}{\lambda}p_i^k$$

d'où

$$\boxed{T_i^{k+1} \simeq T_i^k + \frac{D\delta t}{\delta x^2}\left(T_{i+1}^k + T_{i-1}^k - 2T_i^k\right) + \frac{D\delta t}{\lambda}\,p_i^k} \quad . \tag{3}$$

Connaissant l'ensemble des températures à l'instant t_k, la relation (3) permet de les obtenir à l'instant t_{k+1} comme le montre le schéma de la figure (2). Il faut cependant noter que la relation d'itération (3) ne s'applique ni pour $i = 0$ (T_{i-1} n'est alors pas défini), ni pour $i = N - 1$ (T_{i+1} n'est alors pas défini). Les extrémités doivent donc faire l'objet d'un traitement particulier de manière à exprimer les conditions de bord, qui jouent comme dans une résolution exacte un rôle fondamental.

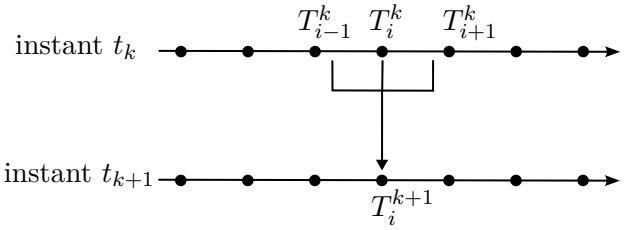


FIGURE 2 – Schéma de discrétisation de la méthode explicite

La méthode des différences finies exposée ici ne concerne pas seulement l'équation de la chaleur, mais plus généralement tout phénomène régi par l'équation de la diffusion (comme la diffusion de particules). Moyennant quelques adaptations, on peut même l'appliquer à une vaste classe d'edp. Néanmoins, tous les exemples traités dans la suite concerneront la thermique.

La méthode présentée ici est dite *explicite*, au sens où T_i^{k+1} s'exprime par la relation très simple (3). Il existe des méthodes « implicites », ne figurant pas au programme, dans lesquelles les $\{T_i^{k+1}\}$ s'obtiennent de manière moins directe. Nous ne les étudierons pas.

Choix des pas de temps et d'espace

Par nature, la méthode exposée ici ne fournit pas exactement la température, mais en donne des valeurs approchées, d'autant plus précises que δt et δx sont petits. En particulier, si δt est trop grand, les erreurs s'amplifient au fil des itérations et on finit par obtenir des résultats absurdes et des températures tendant vers l'infini. On dit dans ce cas que la méthode est *instable*. On évite cet écueil et on assure *la stabilité du schéma numérique*, si

$$\frac{D\delta t}{\delta x^2} < \frac{1}{2} \quad \text{ou} \quad \delta t < \frac{\delta x^2}{2D} \quad \text{(condition CFL pour Courant, Friedrichs et Lewy).}$$

Cette condition résulte d'une subtile étude mathématique mais il est possible de l'interpréter avec un regard de physicien. D'après la loi d'échelle des phénomènes diffusifs, $\delta x^2/D$ représente le temps de diffusion sur la longueur δx. Le pas de temps δt doit être inférieur à la moitié du temps de diffusion sur la longueur δx. En pratique, on choisira δt légèrement inférieur à $\delta x^2/(2D)$ (ou δx légèrement supérieur à $\sqrt{2D\delta t}$).

Dans un problème physique donné, le choix de δt et δx est aussi lié aux échelles d'espace ℓ et de temps τ que l'on souhaite analyser : il convient évidemment de choisir $\delta x < \ell$ et $\delta t < \tau$ pour que les variations intéressantes ne passent pas « entre les mailles du filet ». La longueur ℓ et la durée τ étant liées par la loi d'échelle $\ell^2 \simeq D\tau$, δx et δt peuvent être pris assez petits tout en respectant la condition CFL. Considérons par exemple une barre de cuivre ($D = 1.10^{-4}\,\mathrm{m^2.s^{-1}}$) dans laquelle on étudie des variations de température à l'échelle du centimètre. Pour $\ell = 1\,\mathrm{cm}$, on a $\tau = 1\,\mathrm{s}$. On peut choisir $\delta x = 0,5\,\mathrm{cm}$ et $\delta t = 0,4\,\delta x^2/D = 0,1\,\mathrm{s}$.

II Codage sous Python

À un instant t_k donné, les températures $\{T_i^k, i \in [\![0, N-1]\!]\}$ sont stockées dans un tableau de numpy de N flottants. De même la puissance volumique p, supposée pour l'instant indépendante du temps, est représentée par le tableau `p_source` contenant les valeurs $p(x_i)$. Ce tableau, tout comme les paramètres physiques du problème (L, D, λ, ...) et ceux concernant la discrétisation (N, δt, δx, ...) sont traités comme des variables globales, sauf mention contraire. Certaines fonctions seront fournies par le professeur et d'autres seront à compléter dans le fichier destiné aux élèves. On pose $r = D\delta t/\delta x^2$ et $q = D\delta t/\lambda$.

1. Coder la fonction `pdt_naif(T_anc)` qui, prenant en argument le tableau `T_anc` représentant la température à l'instant t_k, renvoie le tableau représentant la température à l'instant t_{k+1}. Les relations (3) seront appliquées dans boucle `for` portant sur i. Les valeurs T_0^k et T_{N-1}^k sont pour l'instant choisies arbitrairement (nulles si cela vous chante) ; les conditions de bord seront traitées plus tard.

2. On accélère considérablement le calcul en remplaçant la boucle `for` par une opération directe sur des tableaux de numpy. Voici la fonction correspondante que je vous demande de comprendre et de savoir réécrire.

```
def pdt_vecto(T_anc) :
    T_nouv = np.zeros(N)
    T_nouv[1:-1] = T_anc[1:-1]  \
                 + prefac * (T_anc[0:-2] + T_anc[2:] - 2 * T_anc[1:-1]) \
                 + q * p_source[1:-1]                                          5
    return T_nouv
```

3. La linéarité des relations (3) permet d'en donner un écriture matricielle.

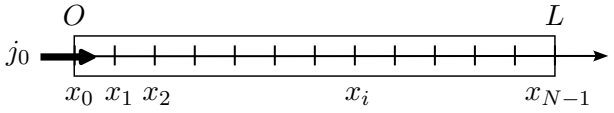
$$\begin{pmatrix} T_0^{k+1} \\ T_1^{k+1} \\ T_2^{k+1} \\ \dots \\ \dots \\ T_{N-2}^{k+1} \\ T_{N-1}^{k+1} \end{pmatrix} = \begin{pmatrix} 1 & & & & & & & \\ r & 1-2r & r & & & & & \\ & r & 1-2r & r & & & & \\ & & & \dots & \dots & \dots & & \\ & & & & \dots & \dots & \dots & \\ & & & & & r & 1-2r & r \\ & & & & & & & 1 \end{pmatrix} \begin{pmatrix} T_0^k \\ T_1^k \\ T_2^k \\ \dots \\ \dots \\ T_{N-2}^k \\ T_{n-1}^k \end{pmatrix}$$

En toute rigueur, les relations (3) ne s'appliquent ni pour $i = 0$, ni pour $i = N-1$, de sorte que la première et la dernière ligne auraient dû être laissées vierges. Ici on a posé arbitrairement $T_0^{k+1} = T_0^k$ et $T_{N-1}^{k+1} = T_N^k$, comme si la température était constante aux deux extrémités. L'expression correcte des conditions de bord viendra plus loin.

Coder la fonction `construit_matrice(N, prefac)` (`prefac` désigne le quotient r) qui renvoie la matrice M ci-dessus, puis la fonction `pdt_matrice(T_anc)` jouant le même rôle que `pdt_naif`, mais en utilisant un produit matriciel sous la forme `np.dot(M, T_anc)`.

4. Passons au traitement des conditions de bord. Dans un premier temps, on envisage la situation la plus simple dans laquelle la température aux bords reste fixée. Coder la procédure `bords_fixes(T_anc, T_nouv)` qui ne renvoie rien, mais qui modifie le tableau `T_nouv` de manière à exprimer cette fixité.

5. Considérons maintenant une condition de bord dans laquelle la température est fixée à l'extrémité droite ($i = N-1$). À l'extrémité gauche ($i = 0$), un flux thermique constant, pénétrant la matière avec une densité surfacique j_0, est imposé.



D'après la loi de Fourier, on a

$$\forall t \geqslant 0,\ -\lambda \frac{\partial T}{\partial x}(0,t) = j_0 \quad .$$

Dans le cadre des différences finies,

$$\frac{\partial T}{\partial x}(0,t_k) \simeq \frac{T_1^k - T_0^k}{\delta x} \quad \text{et la condition de bord ci-dessous s'écrit} \quad T_0^k = T_1^k + \frac{j_0 \delta x}{\lambda} \quad .$$

Coder la procédure `impose_j_gauche(T_anc, T_nouv)` qui ne renvoie rien, mais qui modifie le tableau `T_anc` de manière à exprimer ces conditions de bord.

6. Le champ de température initial est stocké dans un tableau `T_init` contenant les N valeurs T_i^0. On note N_{pas} le nombre de pas de temps à effectuer, c'est à dire le nombre d'itérations des relations (3). Les dates t_k, au nombre de $N_{\text{pas}}+1$, s'étendent de $t_0 = 0$ à $t_{N_{\text{pas}}} = N_{\text{pas}}\delta t$. Le champ de température évolue au fil des itérations et nous choisissons de stocker toutes les valeurs qu'il prend successivement. Pour cela, on utilise un tableau `champ_T` de taille $(N_{\text{pas}}+1) \times N$; `champ_T[k, i]` est la valeur flottante de T_i^k. De cette manière,

— `champ_T[k,:]` contient les N températures à l'instant t_k ; `champ_T[0,:]` représente en particulier la condition initiale ;

— `champ_T[:, i]` contient les $(N_{\text{pas}}+1)$ valeurs de la température au point d'abscisse x_i.

Coder la fonction `evolution(T_init, N_pas, fonction_pdt, fonction_bords)` qui calcule et renvoie le tableau `champ_T`. Ses deux derniers paramètres désignent eux-mêmes des fonctions :

— `fonction_pdt` est la fonction choisie pour traduire les relations (3) (à l'appel, l'utilisateur aura le choix entre `pdt_naif`, `pdt_vecto` et `pdt_matrice`) ;

— `fonction_bords` est la fonction choisie pour exprimer les conditions de bord (à l'appel, l'utilisateur devra la choisir selon le type de problème qu'il traite).

III Applications directes

Le fichier destiné aux élèves contient des fonctions permettant de générer diverses conditions initiales, divers termes sources et solutions exactes de l'équation de la chaleur. Il comporte aussi une fonction permettant de représenter commodément le champ de température, et même une fonction `studio_poinca` pour produire un dessin animé.

En complétant le code, traiter les différents problèmes proposés dans le fichier. Lorsque c'est possible, comparer le résultat à celui connu par une résolution exacte. En général, un régime permanent finit par s'établir et dans ce cas, l'expression stationnaire de $T(x)$ est facile à trouver, ce qui permet de vérifier que la méthode numérique ne donne pas n'importe quoi. Comparer aussi le temps d'exécution selon que l'on utilise `pdt_naif`, `pdt_vecto` ou `pdt_matrice`.

IV Applications plus élaborées

IV.1 Stabilité d'un réacteur chimique

La feuille de TD sur les transferts thermiques comporte un exercice intitulé « explosion thermique ». Il concerne un réacteur de longueur $L = 2\ell$ dont les extrémités sont maintenues à la température T_0 et dans lequel se dégage une puissance thermique dépendant de la température selon

$$p = p_0 \exp(\alpha(T - T_0)) \quad \text{avec} \quad p_0 > 0.$$

On pose $\gamma = p_0 \alpha \ell^2 / \Lambda$ et on établit les résultats suivants :

— si $\gamma < 0,88$, un régime stationnaire s'établit peu à peu. La température est maximale au centre et atteint $T_{1\max} = T_0 + \theta_1/\alpha$ où θ_1 est la plus petite solution de l'équation

$$e^{-\theta_1/2} \operatorname{argch}\left(e^{\theta_1/2}\right) = \sqrt{\gamma/2} \quad .$$

— si $\gamma > 0,88$, la température croît sans cesse et tend vers l'infini, ce qui correspond à une explosion.

1. Coder la fonction `evolution_reacteur(T_init)` qui joue le même rôle que `evolution`. Attention : le terme source se modifie au fil du temps donc il convient de le recalculer à chaque itération. Il a été traité comme une variable globale et pour que ses modifications se répercutent hors de la fonction, y compris dans l'exécution des autres fonctions, il faut utiliser le mot clé `global`.

2. Calculer et observer l'évolution de la température pour diverses valeurs de γ (par exemple en modifiant p_0). Observer le changement de régime selon que $\gamma < 0,88$ ou que $\gamma > 0,88$.

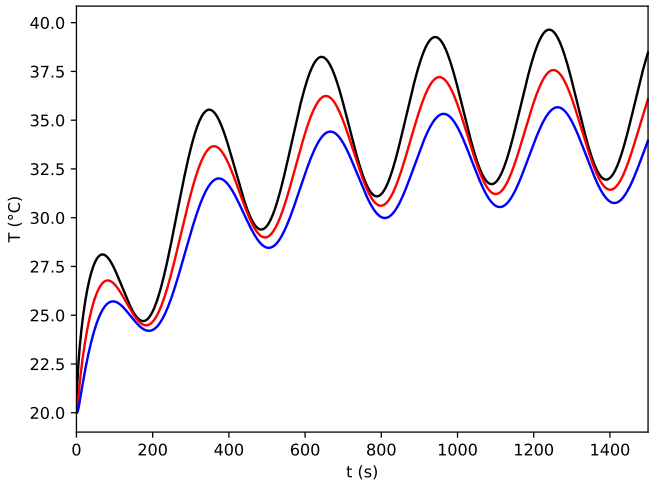
IV.2 Chauffage alternatif d'une barre, onde thermique

En travaux pratique, nous utiliserons une barre de cuivre dont l'extrémité d'abscisse $x = L$ est maintenue à température constante T_0 par une circulation d'eau, alors qu'un flux réglable est imposé en $x = 0$. Pour produire des ondes thermiques, nous choisirons un flux alternatif de période T :

$$j(0,t) = \frac{\Phi_0}{S}\left(\frac{1+\cos(2\pi t/T)}{2}\right) \quad .$$

1. Coder la fonction `evolution_onde_flux(T_init)` jouant le même rôle que `evolution`. Attention : j_0 se modifie au fil du temps donc il convient de le recalculer à chaque itération. Il a été traité comme une variable globale et pour que ses modifications se répercutent hors de la fonction, y compris dans l'exécution des autres fonctions, il faut utiliser le mot clé `global`.

2. Calculer et représenter l'évolution de la température sur 40 minutes. Le dessin animé est particulièrement instructif. Je vous demande aussi de tracer $T(x_i, t)$ en fonction du temps, pour diverses valeurs de x_i. Vous devriez ainsi obtenir des courbes du type de celle ci-dessous, en accord avec les relevés expérimentaux en travaux pratiques.

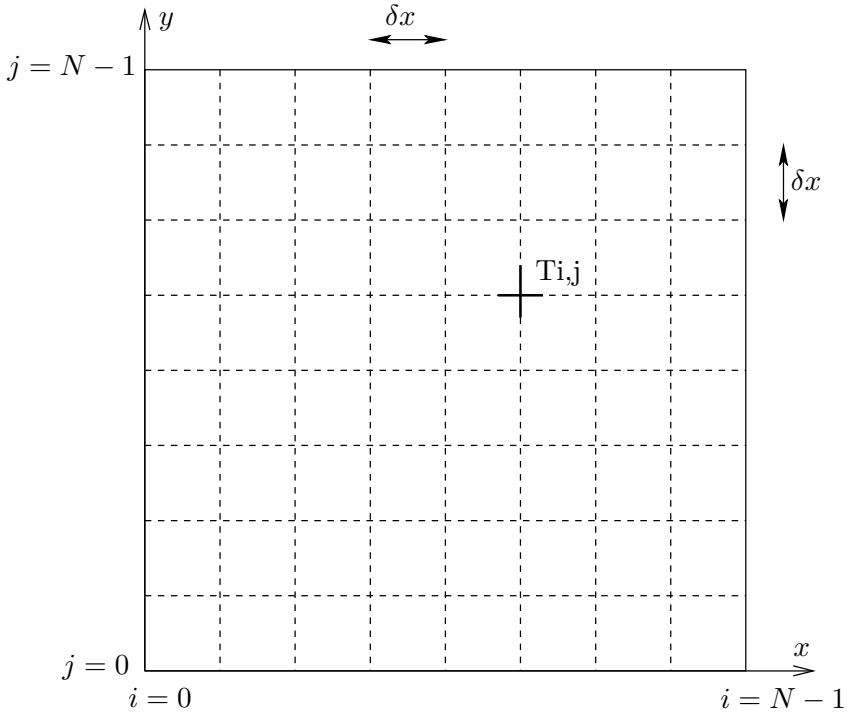


IV.3 Passage en dimension 2

Le programme prévoit que nous nous limitions à des problèmes unidimensionnels, mais l'extension de la méthode en dimension 2 ne pose pas de difficulté de principe. Dans ce cas, la discrétisation de l'espace se fait à la fois selon (Ox) et selon (Oy), ce qui amène à représenter le champ de température à un instant donné par un tableau à deux indices `T[i, j]` $= T(x_i, y_j)$ et son évolution spatio-temporelle par un tableau à 3 indices `champ_T` de taille $(N_{\text{pas}} + 1) \times N \times N$. Dans ce cas, le laplacien intervenant dans l'équation de la diffusion s'écrit

$$\Delta T = \frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} \quad .$$

Je vous demande simplement de réfléchir à ce que devient la relation d'itération (3) en dimension 2 et à la manière dont il convient de l'implémenter dans les fonctions `pdt_naif` et `pdt_vecto`.


y
δx
j = N − 1
δx
Ti,j
x
j = 0
i = 0
i = N − 1