CONCOURS COMMUNS POLYTECHNIQUES

ÉPREUVE SPÉCIFIQUE - FILIÈRE PC

MODÉLISATION DE SYSTÈMES PHYSIQUES OU CHIMIQUES

Jeudi 3 mai : 8 h - 12 h

N.B. : le candidat attachera la plus grande importance à la clarté, à la précision et à la concision de la rédaction. Si un candidat est amené à repérer ce qui peut lui sembler être une erreur d'énoncé, il le signalera sur sa copie et devra poursuivre sa composition en expliquant les raisons des initiatives qu'il a été amené à prendre.

Les calculatrices sont autorisées

Le sujet est composé de deux parties (pages 1 à 14)
et d'une annexe (pages 15 à 18).

PROBLÈME

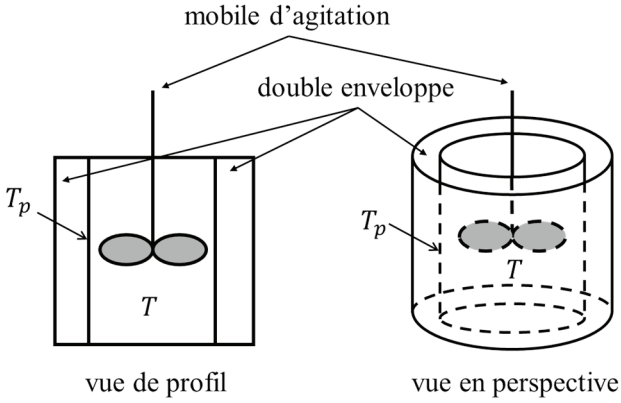
Étude d'une réaction exothermique : stabilité thermique en réacteur fermé

Présentation du problème

De nombreux procédés industriels font intervenir des réacteurs fermés pour la synthèse de molécules à haute valeur ajoutée. Pour optimiser le rendement de la synthèse, il est nécessaire de bien comprendre l'influence des paramètres physiques (comme la température de la réaction,…) sur la marche du réacteur. La maîtrise des échanges thermiques est cruciale dans le cas des réactions exothermiques car la chaleur dégagée par la réaction provoque une augmentation de la température du mélange réactionnel. Selon les conditions opératoires, cette augmentation de température peut entraîner un emballement thermique du réacteur et conduire à des dégâts irréversibles.

L'accident de Seveso le 10 juillet 1976 illustre les problèmes liés à l'emballement thermique des réacteurs. Il s'agissait d'un procédé de production de 2,4,5-trichloro-phénol à partir de 1,2,4,5-tétrachloro-benzène et de soude dans un solvant (l'éthylène glycol) à une température voisine de 150 °C et à pression atmosphérique en réacteur fermé. La mauvaise maîtrise de la température de la réaction a entraîné le déroulement de réactions secondaires conduisant d'une part à une augmentation de la température et de la pression et d'autre part à la formation de produits secondaires toxiques : les dioxines. La rupture de la soupape de sécurité due à l'augmentation de la pression a conduit au rejet de dioxines dans l'atmosphère.

L'étude de l'influence des paramètres physiques sur la marche d'un réacteur se fait la plupart du temps à l'échelle du laboratoire dans des dispositifs de dimensions beaucoup plus petites que celles des réacteurs utilisés dans les procédés industriels. La particularité du réacteur utilisé pour la présente étude est qu'il possède une géométrie cylindrique et qu'il est équipé d'une double enveloppe externe dans laquelle circule un fluide permettant de refroidir la paroi du réacteur et d'empêcher un emballement thermique (**figure 1**). L'emballement thermique survient lorsque la chaleur dégagée par la réaction excède la capacité du réacteur à évacuer l'énergie.



T et T_p représentent respectivement la température dans le réacteur et la température à la paroi (côté refroidissement).

Figure 1 – Schéma simplifié d'un réacteur fermé parfaitement agité avec une double enveloppe pour son refroidissement

Pour caractériser le comportement thermique du réacteur, on commence la plupart du temps par une étude en l'absence de réaction. Cette étude permet dans un premier temps de caractériser la capacité du réacteur à évacuer l'énergie avec la détermination du coefficient de transfert thermique à la paroi. Dans un deuxième temps, on met en œuvre dans ce réacteur une réaction exothermique $R \rightarrow produits$. Ces études permettent de déterminer les valeurs de paramètres clefs intervenant dans les équations décrivant le comportement du réacteur (modèle théorique). L'utilisation de ce modèle théorique permet de prédire la stabilité thermique du réacteur. L'établissement du modèle théorique repose sur l'écriture de bilans de matière et de chaleur. Une fois que l'influence des conditions physiques sur la marche du réacteur est déterminée, on peut en déduire les conditions de stabilité du réacteur industriel.

Ce sujet est constitué de deux parties. La première partie porte sur la modélisation du réacteur fermé parfaitement agité avec double enveloppe. Elle permet l'établissement du système d'équations différentielles régissant les variations de la conversion du réactif et de la température en fonction du temps, ainsi que la détermination de la valeur de paramètres physico-chimiques intervenant dans ces équations. La deuxième partie porte sur le traitement numérique des données expérimentales avec la détermination des paramètres d'un modèle par régression linéaire, puis la prédiction du comportement thermique du réacteur par résolution d'un système d'équations différentielles par la méthode d'Euler implicite.

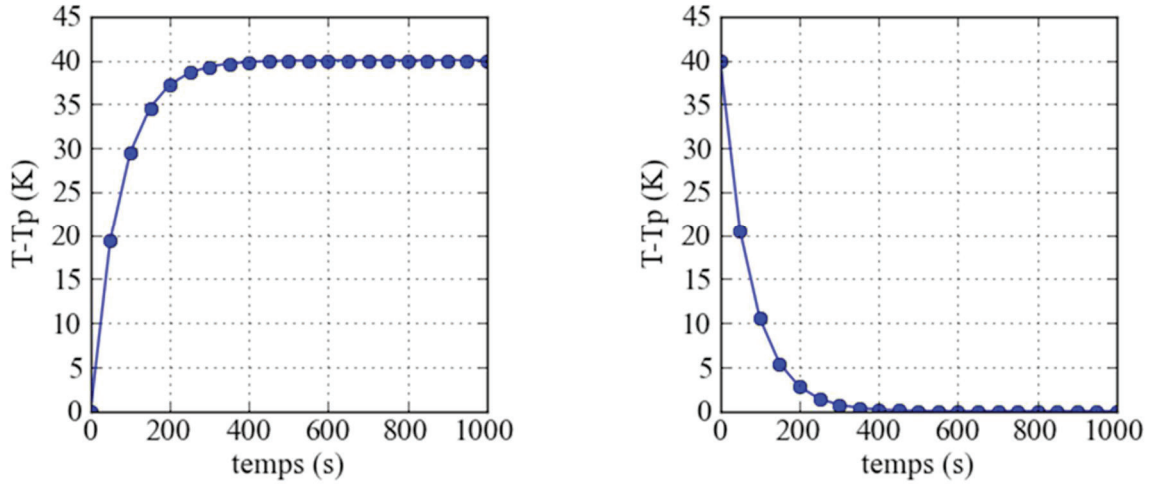
Partie I – Modélisation du réacteur fermé parfaitement agité avec double enveloppe

I.1 – Etalonnage thermique du réacteur

Dans cette partie, on souhaite caractériser les transferts de chaleur entre un liquide contenu à l'intérieur du réacteur et la paroi en l'absence de toute réaction chimique. On supposera que la capacité thermique massique et la masse volumique de ce liquide sont constantes quelle que soit la température. Pour caractériser ces transferts de chaleur, on réalise deux expériences successives avec la température de la paroi, T_p, maintenue constante dans les deux cas.

- La première expérience consiste à chauffer le liquide (initialement à une température identique à celle de la paroi) avec un dispositif annexe (une résistance chauffante) dissipant une puissance thermique P_{th} de 96,0 W. On constate que la température de la phase liquide augmente, puis atteint une valeur asymptotique en régime permanent (**figure 2a**).

- Après avoir atteint le régime permanent lors de la phase de chauffe, on réalise une seconde expérience en coupant le chauffage. La température du liquide décroit jusqu'à ce qu'elle tende vers la température de la paroi (**figure 2b**).



a) pendant la phase de chauffe b) pendant la phase de refroidissement

Figure 2 – Évolution de la température du liquide dans le réacteur

On exploite d'abord la courbe obtenue lors de la phase de chauffe (**figure 2a**) pour déterminer le coefficient de transfert de chaleur à la paroi, noté U (unité : $W \cdot m^{-2} \cdot K^{-1}$). Ce coefficient rend compte des échanges de chaleur entre la phase réactionnelle et le fluide caloporteur dans la double enveloppe à travers la paroi du réacteur. Dans le cas présent, la température T_p étant la température de paroi du côté du fluide caloporteur, il s'agit d'un coefficient de transfert thermique global qui tient compte du transfert convectif côté réaction et du transfert par conduction dans la paroi qui sépare les deux fluides.

Pour obtenir la valeur de U, on commence par établir l'équation différentielle qui régit l'évolution de la température T en fonction du temps en réalisant un bilan d'énergie.

Le bilan d'énergie, conséquence directe du premier principe de la thermodynamique, appliqué au système constitué par la phase réactionnelle lors de la phase de chauffe, conduit à l'équation différentielle suivante (équation (1))

$$(\rho \times V \times Cp)\frac{dT}{dt} = P_{th} - U \times A \times (T - T_p), \quad (1)$$

où T est la température du fluide à l'intérieur du réacteur, ρ, V et C_p sont respectivement la masse volumique, le volume et la capacité thermique massique du fluide, P_{th} est la puissance thermique cédée par la résistance chauffante au milieu réactionnel, T_p est la température à la paroi, maintenue constante ($T_p = 320{,}0$ K) et A la surface latérale du réacteur sur laquelle le fluide à l'intérieur du réacteur est en contact avec la double enveloppe.

Q1. Interpréter concrètement chacun des trois termes du bilan d'énergie en précisant leur signification physique et vérifier qu'ils sont homogènes à des puissances.

Q2. Donner l'expression de $T - T_p$ en régime permanent. Il est rappelé que la température de la paroi, T_p, est maintenue constante tout au long des essais. Il est précisé que la température de la phase liquide à l'instant initial est égale à T_p.

Q3. D'après les résultats obtenus lors de la première expérience (**figure 2a**), donner la valeur de la différence de température $T - T_p$ lorsqu'on atteint le régime permanent. Calculer la valeur du coefficient de transfert de chaleur à la paroi dans les unités SI. On donne $T_p = 320{,}0$ K, $\rho = 1\,000{,}0$ kg·m^{-3}, $V = 0{,}1 \times 10^{-3}$ m^3 et $C_p = 1\,800{,}0$ J·kg^{-1}·K^{-1}, $P_{th} = 96{,}0$ W et $A = 8{,}0 \times 10^{-3}$ m^2.

Q4. On souhaite faire apparaître un temps caractéristique d'échange thermique τ_c du système. Montrer que le bilan d'énergie peut se mettre sous la forme suivante (équation (2)) :

$$\frac{dT}{dt} = s + \frac{T_p - T}{\tau_c}. \quad (2)$$

Donner les expressions de s et τ_c. Vérifier que τ_c est homogène à un temps.

On souhaite maintenant exploiter les résultats obtenus lors de la phase de refroidissement (**figure 2b**) pour confirmer la valeur du temps caractéristique d'échange thermique déterminé précédemment.

Q5. Le bilan d'énergie établi à la question **Q4** est-il modifié ? Si oui, donner la nouvelle expression de $\frac{dT}{dt}$.

Q6. Donner l'expression de $T - T_p$ en fonction du temps t par résolution de l'équation différentielle. On notera T_{max} la température initiale lors de la phase de refroidissement.

Q7. Le tracé de $ln(T - T_p)$ (avec $T - T_p$ en K) en fonction du temps t (**figure 3**) donne une droite d'équation $y = -1{,}33 \times 10^{-2} \times x + 3{,}68$ (avec x en secondes). En déduire la valeur du temps caractéristique d'échange thermique τ_c. Calculer la valeur du coefficient de transfert de chaleur à la paroi et vérifier que cette valeur correspond à celle obtenue avec la première expérience.

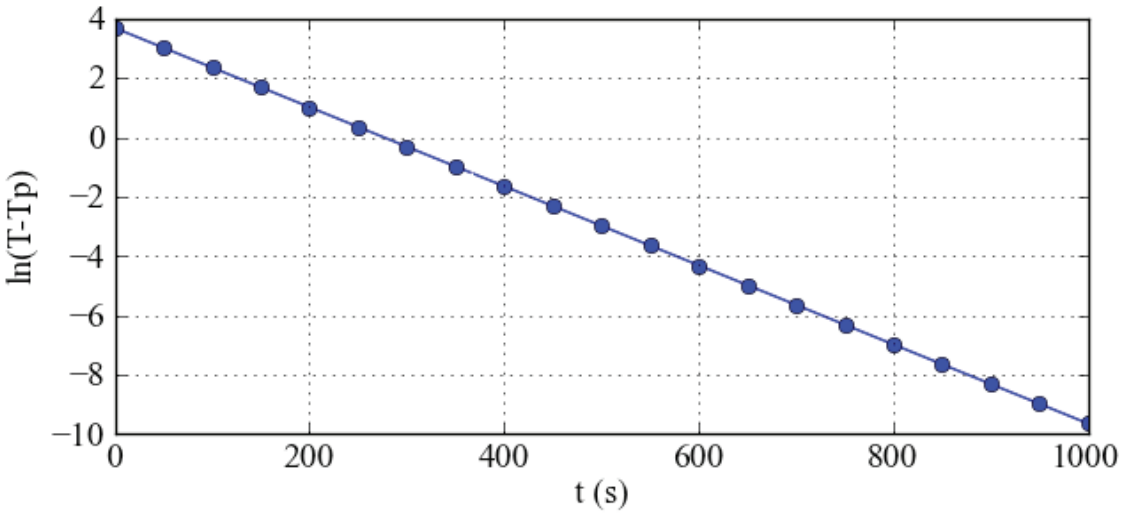


Figure 3 – Tracé de $ln(T - T_p)$ en fonction de t à l'aide des points enregistrés lors de la phase de refroidissement (**figure 2b**)

I.2 – Etude d'une réaction exothermique en réacteur fermé à double enveloppe

Dans cette sous-partie, on considère que l'on met en œuvre une réaction chimique exothermique $R \rightarrow produits$ (R est dissous dans un solvant) dans le même réacteur que celui dont on a étudié les échanges thermiques dans la sous-partie précédente. Il s'agit ici de caractériser le comportement thermique du réacteur en présence d'une réaction exothermique.

Le comportement du réacteur fermé parfaitement agité avec double enveloppe peut être représenté par un système constitué de deux équations différentielles ordinaires. La première équation différentielle ordinaire représente l'évolution temporelle de la conversion du réactif R ; la deuxième permet de caractériser l'évolution de la température de la réaction en fonction du temps.

Le réactif R étant dissout dans un solvant, on considère que le volume du mélange réactionnel V reste constant au cours du temps. On considère également que la réaction est homogène et qu'elle a lieu dans tout le volume du mélange réactionnel.

On commence par déterminer l'équation différentielle qui régit l'évolution de la conversion du réactif R en fonction du temps.

Q8. On considère que la réaction est d'ordre 1 par rapport au réactif R. Donner l'expression de la vitesse de la réaction (exprimée par unité de volume de mélange réactionnel) que l'on notera r (on notera C_R la concentration molaire du réactif R et k la constante cinétique). Préciser la dimension et l'unité de r dans le Système International.

Q9. Rappeler la loi d'Arrhenius donnant les variations de la constante de réaction en fonction de la température. On notera k_0 le facteur pré-exponentiel et E_a l'énergie d'activation. Préciser les dimensions et les unités SI de k, k_0 et E_a.

Q10. Écrire le bilan de matière sous la forme $\frac{dC_R}{dt} = f(C_R, T)$. Préciser l'expression de $f(C_R, T)$. Il est rappelé que le volume de la phase réactionnelle reste constant au cours du temps.

Q11. Dans le cas où le réacteur fonctionnerait en marche isotherme, résoudre l'équation différentielle et donner l'expression de la concentration de R en fonction du temps sous la forme $C_R = g(t)$. On notera C_R^0 la concentration initiale en R.

Q12. Pour simplifier les bilans, on introduit le taux de conversion de R, noté X_R, défini par la relation suivante : $X_R = (C_R^0 - C_R)/C_R^0$. Exprimer l'évolution de taux de conversion X_R en fonction du temps pour le cas de la marche isotherme.

Q13. Donner l'expression de l'équation différentielle qui régit l'évolution du taux de conversion X_R en fonction du temps dans le cas général (marche quelconque, c'est-à-dire non isotherme), sans chercher à la résoudre.

L'évolution de la température en fonction du temps est régie par une seconde équation différentielle obtenue en réalisant un bilan d'énergie sur le réacteur, conséquence directe du premier principe de la thermodynamique.

La réaction chimique qui se déroule dans le réacteur produit par unité de temps une variation de l'enthalpie du système réactionnel donnée par $S(t, X, T)$ (équation (3)) qui correspond à la chaleur dégagée par la réaction. Ce paramètre est appelé « terme source » dans la suite.

$$S(t, X_R, T) = -\Delta_r H^0(T) \times r(t, X_R, T) \times V, \qquad (3)$$

où V est le volume du mélange réactionnel, r est la vitesse de la réaction et $\Delta_r H^0(T)$ est l'enthalpie molaire standard de la réaction. Dans la suite, on suppose que $\Delta_r H^0(T)$ ne dépend pas de la température. On prendra $\Delta_r H^0(T) = \Delta_r H^0(T_p)$ que l'on notera $\Delta_r H^0$ pour simplifier.

Q14. Donner la dimension du terme source $S(t, X_R, T)$.

Q15. Montrer qu'un bilan enthalpique appliqué à un système que l'on précisera avec soin permet d'aboutir à la relation suivante (équation (4))

$$\frac{dT}{dt} = J\frac{dX_R}{dt} - \frac{T - T_p}{\tau_c}, \qquad (4)$$

où l'on exprimera J et τ_c en fonction de $\Delta_r H$, C_R^0, ρ, C_p, V, U et A. On admettra qu'il est légitime de négliger la contribution des réactifs, des produits et des accessoires situés à l'intérieur du réacteur au travers de leur capacités thermiques devant celle du solvant.

Q16. Donner l'expression du paramètre J ainsi que sa dimension.

Q17. Calculer la valeur du paramètre J en unité SI. On donne $\Delta_r H^0 = -360{,}0$ kJ·mol^{-1}, $\rho = 1\,000{,}0$ kg·m^{-3}, $C_p = 1\,800{,}0$ J·kg^{-1}·K^{-1} et $C_R^0 = 500{,}0$ mol·m^{-3}.

I.3 – Stabilité thermique du réacteur

Une première condition de stabilité, valable pour le cas d'une marche adiabatique, est que la température finale T_f de la réaction soit inférieure à une température maximale T_{max}, telle que $T_{max} = 1{,}25 \times T_p$.

Q18. Exprimer l'équation différentielle (équation (4)) dans le cas d'un fonctionnement adiabatique.

Q19. Déterminer alors l'expression de la température T en fonction du taux de conversion X_R, du paramètre J et de T_0 la température initiale du mélange réactionnel.

Q20. Donner l'expression de la température T_f atteinte en fin de réaction dans le cas d'une marche adiabatique sachant que $T_0 = T_p = 320{,}0$ K. Conclure quant à la stabilité du réacteur dans le cas de cette étude. Donner la signification physique du paramètre J.

Partie II – Traitement numérique des données expérimentales

Les algorithmes demandés au candidat **devront être réalisés dans le langage Python**. On supposera les bibliothèques « numpy » et « matplotlib.pyplot » chargées. Une **annexe** présentant les fonctions usuelles de Python est disponible pages 15 à 18. Les commentaires suffisants à la compréhension du programme devront être apportés et des noms de variables explicites devront être utilisés lorsque ceux-ci ne sont pas imposés.

II.1 – Détermination des paramètres d'un modèle par régression linéaire

Pour calculer la valeur du temps caractéristique d'échange thermique du réacteur à la question **Q7**, un modèle de régression linéaire simple a été estimé à partir des points expérimentaux enregistrés lors de la phase de refroidissement.

Le modèle de régression linéaire simple (fonction affine) est un modèle de régression d'une variable expliquée ($\ln(T - T_p)$ dans notre cas) sur une variable explicative (le temps t dans notre cas) dans lequel on fait l'hypothèse que la fonction qui relie la variable explicative à la variable expliquée est linéaire dans ses paramètres.

Soit n le nombre de points expérimentaux. Le modèle linéaire simple s'écrit de la manière suivante pour un point i ($1 \leq i \leq n$)

$$y_i = \widehat{\beta_1} \times x_i + \widehat{\beta_0}, \tag{5}$$

où $\widehat{\beta_0}$ et $\widehat{\beta_1}$ sont les paramètres du modèle, y_i est la variable expliquée et x_i est la variable explicative.

On propose de déterminer les paramètres du modèle par deux méthodes directes.

La méthode consiste à écrire le modèle (équation (5)) sous la forme matricielle $Y = L \times \hat{B}$. $\hat{B}$ est un vecteur colonne contenant les paramètres du modèle $\widehat{\beta_0}$ et $\widehat{\beta_1}$, Y est un vecteur colonne contenant les n valeurs y_i et L une matrice à n lignes et 2 colonnes, telle que $L(i,1) = \dfrac{\partial y_i}{\partial \widehat{\beta_0}}$ et $L(i,2) = \dfrac{\partial y_i}{\partial \widehat{\beta_1}}$. Rappelons que $\hat{B} = (L^t \times L)^{-1} \times L^t Y$ où L^t est la matrice transposée de L.

Q21. Donner les expressions de $\dfrac{\partial y_i}{\partial \widehat{\beta_0}}$ et $\dfrac{\partial y_i}{\partial \widehat{\beta_1}}$. En déduire la valeur des coefficients de la matrice L.

Q22. On suppose que les vecteurs colonnes Y et X, qui contiennent les valeurs y_i et x_i ($1 \leq i \leq n$), sont déjà créés. Donner le code permettant de créer la matrice L.

Q23. Donner le code permettant de déterminer les coefficients de la matrice $P = L^t \times L$. Préciser les dimensions de la matrice P.

Q24. Donner le code permettant de déterminer les coefficients de la matrice $Q = L^t \times Y$. Préciser les dimensions de la matrice Q.

Q25. Donner le code permettant de créer une fonction **`inv_mat(M)`** qui renvoie la matrice inverse de la matrice M de dimension (2×2) donnée comme argument d'entrée.

Q26. On note N la matrice inverse de M. Donner le code permettant de déterminer les coefficients $\widehat{\beta_0}$ et $\widehat{\beta_1}$ de la matrice $\hat{B}$.

II.2 – Prédiction du comportement thermique du réacteur

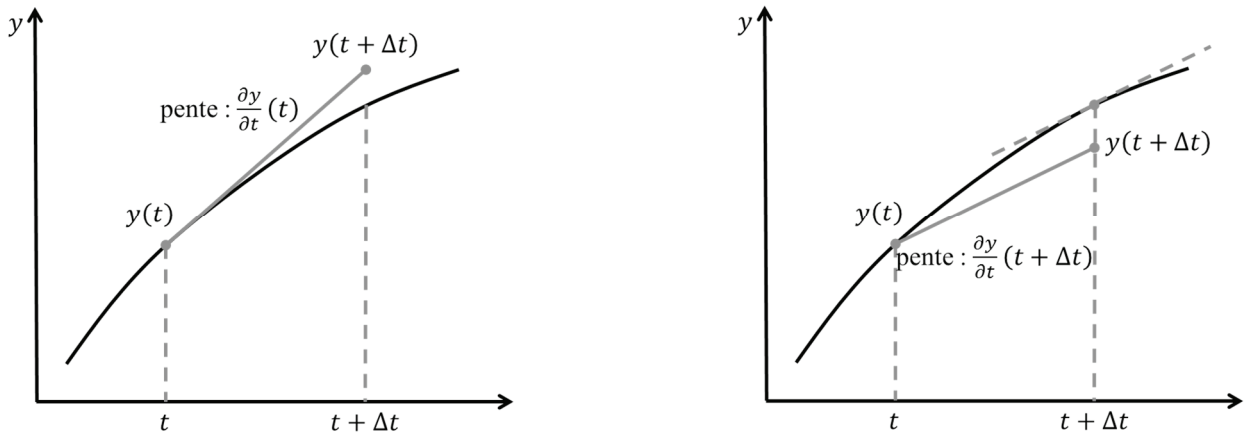
Dans cette sous-partie, on souhaite utiliser le modèle constitué du système d'équations différentielles établies dans la **Partie I** qui décrit l'évolution du taux de conversion du réactif X_R et de la température de réaction T en fonction du temps pour prédire le comportement thermique du réacteur en présence d'une réaction exothermique.

Pour simplifier les notations, on met le système d'équations différentielles sous la forme suivante (équation (6)) :

$$\begin{cases} \dfrac{dX_R}{dt} = f_1(t, X_R, T), \\ \dfrac{dT}{dt} = f_2(t, X_R, T). \end{cases} \tag{6}$$

La méthode de résolution proposée pour résoudre le système d'équations différentielles est la méthode d'Euler implicite à pas fixe. Cette méthode est préférée car elle donne de meilleurs résultats que la méthode explicite pour les systèmes dits raides (un système raide est un système qui est caractérisé par une évolution rapide des phénomènes en fonction du temps, ce qui est le cas ici pour la température de réaction).

Soit une variable y qui dépend du temps t. Comme la méthode d'Euler explicite, la méthode d'Euler implicite consiste à évaluer la valeur de $y(t + \Delta t)$ à partir de celle de $y(t)$ et de la dérivée $\dfrac{\partial y}{\partial t}$ (**figure 4**, page suivante). La différence entre les deux méthodes réside dans le choix de l'abscisse à laquelle est évaluée la dérivée $\dfrac{\partial y}{\partial t}$. Dans le cas de la méthode explicite, elle est évaluée en t : $\dfrac{\partial y}{\partial t}(t)$ comme le montre le schéma de la **figure 4a**, page suivante. Pour la méthode implicite, elle est évaluée en $t + \Delta t$: $\dfrac{\partial y}{\partial t}(t + \Delta t)$ (**figure 4b**, page suivante).



a) cas de la méthode explicite b) cas de la méthode implicite

Figure 4 – Approximation de $y(t+\Delta t)$ par la méthode d'Euler

Dans le cas d'un schéma implicite (**figure 4b**), l'expression de $y(t+\Delta t)$ en fonction de $y(t)$ et de la dérivée $\frac{\partial y}{\partial t}(t+\Delta t)$ évaluée en $t+\Delta t$ est obtenue en réalisant un développement limité dit rétrograde : $y(t) = y(t+\Delta t) - \Delta t \times \frac{\partial y}{\partial t}(t+\Delta t) + o(\Delta t)$.

Q27. À l'aide d'un développement limité rétrograde de la fonction $X_R(t)$, donner l'expression de $X_R(t+\Delta t)$ à l'ordre 1 en fonction de $X_R(t)$ et de sa dérivée partielle par rapport à t, $\frac{dX_R}{dt}(t+\Delta t)$ évaluée en $t+\Delta t$.

Q28. En déduire une valeur approchée de $\frac{dX_R}{dt}(t+\Delta t)$ à l'ordre 0 en fonction de $X_R(t)$, $X_R(t+\Delta t)$ et Δt.

Q29. À l'aide d'un développement limité rétrograde de la fonction $T(t)$, donner l'expression de $T(t+\Delta t)$ à l'ordre 1 en fonction de $T(t)$ et de sa dérivée partielle par rapport à t, $\frac{dT}{dt}(t+\Delta t)$.

Q30. En déduire une valeur approchée de $\frac{dT}{dt}(t+\Delta t)$ à l'ordre 0 en fonction de $T(t)$, $T(t+\Delta t)$ et Δt.

On procède à la discrétisation des équations. On note X_{Ri} la conversion évaluée au temps t_i, X_{Ri+1} la conversion évaluée au temps t_{i+1} et $\left.\frac{dX_R}{dt}\right|_{t_{i+1}}$ la dérivée de X_R évaluée à l'instant t_{i+1}. De même, on note T_i la température évaluée au temps t_i, T_{i+1} la température évaluée au temps t_{i+1} et $\left.\frac{dT}{dt}\right|_{t_{i+1}}$ la dérivée de T évaluée à l'instant t_{i+1}.

Q31. Donner l'expression de $\left.\frac{dX_R}{dt}\right|_{t_{i+1}}$ en fonction de X_{Ri}, X_{Ri+1} et Δt.

Q32. Donner l'expression approchée de X_{Ri+1} en fonction de X_{Ri}, Δt et de la fonction $f_1(t_{i+1}, X_{Ri+1}, T_{i+1})$, évaluée en t_{i+1}.

Q33. Donner l'expression de $\left.\frac{dT}{dt}\right|_{t_{i+1}}$ en fonction de T_i, T_{i+1} et Δt.

Q34. Donner l'expression approchée de T_{i+1} en fonction de T_i, Δt et de la fonction $f_2(t_{i+1}, X_{Ri+1}, T_{i+1})$, évaluée en t_{i+1}.

On constate que les expressions obtenues aux questions **Q32** et **Q34** constituent un système non linéaire dont les inconnues sont X_{Ri+1} et T_{i+1}. On propose d'utiliser la méthode de Newton-Raphson pour trouver les valeurs de X_{Ri+1} et T_{i+1} à chaque itération de la méthode d'Euler.

La méthode de Newton-Raphson pour la résolution d'un système de n équations non linéaires à n inconnues $x = (x_1, \dots, x_n)$, mis sous la forme de l'équation (7) suivante,

$$g(x) = \begin{bmatrix} g_1(x_1, \dots, x_n) \\ \dots \\ g_n(x_1, \dots, x_n) \end{bmatrix} = 0, \tag{7}$$

est une extension de la méthode de Newton permettant de trouver la racine d'une fonction d'une variable.

On peut démontrer la formule de récurrence suivante (équation (8)) :

$$x^{j+1} = x^j - \left(Dg\left(x^j\right)\right)^{-1} g\left(x^j\right), \tag{8}$$

où x^{j+1} est la valeur du vecteur x à l'itération $j+1$, x^j est la valeur du vecteur x à l'itération j, $g\left(x^j\right)$ est la valeur de $g(x)$ à l'itération j et $Dg\left(x^j\right)$ est la matrice Jacobienne évaluée en x^j (équation (9)) :

$$Dg\left(x^j\right) = \begin{bmatrix} \frac{\partial g_1}{\partial x_1} & \dots & \frac{\partial g_1}{\partial x_n} \\ \dots & & \dots \\ \frac{\partial g_2}{\partial x_1} & \dots & \frac{\partial g_2}{\partial x_n} \end{bmatrix}_{x=x^j}. \tag{9}$$

La formule de récurrence s'accompagne du choix d'une valeur initiale, notée x^0, et d'un critère d'arrêt, par exemple $\left\|x^{j+1} - x^j\right\| \leq \varepsilon$.

Q35. Transformer les expressions obtenues aux questions **Q32** et **Q34** pour les mettre sous la forme $g_1(X_{Ri+1}, T_{i+1}) = 0$ et $g_2(X_{Ri+1}, T_{i+1}) = 0$.

Q36. Donner les expressions de $\frac{\partial g_1}{\partial X_{Ri+1}}$, $\frac{\partial g_1}{\partial T_{i+1}}$, $\frac{\partial g_2}{\partial X_{Ri+1}}$ et $\frac{\partial g_2}{\partial T_{i+1}}$ permettant de construire la matrice Jacobienne $Dg(X_{Ri+1}, T_{i+1})$.

Q37. Écrire une fonction **`mat_Dg(x)`** qui a pour argument d'entrée un vecteur x contenant les valeurs de X_{Ri+1} et T_{i+1} à l'itération j et qui retourne la matrice Jacobienne $Dg(x^j)$. On supposera que les paramètres suivants ont été au préalable déclarés comme variables globales : $\Delta t = 0{,}01$ s, $k_0 = 5{,}0$ s^{-1}, $E_a = 20\,000{,}0$ J.mol^{-1}, $J = 100{,}0$ K, $\tau_c = 75$ s et $R = 8{,}314$ J.K^{-1}.mol^{-1}.

Q38. Écrire le code permettant de calculer les valeurs du vecteur x à l'itération $j + 1$ à l'aide de l'équation (8) lors d'une boucle de l'algorithme de Newton-Raphson. On notera **`x_old`** la valeur de x a l'itération j et **`x_new`** la valeur de x à l'itération $j + 1$. De même, on notera **`X_euleri`** et **`T_euleri`** les vecteurs contenant les valeurs de X_{Ri} et T_i à l'itération i de la méthode d'Euler implicite. Pour l'inversion de matrice, on utilisera la fonction **`inv_mat(M)`** écrite à la question **Q25**.

Q39. Pour obtenir la valeur de **`x_new`** par la méthode de Newton-Raphson, on souhaite créer une boucle itérative avec condition. La condition d'arrêt porte sur la valeur absolue de la différence des températures T_{i+1}^{j} et T_{i+1}^{j+1} que l'on souhaite inférieure à 10^{-5} K. Pour les valeurs initiales de X_{ri+1}^{0} et T_{i+1}^{0} on prendra respectivement 0,5 et $T_p + J/2$. Écrire le code correspondant. Il est inutile de recopier l'intégralité du code écrit à la question précédente ; on indiquera néanmoins sa place dans le code de cette question.

Maintenant que le code permettant de trouver les valeurs de X_{Ri+1} et T_{i+1} par la méthode de Newton-Raphson lors d'une itération de la méthode d'Euler implicite a été établi, on souhaite calculer les valeurs pour l'ensemble des itérations de la méthode d'Euler implicite. On rappelle que $X_R(t = 0) = 0$ et $T(t = 0) = T_p = 320{,}0$ K. En plus de noter **`X_euleri`** et **`T_euleri`** les vecteurs contenant les valeurs de X_{Ri} et T_i pour chaque itération i de la méthode d'Euler implicite, on notera **`t_euleri`** le vecteur contenant les valeurs de t_i à chaque itération. L'intégration sera réalisée sur l'intervalle $[t_0, t_f]$ avec $t_0 = 0$ s et $t_f = 1\,000$ s. Le pas de temps Δt est de 0,01 s.

Q40. Donner le code permettant de calculer le nombre m d'intervalles Δt compris dans l'intervalle $[t_0, t_f]$ (m est un entier).

Q41. Écrire le code permettant de calculer les valeurs des éléments des vecteurs **`X_euleri`**, **`T_euleri`** et **`t_euleri`** à chaque itération de la méthode d'Euler implicite.

Q42. Donner le code permettant de tracer les graphes de la **figure 5** montrant l'évolution de la conversion et de la température en fonction du temps que l'on obtiendrait en réalisant la résolution numérique du système d'équations différentielles (simulation réalisée avec $k_0 = 5{,}0\ \text{s}^{-1}$, $E_a = 20\,000{,}0\ \text{J·mol}^{-1}$, $J = 100{,}0\ \text{K}$, $\tau_c = 75\ \text{s}$).

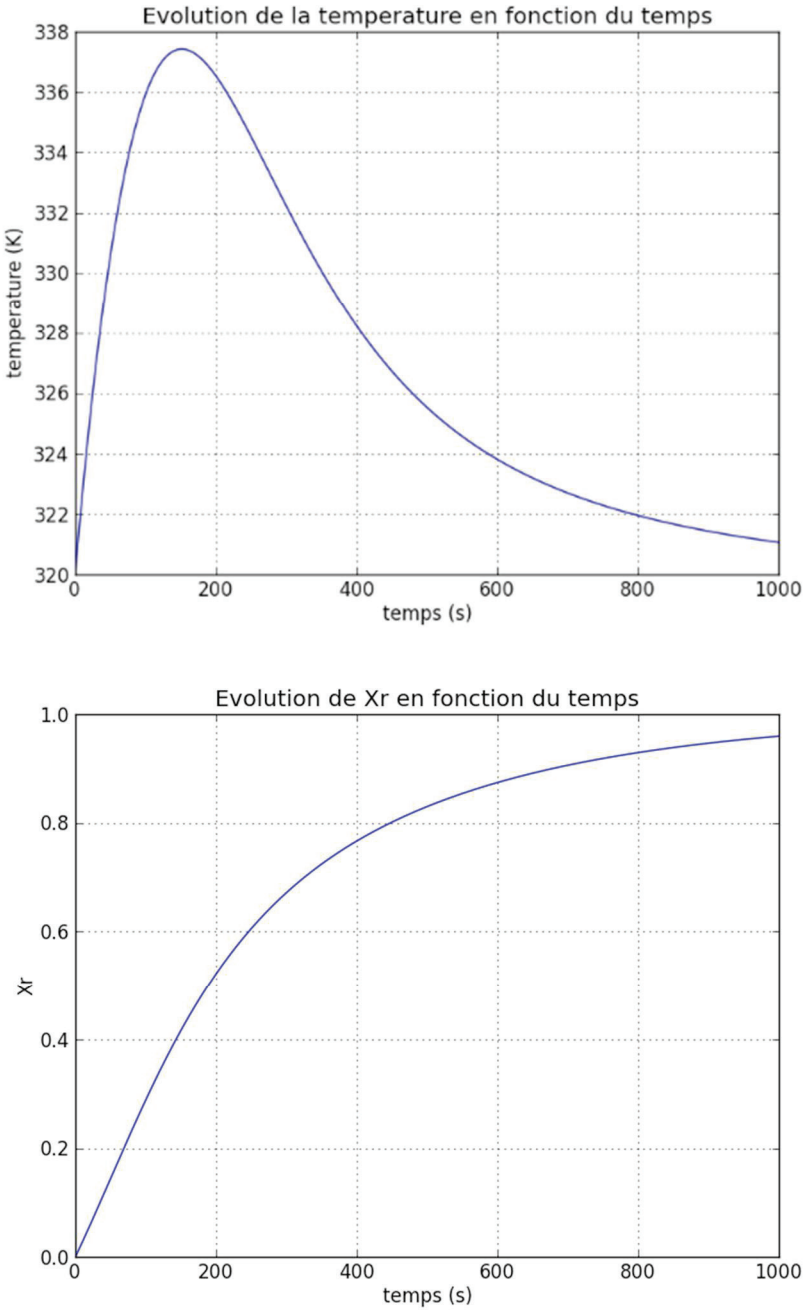


Figure 5 – Température et conversion calculées à partir du modèle constitué du système d'équations différentielles déterminées dans la **Partie I**

FIN

Annexe : Fonctions de Python

1. Bibliothèque numpy de Python

Dans les exemples ci-dessous, la bibliothèque numpy a préalablement été importée à l'aide de la commande : **import numpy as np**

On peut alors utiliser les fonctions de la bibliothèque, dont voici quelques exemples :

np.array(liste)

Description : fonction permettant de créer une matrice (de type tableau) à partir d'une liste.
Argument d'entrée : une liste définissant un tableau à 1 dimension (vecteur) ou 2 dimensions (matrice).
Argument de sortie : un tableau (matrice).

Exemples : np.array([4,3,2])
⇨ [4 3 2]

np.array([[5],[7],[1]])
⇨ [[5]
[7]
[1]]

np.array([[3,4,10],[1,8,7]])
⇨ [[3 4 10]
[1 8 7]]

$A[i,j]$.

Description : fonction qui retourne l'élément $(i+1, j+1)$ de la matrice A. Pour accéder à l'intégralité de la ligne i+1 de la matrice A, on écrit $A[i,:]$. De même, pour obtenir toute la colonne j+1 de la matrice A, on utilise la syntaxe $A[:,j]$.
Arguments d'entrée : une liste contenant les coordonnées de l'élément dans le tableau A.
Argument de sortie : l'élément $(i+1, j+1)$ de la matrice A.
ATTENTION : en langage Python, les lignes d'un tableau A de dimension $n \times m$ sont numérotées de 0 à $n-1$ et les colonnes sont numérotées de 0 à $m-1$

Exemple : A=np.array([[3,4,10],[1,8,7]])

A[0,2]
⇨ 10

A[1,:]
⇨ [1 8 7]

A[:,2]
⇨ [10 7]

np.zeros((n,m))

Description : fonction créant une matrice (tableau) de dimensions $n \times m$ dont tous les éléments sont nuls.
Arguments d'entrée : un tuple de deux entiers correspondant aux dimensions de la matrice à créer.
Argument de sortie : un tableau (matrice) d'éléments nuls.

Exemple : np.zeros((3,4))

⇨ [[0 0 0 0]
[0 0 0 0]
[0 0 0 0]]

np.linspace(Min,Max,nbElements)

Description : fonction créant un vecteur (tableau) de $nbElements$ nombres espacés régulièrement entre Min et Max. Le 1er élément est égal à Min, le dernier est égal à Max et les éléments sont espacés de $(Max - Min)/(nbElements - 1)$:
Arguments d'entrée : un tuple de 3 entiers.
Argument de sortie : un tableau (vecteur).

Exemple : np.linspace(3,25,5)

⇨ [3 8.5 14 19.5 25]

np.loadtxt('nom_fichier',delimiter='string',usecols=[n])

Description : fonction permettant de lire les données sous forme de matrice dans un fichier texte et de les stocker sous forme de vecteurs.
Arguments d'entrée : le nom du fichier qui contient les données à charger, le type de caractère utilisé dans ce fichier pour séparer les données (par exemple un espace ou une virgule) et le numéro de la colonne à charger (ATTENTION, la première colonne porte le numéro 0).
Argument de sortie : un tableau.

Exemple : data=np.loadtxt('fichier.txt',delimiter=' ',usecols=[0])
#dans cet exemple data est un vecteur qui correspond à la première #colonne de la matrice contenue dans le fichier 'fichier.txt'.

2. Bibliothèque matplotlib.pyplot de Python

Cette bibliothèque permet de tracer des graphiques. Dans les exemples ci-dessous, la bibliothèque matplotlib.pyplot a préalablement été importée à l'aide de la commande :
import matplotlib.pyplot as plt

On peut alors utiliser les fonctions de la bibliothèque, dont voici quelques exemples :

plt.plot(x,y)

Description : fonction permettant de tracer un graphique de n points dont les abscisses sont contenues dans le vecteur x et les ordonnées dans le vecteur y. Cette fonction doit être suivie de la fonction **plt.show()** pour que le graphique soit affiché.
Arguments d'entrée : un vecteur d'abscisses x (tableau de dimension n) et un vecteur d'ordonnées y (tableau de dimension n).
Argument de sortie : un graphique.

Exemple :

```
x= np.linspace(3,25,5)
y=sin(x)
plt.plot(x,y)
plt.title('titre_graphique')
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```

plt.title('titre')

Description : fonction permettant d'afficher le titre d'un graphique.
Argument d'entrée : une chaîne de caractères.

plt.xlabel('nom')

Description : fonction permettant d'afficher le contenu de nom en abscisse d'un graphique.
Argument d'entrée : une chaîne de caractères.

plt.ylabel('nom')

Description : fonction permettant d'afficher le contenu de nom en ordonnée d'un graphique.
Argument d'entrée : une chaîne de caractères.

plt.show()

Description : fonction réalisant l'affichage d'un graphe préalablement créé par la commande **plt.plot(x,y)**. Elle doit être appelée après la fonction plt.plot et après les fonctions plt.xlabel et plt.ylabel.

3. Fonction intrinsèque de Python

sum(x)

Description : fonction permettant de faire la somme des éléments d'un vecteur ou tableau.
Arguments d'entrée : un vecteur ou un tableau de réel, entier ou complexe.
Argument de sortie : un scalaire y qui est la somme des éléments de x.

Exemple : y= sum(x)
//y retourne la somme des éléments de x.

CONCOURS COMMUNS POLYTECHNIQUES

RAPPORT SUR L'ÉPREUVE ÉCRITE 2018 DE MODÉLISATION DE SYSTÈMES PHYSIQUE OU CHIMIQUE DE LA FILIÈRE PC

1/ CONSIGNES GÉNÉRALES :

a) Présentation du sujet

Le sujet portait sur l'étude de la stabilité thermique d'un réacteur au sein duquel était mise en œuvre une réaction chimique exothermique.

La première partie était relative à la modélisation du réacteur. Il s'agissait d'un réacteur parfaitement agité avec double enveloppe. Dans cette partie, le premier travail consistait à caractériser le comportement thermique du réacteur avec la détermination de paramètres physiques intervenant dans le bilan enthalpique. Le second travail demandé permettait d'établir le système d'équations différentielles ordinaires couplées dans le cas où une réaction exothermique a lieu dans le réacteur. Ce système permettait de caractériser l'évolution du taux de conversion du réactif et de la température de la phase réactionnelle en fonction du temps. Enfin, la troisième sous-partie, plus courte que les précédentes, était consacrée à l'étude de la stabilité thermique du réacteur.

La deuxième partie, consacrée au traitement numérique des données expérimentales, était divisée en deux tâches bien distinctes. La première tâche portait sur la détermination des paramètres d'un modèle par régression linéaire avec une méthode utilisant une écriture sous forme matricielle. Cette méthode nécessitait la construction de matrices et de vecteurs ainsi que l'inversion d'une matrice de dimension 2x2. La deuxième tâche portait sur l'intégration numérique du système d'équations différentielles ordinaires couplées établies dans la première partie. La méthode proposée était la méthode d'Euler implicite. Cette méthode nécessitait l'écriture de développements limités rétrogrades (évaluation de la dérivée en $t + \Delta t$ au lieu de t pour la méthode d'Euler explicite). Cette méthode conduisant à des expressions discrétisées dont les inconnues étaient le taux de conversion et la température à l'itération i+1. L'intégration numérique nécessitait donc à chaque pas la résolution numérique du système d'équations pour trouver les valeurs du taux de conversion et de la température à l'itération i+1. La méthode proposée était la méthode de Newton-Raphson, une extension de la méthode de Newton, permettant de résoudre le système par l'intermédiaire d'une matrice Jacobienne. Il s'agit d'une méthode itérative associée à un critère de convergence.

Le sujet était de difficulté moyenne et faisait appel à des notions transversales et complémentaires de chimie, de physique, de mathématique et d'informatique. Les parties étaient rédigées de manière indépendante pour ne pas bloquer les candidats qui auraient pu être en difficulté sur l'une ou l'autre des parties.

Le niveau de difficulté des questions était varié, ce qui a permis de classer les candidats. La longueur du sujet était adéquate étant donné le nombre de candidats ayant pu aborder le sujet dans son intégralité.

Une annexe présentait les principales fonctions de Python utiles à la résolution de ce sujet, ce qui permettait d'aider les candidats ne se souvenant plus de la syntaxe exacte des fonctions à utiliser.

b) Sur la prestation des candidats

Le seul langage informatique autorisé était PYTHON et cette consigne a été parfaitement suivie.

La première partie a été dans l'ensemble souvent traitée dans son intégralité. Ce n'est pas le cas de la partie informatique. L'écriture des développements limités à souvent été réalisée, mais les questions liées à l'écriture de codes beaucoup moins. Ceci est dommage pour une épreuve de modélisation dont la finalité n'est pas seulement l'établissement d'un modèle, mais aussi sa traduction en langage informatique en vue de son utilisation.

Le soin apporté à la rédaction des copies était dans l'ensemble correct, même si parfois les codes fournis dans les copies sont difficiles à lire (mal écrit, pas de couleur, pas de commentaires). Les indentations sont dans la majorité des cas respectées.

Les consignes ne sont pas toujours respectées (emploi de fonction alors que cela n'est pas demandé et que ce n'est pas utile).

Les dernières questions ont souvent été abordées dans l'esprit d'obtenir un maximum de points et n'ont pas été très bien traitées.

Les annexes ont été peu utilisées.

On peut classer les candidats en trois groupes :
• ceux qui ne sont ni à l'aise en physique/chimie, ni en informatique.
• ceux qui se débrouillent en informatique mais qui ne maitrisent pas la physique/chimie.
• ceux qui ont les bases dans les deux domaines.

2/ REMARQUES SPÉCIFIQUES :

Q1) Des confusions entre température, énergie et puissance ont été relevées dans certaines copies. Le sens physique des différents termes est mal compris. La rédaction des analyses dimensionnelles n'a pas toujours été très rigoureuse et justifiée de manière explicite (sachant que le résultat final était connu).

Q2) Certains candidats ont procédé à l'intégration de l'équation différentielle pour obtenir l'expression de la température en régime permanent alors que ce n'était pas nécessaire et un temps précieux a été perdu.

Q5) Pour la phase de refroidissement, la seule différence était l'absence de terme source ($P_{th} = 0$) puisqu'on ne chauffe plus le fluide. Dans certaines copies, on a lu que le refroidissement était traduit par un changement de signe de la température ou du terme P_{th}.

Q6) Un manque de rigueur a été observé dans l'intégration de l'équation différentielle, ce qui conduit à des erreurs notamment au niveau de la détermination de la condition initiale (confusion entre T et $T - T_p$).

Q8) Il y a parfois confusion entre l'expression de la vitesse de la réaction ($r = k \times C_R$) et la mesure de la vitesse dans le réacteur fermé ($\frac{dC_R}{dt} = -r$). Dans certaines copies, on a trouvé $r = -k \times C_R$.

Q9) Beaucoup d'erreurs ont été observées au niveau des dimensions et des unités. L'unité de l'énergie d'activation est trop souvent le joule alors qu'on attendait J/mol.

Q10) Très peu de copies donnent le bilan en terme de débit molaire ($\frac{dn_R}{dt} = -k \times C_R \times V$) en justifiant ensuite la simplification possible grâce au volume constant.

Q11) La notion « isotherme » est connue. Par contre, très peu pensent à donner la conséquence sur la constante cinétique qui est par conséquent constante.

Q14) La réponse donnée est souvent une puissance.

Q15) Le système est rarement défini. L'écriture du bilan est rarement justifiée.

Q16) L'expression de J a souvent été déduite à partir de sa dimension (au signe près), elle-même obtenue par analyse dimensionnelle de l'équation donnée dans l'énoncé.

Q17) Le signe de J est souvent faux, en particulier lorsque l'expression du paramètre a été obtenue de manière intuitive.

Q18) Il y a confusion entre adiabatique et isotherme. Pour certains, le système adiabatique se traduisait par $T - T_p = 0$. Pour d'autres, il se traduit par $\frac{dT}{dt} = 0$.

Q20) Quand T_f est calculée correctement, l'analyse de la stabilité du réacteur est en général spontanée et correcte.

Q21 – Q26) Les questions ont plutôt été bien traitées. On a noté toutefois des utilisations pas assez rigoureuses de la fonction intrinsèque sum avec des indices sans spécifications des bornes.

Attention à l'utilisation de fonctions de la bibliothèque numpy : la syntaxe est rarement correcte. Parfois, il vaut mieux passer un peu plus de temps à écrire un bout de code supplémentaire que d'utiliser une fonction intrinsèque dont on ne maitrise pas la syntaxe. Le sujet avait d'ailleurs été imaginé dans ce sens.

Attention également à l'écriture des noms de variables qui sont parfois non réalistes. Par exemple $\hat{B}$ ne convient pas pour l'écriture d'une variable dans un code. Il faut utiliser un nom du style B_chapeau.

Dans certaines copies, on trouve bien la formule de l'inverse d'une matrice 2x2, mais pas le code correspondant (Q25). La nullité du déterminant est rarement testée.

Q27 – Q34) Les questions ont été bien traitées dans l'ensemble. Il y a encore des copies où l'écriture des développements limités n'est pas rigoureuse. Il manque par exemple le $o(\Delta t)$ ou alors le signe $=$ est utilisé à la place du signe $\approx$.

Q35) Certains candidats n'ont pas bien compris la question et ont voulu démontrer que les fonctions g_1 et g_2 étaient nulles.

Q36) Les expressions des dérivées ont rarement été déterminées. Lorsqu'elles l'ont été, beaucoup d'erreurs, notamment de signes, ont été observées.

Q37) Voir ci-dessus.

Q38) La question a été peu abordée.

Q39) La boucle while est rarement correctement traitée, même si l'instruction while figure souvent dans le code fourni.

Q40) La question a été comprise mais la syntaxe est souvent fausse. Il y a une confusion entre entier (2 par exemple) et réel (2.0). Ainsi, la double division ne fonctionnait pas car elle renvoyait un réel.

Q41) La question a été peu abordée. Lorsqu'elle l'a été, l'écriture du code n'est pas assez rigoureuse (comme pour Q37 et Q39).

Q42) Les réponses sont généralement trop compliquées par rapport à ce qui était demandé.

CONCOURS COMMUNS POLYTECHNIQUES

EPREUVE SPECIFIQUE - FILIERE PC

MODELISATION DE SYSTEMES PHYSIQUES OU CHIMIQUES

Jeudi 4 mai : 8 h - 12 h

N.B. : le candidat attachera la plus grande importance à la clarté, à la précision et à la concision de la rédaction. Si un candidat est amené à repérer ce qui peut lui sembler être une erreur d'énoncé, il le signalera sur sa copie et devra poursuivre sa composition en expliquant les raisons des initiatives qu'il a été amené à prendre.

Les calculatrices sont autorisées

Le sujet est composé de deux parties, largement indépendantes.

Autour de l'équation de Poisson

Ce problème s'intéresse à la résolution numérique de quelques problèmes d'électrostatique. Il se compose de deux parties.

I. *tude de l'équation de Poisson et de différentes méthodes de résolution numérique.*

II. *Deux études de cas : fil infini chargé et mouvement d'un électron entre les plaques d'un condensateur.*

Les différentes parties sont largement indépendantes.

Un aide-mémoire `numpy/matplotlib/pyplot` *est présent à la fin du sujet.*

Partie I - quation de Poisson

I.1 - tablissement de l'équation

Q1. Rappeler l'équation de Maxwell-Gauss ainsi que la relation entre le champ $\vec{E}$ et le potentiel électrostatique V. En déduire l'équation de Poisson :

$$V + \frac{\rho}{\varepsilon_0} = 0 \, .$$

Préciser les noms et les unités usuelles de ρ et ε_0.

Q2. Citer plusieurs situations physiques en dehors de l'électrostatique pour lesquelles il existe une équation analogue.

I.2 - quation adimensionnée pour un problème plan

On veut résoudre l'équation de Poisson dans une portion de plan $\mathcal{P}$ carrée de côté L. On pose :

$$X = x/L, Y = y/L \, .$$

Q3. Montrer qu'on peut écrire l'équation sous la forme suivante :

$$\frac{\partial^2 V(X,Y)}{\partial X^2} + \frac{\partial^2 V(X,Y)}{\partial Y^2} + \rho'(X,Y) = 0$$

où $\rho'(X,Y)$ sera exprimé en fonction de ρ, L et ε_0.

I.3 - Discrétisation

Afin de résoudre numériquement l'équation de Poisson, on va utiliser un maillage de $\mathcal{P}$, de pas $h = 1/N$, et on va transformer les dérivées partielles par des différences entre les valeurs de V aux différents points du maillage (on parle aussi des *nœuds* du maillage). La **figure 1** (page suivante) représente le maillage de $\mathcal{P}$ pour $N = 5$.

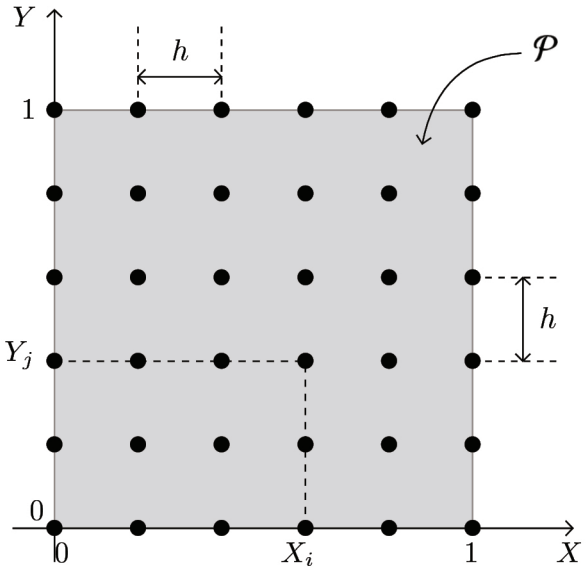


Figure 1 – Maillage de $\mathcal{P}$ pour $N = 5$

Q4. En faisant un développement limité à l'ordre 2 autour du point de coordonnées (X_i, Y_j), montrer qu'on peut exprimer la valeur de $\dfrac{\partial^2 V}{\partial X^2} + \dfrac{\partial^2 V}{\partial Y^2}$ en ce point sous la forme suivante :

$$\frac{\partial^2 V}{\partial X^2} + \frac{\partial^2 V}{\partial Y^2} = \frac{V(X_i + h, Y_j) + V(X_i - h, Y_j) + V(X_i, Y_j + h) + V(X_i, Y_j - h) - 4V(X_i, Y_j)}{h^2} + O(h)\,.$$

Q5. Comme $X_i = ih$ et $Y_j = jh$, on note désormais $V(i, j)$ le potentiel $V(X_i, Y_j)$ en un point (X_i, Y_j) du maillage. Montrer alors qu'on peut écrire l'équation de Poisson sous la forme suivante :

$$V(i + 1, j) + V(i - 1, j) + V(i, j + 1) + V(i, j - 1) - 4V(i, j) + \rho''(i, j) = 0 \qquad (1)$$

$\rho''(i, j)$ étant une fonction à définir en fonction de ρ, L, ε_0 et h.

I.4 - Résolution

La fonction $\rho''(i, j)$ étant connue, on montre en mathématiques que la solution de l'équation de Poisson est unique si on fixe les conditions aux limites sur la frontière $\mathcal{F}$ du domaine $\mathcal{P}$. Ces conditions sont essentiellement de deux types :

- on impose le potentiel en tout point de $\mathcal{F}$ (conditions de Dirichlet),
- on impose une condition sur les dérivées partielles de V en tout point de $\mathcal{F}$ (conditions de Neumann).

Dans ce problème, on ne va considérer que des conditions de Dirichlet.

La frontière $\mathcal{F}$ contient naturellement les points du bord de $\mathcal{P}$ (donc appartenant aux quatre côtés du carré), mais elle peut aussi contenir certains points à l'intérieur de $\mathcal{P}$ où le potentiel est fixé en raison de la présence d'électrodes.

L'ensemble des points de coordonnées (i, j) est donc composé de deux sous-ensembles :

- ceux dont le potentiel est connu, appartenant à la frontière $\mathcal{F}$,
- ceux dont le potentiel est inconnu, appartenant à $\mathcal{P}$ mais pas à $\mathcal{F}$ (donc dans $\mathcal{P}\backslash\mathcal{F}$).

Méthode de Jacobi

partir de l'équation (1), on peut exprimer :

$$V(i,j) = \frac{1}{4}(V(i+1,j) + V(i-1,j) + V(i,j+1) + V(i,j-1) + \rho''(i,j)) \ . \tag{2}$$

La résolution s'effectue alors en deux étapes.

— Initialisation

a) On fixe le potentiel des points de $\mathcal{F}$ à la valeur imposée physiquement (bords et électrodes).

b) On donne aux points de potentiel inconnu, donc appartenant à $\mathcal{P}\backslash\mathcal{F}$, une valeur arbitraire $V_0(i,j)$, en général nulle.

— Itérations

On calcule une nouvelle valeur $V_1(i,j)$ des potentiels en appliquant l'équation (2) pour tous les points de $\mathcal{P}\backslash\mathcal{F}$, tandis que $V_1(i,j) = V_0(i,j)$ pour les points de $\mathcal{F}$.

Le processus est répété jusqu'à obtenir des valeurs du potentiel quasiment stables. En notant k le nombre d'itérations, on a donc pour le point de coordonnées (i,j) n'appartenant pas à la frontière :

$$V_{k+1}(i,j) = \frac{1}{4}(V_k(i+1,j) + V_k(i-1,j) + V_k(i,j+1) + V_k(i,j-1) + \rho''(i,j)) \ . \tag{3}$$

La convergence de la méthode est vérifiée à l'aide du critère de convergence e_k, défini par :

$$e_k = \sqrt{\frac{1}{N^2}\sum_{i,j}(V_{k+1}(i,j) - V_k(i,j))^2} \ . \tag{4}$$

Le calcul sera stoppé au bout de k itérations, quand e_k deviendra inférieur à un seuil de convergence ε fixé arbitrairement.

Implémentation informatique

On va utiliser la bibliothèque `numpy` permettant une utilisation simple des tableaux de flottants à deux dimensions ; un aide-mémoire est disponible en fin de sujet.

Le chargement des bibliothèques classiques est assuré par les lignes suivantes :

```
# importation des bibliothèques
import numpy as np
import matplotlib.pyplot as plt
import math
```

On supposera que les tableaux `numpy` suivants, utilisés comme arguments dans les fonctions à définir dans les questions qui suivent, ont pour signification :

— `V[i,j]`, $(i,j) \in [\![0 \dots N]\!]^2$: tableau courant du potentiel en un point de $\mathcal{P}$,

— `rhos[i,j]`, $(i,j) \in [\![0 \dots N]\!]^2$: tableau contenant la densité de charge ρ'' en un point de $\mathcal{P}$,

— `frontiere[i,j]`, $(i,j) \in [\![0 \dots N]\!]^2$: tableau de booléens indiquant si le point de coordonnées (i,j) appartient ou non à $\mathcal{F}$. En particulier, tous les points du bord du domaine seront tels que `frontiere[i,j]==True`.

Q6. Écrire la fonction `nouveau_potentiel(V, rhos, frontiere, i, j)` retournant la nouvelle valeur du potentiel au point $(i, j) \in [\![0 \dots N]\!]^2$ selon l'équation (3).

Q7. Montrer que pour modifier toutes les valeurs contenues dans `V[i,j]` pendant une itération, il est nécessaire de disposer d'une copie de ce tableau.

On rappelle que l'attribut `shape` *permet de récupérer les dimensions d'un tableau* `numpy`.

Q8. Écrire la fonction `itere_J(V, rhos, frontiere)` modifiant la totalité du tableau `V[i,j]` lors d'une seule itération et retournant l'erreur calculée conformément à l'équation (4).

Q9. Écrire la fonction `poisson(f_iter, V, rhos, frontiere, eps)` ayant pour premier argument une fonction du même type que celle définie à la question précédente, pour dernier argument `eps` le seuil arbitraire de convergence ε et dont le rôle est de modifier le tableau des potentiels `V[i,j]` jusqu'à convergence.

I.5 - Améliorations

Méthode de Gauss-Seidel

C'est une modification de la méthode de Jacobi, pour laquelle on montre que la convergence est légèrement plus rapide. Supposons que l'on balaye le tableau des potentiels selon les indices i et j croissants : dans ces conditions, les points situés à gauche et en dessous du point courant ont déjà été calculés. On va utiliser ces nouvelles valeurs, probablement plus proches de la solution, dans la formule permettant le calcul de $V_{k+1}(i, j)$. Ceci donne l'algorithme de Gauss-Seidel :

$$V_{k+1}(i,j) = \frac{1}{4}(V_k(i+1,j) + V_{k+1}(i-1,j) + V_k(i,j+1) + V_{k+1}(i,j-1) + \rho''(i,j)) \,. \qquad (5)$$

Q10. Montrer qu'il n'est plus nécessaire de copier le tableau `V[i,j]` pour la mise à jour lors d'une itération en utilisant l'équation (5). Faut-il modifier la fonction `nouveau_potentiel` pour passer de la méthode de Jacobi à celle de Gauss-Seidel ?

Q11. Écrire la fonction `itere_GS(V, rhos, frontiere)` modifiant la totalité du tableau `V[i,j]` lors d'une seule itération et retournant l'erreur calculée conformément à l'équation (4).

Méthode de Gauss-Seidel adaptative

Les méthodes de Jacobi et de Gauss-Seidel n'utilisent pas la valeur de $V_k(i, j)$ pour calculer $V_{k+1}(i, j)$. La méthode de sur-relaxation (***S**uccessive ver **R**elaxation method*) consiste à calculer la nouvelle valeur d'un nœud comme une combinaison linéaire de la valeur courante et de celle donnée par le schéma de Gauss-Seidel. En introduisant le paramètre de relaxation ω, on a alors :

$$V_{k+1}(i,j) = (1-\omega)V_k(i,j) + \frac{\omega}{4}(V_k(i+1,j) + V_{k+1}(i-1,j) + V_k(i,j+1) + V_{k+1}(i,j-1) + \rho''(i,j)) \,. \quad (6)$$

L'étude mathématique de cette relation permet de montrer les résultats suivants :

- la méthode converge uniquement si $0 < \omega < 2$ et elle converge plus rapidement que la méthode de Gauss-Seidel si $1 < \omega < 2$,
- il existe une valeur optimale de ω qui permet la convergence avec un nombre d'itérations en $O(N)$ pour une valeur de ε fixée.

Pour la résolution de l'équation de Poisson envisagée dans ce problème (conditions de Dirichlet sur un maillage carré), on montre que la valeur optimale ω_{opt} est :

$$\omega_{\text{opt}} = \frac{2}{1 + \pi/N} \,. \tag{7}$$

Q12. Écrire la fonction `nouveau_potentiel_SOR(V, rhos, frontiere, i, j, omega)` retournant la nouvelle valeur du potentiel au point (i, j) selon l'équation (6).

Q13. Écrire la fonction `itere_SOR(V, rhos, frontiere)` optimale modifiant la totalité du tableau `V[i,j]` lors d'une seule itération et retournant l'erreur calculée conformément à l'équation (4).

La résolution du problème peut alors se faire par un appel de la forme

`poisson(itere_SOR, V, rhos, frontiere, eps)`,

les tableaux carrés `V, rhos, frontiere` étant de dimensions convenables pour représenter un maillage comportant $(N + 1)^2$ nœuds.

Q14. Quelle est la complexité temporelle de l'appel précédent quand $\omega = \omega_{\text{opt}}$?
La **figure 2** représente, pour $\varepsilon = 10^{-4}$, la durée d'exécution T (en secondes) en fonction de N.
Cette courbe est-elle en accord avec la complexité temporelle attendue ?
Quelle serait la durée d'exécution pour $N = 1\,000$? Commenter.

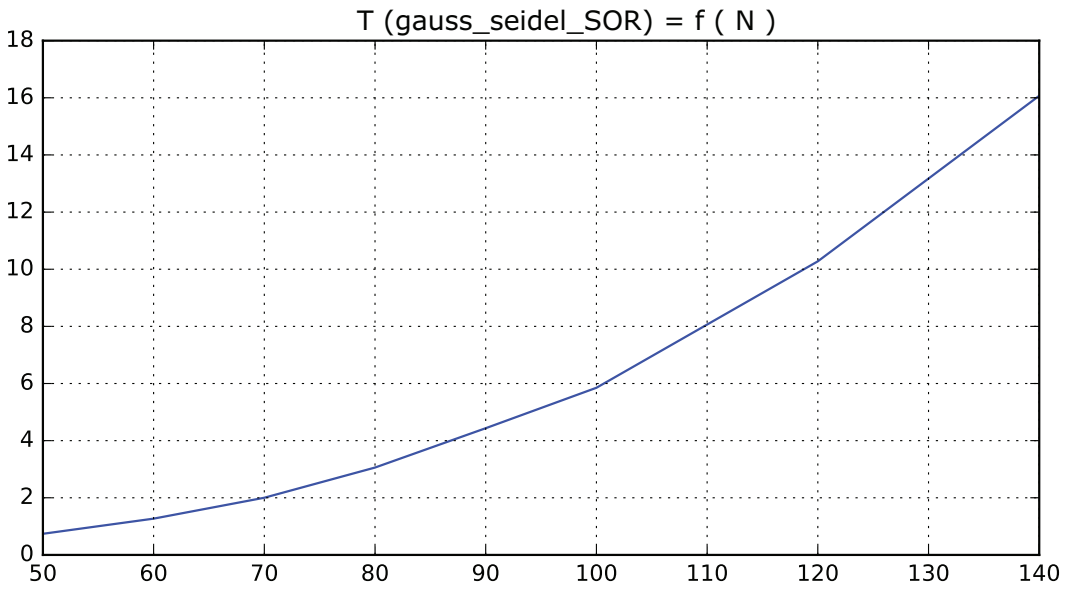


Figure 2 – Durée d'exécution (en secondes) de `poisson(itere_SOR, V, rhos, frontiere, eps)` en fonction de N

I.6 - Détermination du champ électrique

Connaissant le potentiel `V[i,j]`, il est souvent nécessaire de calculer numériquement les composantes E_x et E_y du champ électrique au niveau des nœuds du maillage, qui sont alors conservées dans deux tableaux `Ex` et `Ey` dimensionnés correctement (ce sont donc des tableaux carrés de $(N+1)^2$ éléments).

Q15. Expliquer rapidement comment il serait possible de définir la fonction `calc_ExEy(Ex, Ey, V, h)`, permettant, à partir du tableau `V` et du pas du maillage `h`, le remplissage des deux tableaux `Ex` et `Ey` passés en arguments.
Remarque : on ne demande pas d'écrire le code de la fonction, juste de décrire précisément les étapes de calcul, ainsi que les différents cas à considérer.

Partie II - Deux études de cas

II.1 - Fil cylindrique chargé uniformément

tude théorique

On considère dans le vide un fil cylindrique infini d'axe z et de rayon R, portant une charge volumique constante ρ.

Q16. En se plaçant en coordonnées cylindriques d'axe z, montrer par des considérations de symétrie et d'invariance que le champ $\vec{E} = E(r)\,\vec{u}_r$. En déduire la forme des surfaces équipotentielles.

Q17. En appliquant le théorème de Gauss, calculer le champ $\vec{E}$ dans tout l'espace. Tracer rapidement l'allure de $E(r)$ en fonction de r.

Q18. On donne : $\varepsilon_0 = 8{,}85 \times 10^{-12}$ F.m^{-1}, $\rho = 1{,}00 \times 10^{-5}$ C.m^{-3}, $R = 5{,}00$ cm. Calculer la valeur maximale de la norme du champ électrique, ainsi que la valeur pour $r = 2R$.

tude numérique

Pour pouvoir utiliser la méthode de Gauss-Seidel adaptative, on place le fil infini au centre d'une enceinte de longueur infinie et de section carrée ($L \times L$), portée au potentiel nul (**figure 3**).

Dans la suite, on prendra $L = 4R = 20{,}0$ cm.

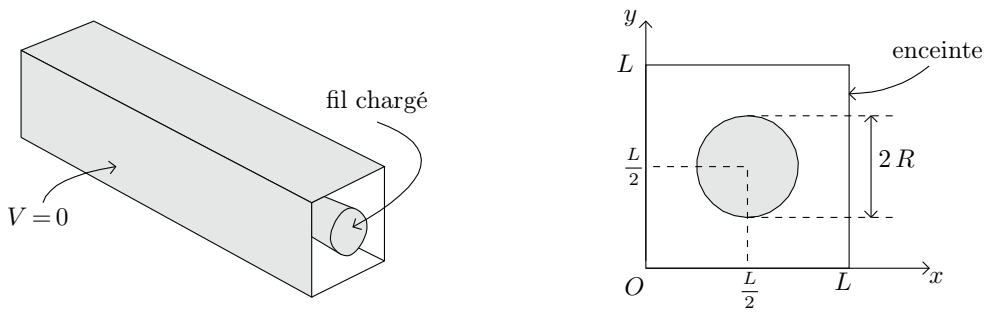


Figure 3 – Fil infini dans une enceinte de section carrée, portée au potentiel nul

Le programme permettant la résolution de ce problème commence ainsi :

```
# initialisations
eps0 = 8.85e-12                 # epsilon_0
L = 20.0e-2                     # 20 cm
N = 100 ; h = L/N               # définition du maillage
rho = 1.00e-5                   # densité vol. de charge rho

# les tableaux globaux numpy pour le cylindre chargé
rhos_cyl = np.zeros((N+1,N+1))  # tableau des valeurs de rho''
V_cyl = np.zeros((N+1,N+1))     # le potentiel
Ex_cyl = np.zeros((N+1,N+1))    # la composante Ex
Ey_cyl = np.zeros((N+1,N+1))    # la composante Ey

# le tableau définissant la frontière est initialement
# rempli entièrement par la valeur False
frontiere_cyl = np.zeros((N+1,N+1), bool)
```

Q19. Écrire la fonction `dans_cylindre(x,y,xc,yc,R)` retournant un résultat booléen indiquant si le point de coordonnées (x, y) est à l'intérieur ou sur le bord du cercle de centre (x_c, y_c) et de rayon R.

Q20. Écrire la fonction `initialise_rhos_cylindre(tab_rhos)`, initialisant le tableau `rhos_cyl` contenant les valeurs $\rho''(i, j)$ pour les nœuds du maillage.

Q21. Écrire la fonction `initialise_frontiere_cylindre(tab_f)`, mettant à `True` les points appartenant à la frontière, donc de potentiel fixé.

La résolution numérique avec la méthode de Gauss-Seidel adaptative, utilisant les valeurs numériques précédentes et un seuil de convergence $\varepsilon = 10^{-5}$, mène à la **figure 4**, où on a tracé un réseau de courbes équipotentielles, le potentiel V et la composante E_x du champ le long de l'axe de symétrie défini par $y = L/2$. En outre, la valeur calculée de `Ex_cyl[50, 50]` est égale à `-0.0023321214257521206`.

Q22. Commenter le plus complètement possible ces résultats ; on veillera, en particulier, à les comparer au modèle théorique (allure des courbes, valeurs numériques ...).

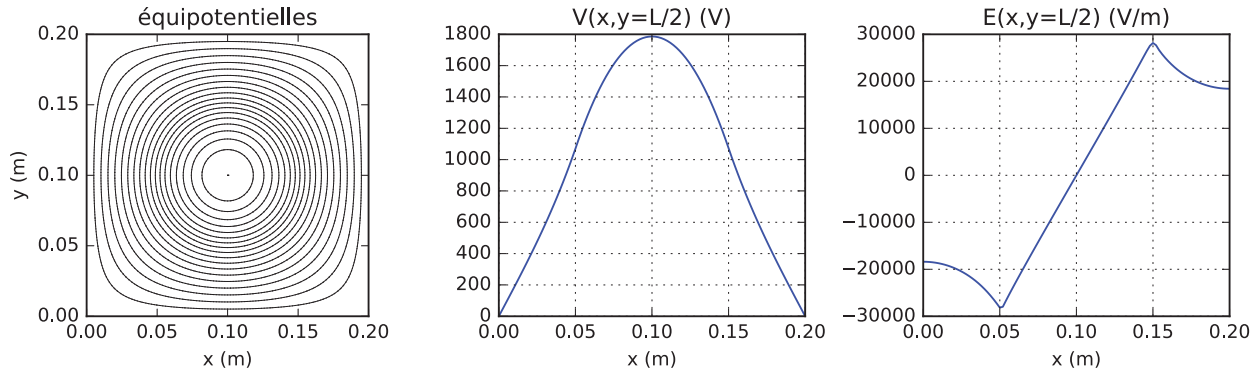


Figure 4 – Équipotentielles, $V(x, y)$ et $\vec{E}(x, y) \cdot \vec{u}_x$ en fonction de x pour $y = L/2$

Une autre résolution est effectuée, avec une répartition de charges dans le cylindre différente de la précédente, utilisant la même valeur de la densité volumique $\rho = 10^{-5}$ C.m^{-3}. Elle mène aux courbes de la **figure 5** (page suivante).

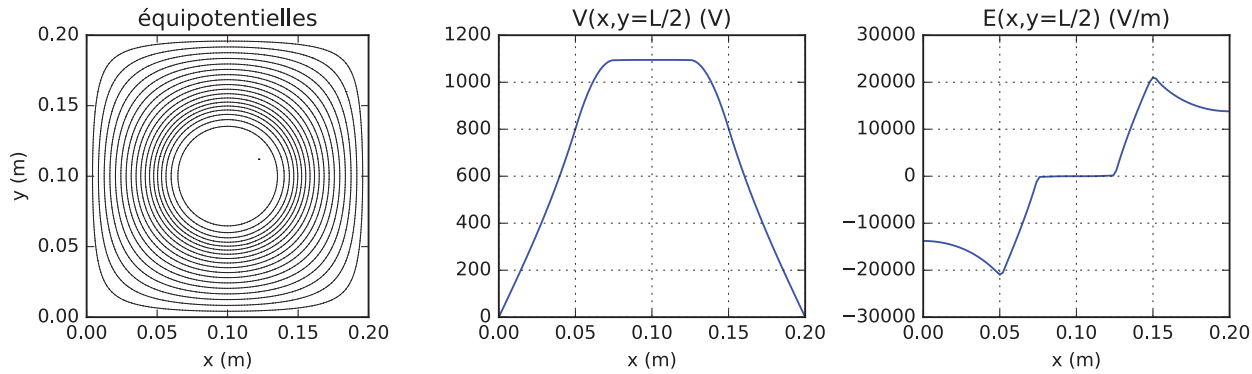


Figure 5 – Équipotentielles, $V(x, y)$ et $\vec{E}(x, y) \cdot \vec{u}_x$ en fonction de x pour la nouvelle répartition de charge

Q23. Déduire de ces courbes la répartition de charges dans le cylindre dans cette deuxième situation. Calculer le champ électrique en tout point pour cette répartition de charges dans le cas d'un cylindre infini seul dans l'espace.
On vérifiera que la valeur maximale du champ électrique calculée à l'aide de cette modélisation est compatible avec celle déduite de la **figure 5**.

II.2 - Mouvement d'un électron dans un tube d'oscilloscope

La **figure 6** montre un tube d'oscilloscope de petite dimension, dans lequel des électrons émis par la cathode sont accélérés et déviés vers un écran luminescent. La déviation est assurée par le passage des électrons entre les plaques de deux condensateurs plans : un pour la déviation horizontale, l'autre pour la déviation verticale. L'étude qui suit ne concernera que le condensateur responsable de la déviation verticale.

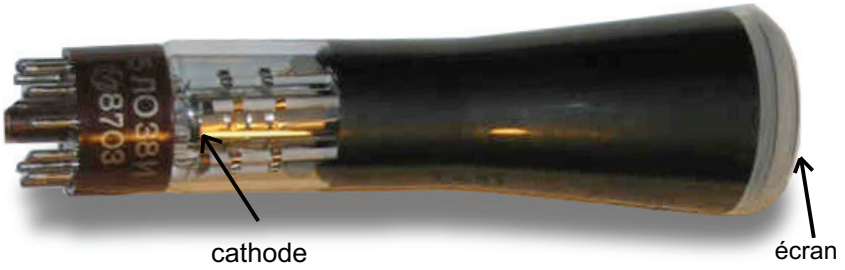


Figure 6 – Petit tube d'oscilloscope, de longueur d'environ 20 cm

On modélise la trajectoire d'un électron de la façon suivante (**figure 7** page suivante, où la zone de déviation est grisée) :

— on négligera l'effet de la pesanteur,
— émis à vitesse nulle par effet thermo-électronique au niveau de la cathode portée au potentiel nul, l'électron est accéléré à l'aide d'une tension $V_0 > 0$ afin d'acquérir à l'entrée de la zone de déviation une vitesse $\vec{v}_0$,
— pendant son trajet dans la zone de déviation, il est soumis à un champ électrique $\vec{E}$ lié aux potentiels $\pm V_p$ des plaques du condensateur, de longueur ℓ et séparées par une distance d,
— poursuivant son mouvement, il arrive sur la surface de l'écran à une distance y_s de l'axe x, l'écran étant situé à la distance D du centre du condensateur.

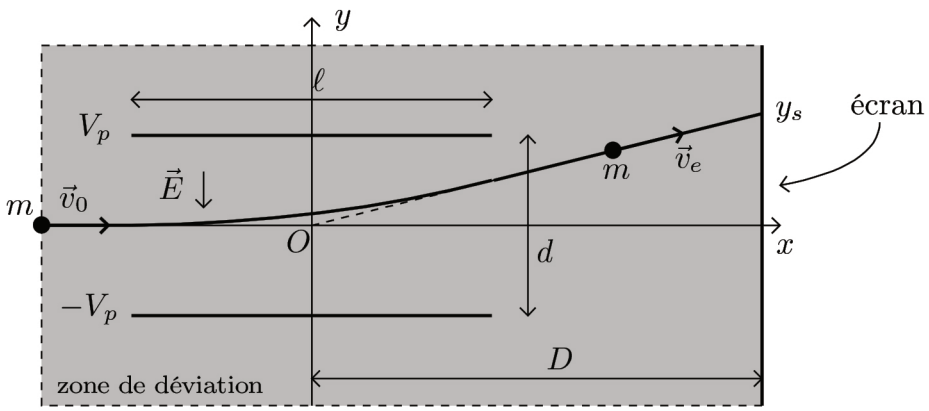


Figure 7 – Schéma de la zone de déviation

Valeurs numériques :

masse d'un électron $m = 9{,}11 \times 10^{-31}$ kg ; charge élémentaire $e = 1{,}60 \times 10^{-19}$ C

$V_0 = 950$ V ; $V_p = 180$ V ; $D = 7{,}00$ cm ; $d = 2{,}00$ cm ; $\ell = 4{,}00$ cm .

Étude physique

Q24. En appliquant la conservation de l'énergie, calculer la vitesse v_0 de l'électron à l'entrée de la zone de déviation. Faire l'application numérique. Commenter.

Q25. On modélise les plaques de déviation comme un condensateur sans effets de bord : le champ électrique est donc considéré comme nul si $|x| > \ell/2$ et uniforme si $|x| \leqslant \ell/2$, ses lignes de champ étant parallèles à l'axe Oy. Exprimer le champ $\vec{E}$ entre les plaques en fonction de V_p et d.

Q26. On suppose que la vitesse d'entrée de l'électron dans la zone de déviation est $\vec{v}_0 = v_0\,\vec{u}_x$. En appliquant les lois de la mécanique, établir l'équation de la trajectoire de l'électron entre les plaques (pour $|x| < \ell/2$).

Q27. Montrer que l'équation de la trajectoire pour $x > \ell/2$ est donnée par : $y = \dfrac{2eV_p\ell}{mdv_0^2}x$.

En déduire que l'ordonnée du spot sur l'écran est : $y_s = \dfrac{V_p}{V_0} \times \dfrac{\ell\, D}{d}$.

Faire l'application numérique pour y_s.

Étude numérique

Pour savoir si la modélisation précédente est pertinente, on va envisager une détermination numérique de la trajectoire de l'électron. Pour cela, on place le condensateur de déviation dans une enceinte carrée au potentiel nul de côté $L = 10{,}0$ cm, le centre du condensateur étant à 3,0 cm du bord gauche de l'enceinte (**figure 8** page suivante). Les autres caractéristiques électriques et géométriques sont les mêmes que précédemment.

La résolution numérique se déroule alors en deux étapes :

— calcul du potentiel et du champ électrique par la méthode de Gauss-Seidel adaptative dans l'enceinte,

— calcul de la trajectoire de l'électron à l'aide de la méthode d'Euler.

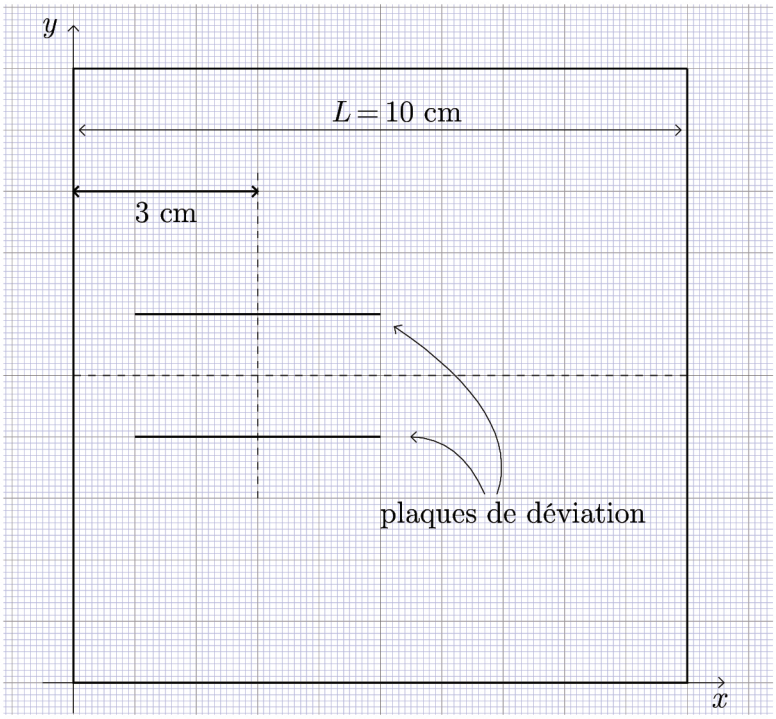


Figure 8 – Modélisation des plaques de déviation dans l'enceinte

Le programme permettant cette résolution numérique commence ainsi :

```
L = 0.100 ; N = 100 ; h = L/N
V0 = 950 ; Vp = 180
m = 9.11e-31 ; e = 1.60e-19
rhos_osc = np.zeros((N+1,N+1))
V_osc = np.zeros((N+1,N+1))
Ex_osc = np.zeros((N+1,N+1))
Ey_osc = np.zeros((N+1,N+1))
frontiere_osc = np.zeros((N+1,N+1), dtype=bool)
```

Calcul du potentiel et du champ électrique dans l'enceinte

Q28. Quelles sont les valeurs qui doivent être contenues dans le tableau `rhos` pour le problème considéré ?

Q29. Écrire la fonction `initialise_frontiere_condensateur(tab_V, tab_f)`, permettant l'initialisation des tableaux `V_osc` et `frontiere_osc` à l'aide de la ligne de code suivante :

```
initialise_frontiere_condensateur(V_osc, frontiere_osc)
```

Pour pouvoir utiliser la méthode d'Euler, il est nécessaire de pouvoir calculer les composantes $E_x(x,y)$ et $E_y(x,y)$ du champ $\vec{E}$ pour $x \in [0, L[$ et $y \in [0, L[$. Cependant, la méthode de résolution (associée à la fonction `calc_ExEy` définie dans la question **15**) ne permet de calculer les composantes `Ex_osc[i,j]` et `Ey_osc[i,j]` qu'aux nœuds du maillage.

Soit un point P de coordonnées $(x, y) \in [0, L[\times[0, L[$. Ce point est dans la cellule (i, j), où $i = \lfloor x/h \rfloor$ et $j = \lfloor y/h \rfloor$. Posons $r_x = x - ih$ et $r_y = y - jh$.

Q30. Montrer alors que :

$$E_x(x, y) \approx \texttt{Ex[i,j] + ((Ex[i+1,j] - Ex[i,j]) * rx + (Ex[i,j+1] - Ex[i,j]) * ry) / h}\,.$$

Écrire de même la formule permettant de calculer $E_y(x, y)$. En déduire qu'il est possible de calculer les composantes du champ en tout point de $\mathcal{P}$.

On supposera dans la suite que les fonctions `val_Ex(Ex,Ey,x,y,h)` et `val_Ey(Ex,Ey,x,y,h)` sont définies et retournent les valeurs des composantes du champ électrique pour le point M de coordonnées (x, y) calculées à l'aide des formules précédentes.

Calcul de la trajectoire par la méthode d'Euler

Q31. Montrer que les équations permettant de décrire le mouvement de l'électron par la méthode d'Euler sont les suivantes, avec δt comme petit incrément temporel et $\delta x, \delta v_x, \delta y, \delta v_y$ les variations pendant δt des grandeurs x, v_x, y, v_y :

$$\begin{cases} \delta x = v_x \delta t & \delta y = v_y \delta t \\ \delta v_x = -\dfrac{e}{m} E_x(x, y) \delta t & \delta v_y = -\dfrac{e}{m} E_y(x, y) \delta t \end{cases}.$$

Q32. Compte tenu du changement de la position d'origine du repère (imposée par la résolution numérique de l'équation de Poisson, **figure 8**), quelles sont les conditions initiales du mouvement de l'électron ?
Déterminer δt pour calculer environ 200 points successifs le long de la trajectoire. Faire l'application numérique.

Q33. En tenant compte des réponses aux questions précédentes, compléter le code d'initialisation des variables de la simulation (`******` dans le code suivant) :

```
Npts = 200 # nombre de points pour le tracé de la trajectoire
v0 = ******  # vitesse initiale de l'électron
dt = ******  # incrément temporel

# tableaux des coordonnées x et y de l'électron
lx = np.zeros(Npts) ; ly = np.zeros(Npts)
# tableaux des vitesses en x et en y
lvx = np.zeros(Npts) ; lvy = np.zeros(Npts)
#conditions initiales
lx[0] = ****** ; ly[0] = ******
lvx[0] = ******* ; lvy[0] = *******
```

Q34. Écrire les lignes de code implémentant la boucle de remplissage des tableaux `lx, ly, lvx, lvy` selon la méthode d'Euler.

Comparaison théorie/simulation

La **figure 9** montre le résultat de la simulation précédente. On y voit le réseau de courbes équipotentielles, ainsi que deux trajectoires 1 et 2, l'une étant associée au calcul théorique, l'autre à la simulation numérique. Chaque trajectoire est constituée de 200 points de calcul séparés d'une durée δt.

Q35. Reproduire sommairement sur la copie la **figure 9**, y ajouter le tracé de quelques lignes de champ orientées dans les différentes parties de la zone de déviation.
Identifier, en le justifiant, chaque trajectoire. Expliquer pourquoi la trajectoire 1 est plus courte que la trajectoire 2.

votre avis, peut-on se contenter de l'étude théorique pour prévoir le point d'impact de l'électron sur l'écran ? (On attend une réponse chiffrée.)

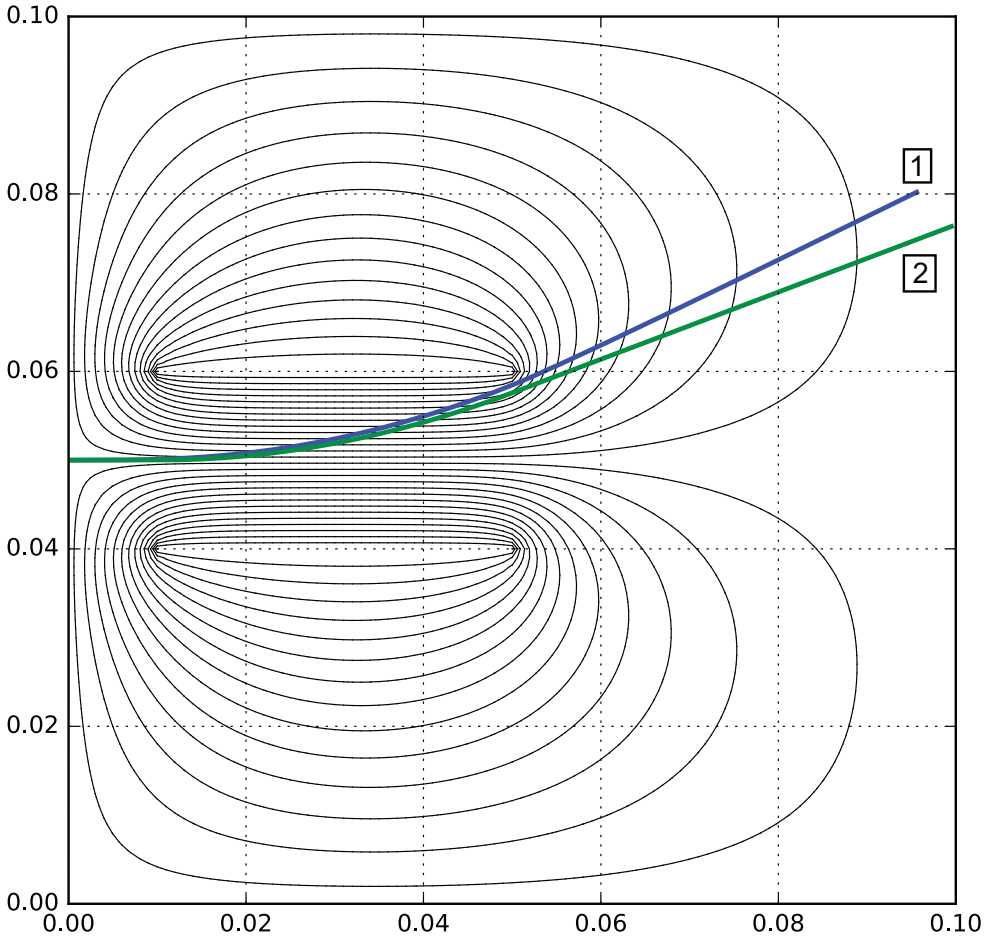


Figure 9 – Tracé dans la zone de déviation de quelques courbes équipotentielles, des trajectoires de l'électron (théorique et simulée numériquement)

Aide-mémoire numpy/matplotlib/pyplot

Importation des bibliothèques

Les bibliothèques sont importées de la façon suivante :

```
import math
import numpy as np
import matplotlib.pyplot as plt
```

Manipulation des tableaux numpy

La création d'un tableau `numpy` à deux dimensions dont toutes les valeurs sont initialisées à 0 est faite par l'instruction `np.zeros(format)`, `format` étant un doublet de la forme `(n_lignes , n_colonnes)` :

```
>> t0=np.zeros((2,3)); print(t0)
[[ 0.  0.  0.]
 [ 0.  0.  0.]]
```

Pour avoir un tableau rempli de 1, on utilise `np.ones(format)` :

```
>> t1=np.ones((2,2)); print(t1)
[[ 1.  1.]
 [ 1.  1.]]
```

On peut récupérer le format d'un tableau en demandant son attribut `shape`, ce qui retourne un doublet :

```
>> print(t0.shape); print(t1.shape)
(2, 3)
(2, 2)
```

Dans le cas d'un tableau carré, on peut donc récupérer le nombre de lignes, égal au nombre de colonnes, en accédant au premier élément du doublet :

```
>> t1.shape[0]
 2
```

On peut créer un tableau `numpy` de booléens en ajoutant le type `bool`. La valeur 0 est associée à `False`, la valeur 1 à `True` :

```
>> np.zeros((2,3), bool)
[[False False False]
 [False False False]]}
>> np.ones((2,3),bool)
[[ True True True]
 [ True True True]]
```

L'accès à un élément du tableau `a` (en lecture ou en modification) se fait par `a[i,j]`, les lignes et les colonnes étant numérotées à partir de 0 :

```
>> a=np.zeros((2,3)) ; a[0,0]=1 ; a[1,2]=2
>> a
[[ 1. 0. 0.]
 [ 0. 0. 2.]]
```

Une copie indépendante d'un tableau `a` se fait à l'aide de `np.copy(a)`

```
>> b = np.copy(a) ; b[0,0]=3 ; b[1,1]=5
>> a
[[ 1. 0. 0.]
 [ 0. 0. 2.]]
>> b
[[ 3. 0. 0.]
 [ 0. 5. 2.]]
```

FIN

CONCOURS COMMUNS POLYTECHNIQUES

RAPPORT SUR L'EPREUVE ECRITE 2017 DE MODELISATION DE SYSTEMES PHYSIQUE OU CHIMIQUE LA FILIERE PC

1/ CONSIGNES GENERALES :

A) Présentation du sujet

Le sujet proposait une résolution de l'équation de Poisson dans le cadre de l'électrostatique afin d'obtenir le potentiel de champ en tout point de l'espace, suivie de deux études de cas permettant de comparer des prévisions théoriques classiques avec celles de la simulation numérique.

Première partie

Après une partie introductive théorique permettant de souligner le rôle central en électrostatique de l'équation de Poisson, une alternance d'analyses numériques et physiques étaient conduites.
Une large part est donnée à l'implémentation numérique de l'équation de Poisson permettant le calcul du potentiel électrostatique aux différents points d'un maillage défini dans le domaine d'étude. Différentes méthodes de plus en plus performantes sont envisagées, tenant compte des problèmes liés aux calculs itératifs. La rapidité de la convergence est présentée, donnant l'occasion de discuter de la complexité du programme le plus performant. Enfin, le calcul du champ électrique à partir du potentiel calculé numériquement est évoqué, donnant l'occasion d'aborder les conditions aux limites.

Deuxième partie

Les résultats de la simulation numérique produite par les algorithmes de la première partie sont alors confrontés aux résultats théoriques (résultant de modélisations simplifiées) concernant deux cas d'école :

- le fil infini chargé uniformément pour lequel il est possible d'extraire une solution analytique facilement. Cet exemple permet en outre de mettre en avant aussi bien les limites de l'approche théorique que celles de l'approche numérique. L'étude se termine en proposant le cheminement inverse : retrouver, à partir d'un résultat numérique, le modèle physique initial.
- le mouvement d'un électron dans un tube d'oscilloscope, la déviation étant induite par le passage de la particule chargée entre les plaques d'un condensateur plan fini. L'étude théorique très classique de ce problème est suivie de son étude numérique : obtention de la carte de champ électrique, puis détermination de la trajectoire de l'électron à l'aide de la méthode d'Euler.

B) Prestation des candidats

Présentation

La présentation de nombreuses copies laisse à désirer sur un assez grand nombre de points :

- questions non ou mal numérotées, écriture peu soignée, code quasi-illisible et/ou non commenté,
- résultats non mis en évidence, applications numériques sans unités,
- absence de figures ou de schémas lorsqu'ils seraient souhaitables.

On ne peut que répéter que la présentation d'une copie est essentielle afin que le correcteur puisse juger en toute objectivité les réponses apportées aux questions posées.

Partie physique

En physique, de nombreuses questions étaient très proches du cours, permettant de mesurer la connaissance indispensable de celui-ci. D'autres étaient plus subtiles et ont permis de distinguer les candidats les plus brillants.
Les démonstrations sont trop souvent partielles, les résultats affirmés sans justification correcte.
Les questions d'interprétation (analyse des graphes et des figures) sont mal traitées.
Les résultats sont globalement décevants voire inquiétants, car de trop nombreux candidats ne maîtrisent pas des notions élémentaires :

- homogénéité des résultats,
- écriture de développements limités à l'ordre 1 ou 2,
- détermination de la surface de Gauss après détermination des symétries/invariances,
- utilisation des théorèmes énergétiques dans le cadre de la mécanique classique, expression de l'énergie potentielle électrique,
- détermination de la trajectoire d'une particule chargée dans un champ électrique.

Partie informatique

En informatique, de nombreux étudiants ne réutilisent pas les fonctions préalablement introduites et codées, perdant en clarté et gaspillant un temps précieux. Un problème d'informatique est souvent conçu de façon à ce que les questions intermédiaires introduisent des fonctions qui facilitent l'écriture de la solution.
Peu ont respecté la manière dont les bibliothèques étaient chargées par l'énoncé, en particulier lors de l'utilisation des fonctions mathématiques ou de la valeur de π.

2/ REMARQUES SPECIFIQUES :

Première partie : équation de Poisson

Q 1 : nom de ε_0 souvent approximatif, unités : quelquefois en $kg.m^{-3}$.
Q 2 : rarement deux exemples sont donnés.
Q 3 : le changement de variable est étonnamment très souvent mal effectué, conduisant à un résultat faux.
Q 4 : le développement limité à l'ordre 2 est non maîtrisé.
Q 6->13 : beaucoup de confusions entre les fonctions qui modifient des tableaux « en place » et celles qui renvoient des valeurs, non utilisation des fonctions définies précédemment, non respect de la valeur de retour demandée. Initialisation des tableaux, terminaisons des boucles souvent effectuées de manière partielle.
Q 14 : la question de complexité était très abordable, mais seul un nombre extrêmement réduit de candidats l'ont traitée et très peu l'ont fait correctement : la réponse était souvent O(N) ou $O(N^2)$, et le passage à N = 1000 quasiment toujours absent.
Q 15 : les bords du tableau sont rarement pris en compte correctement.

Deuxième partie : études de cas

Q 16, 17, 18 : questions très classiques. Beaucoup ne savent pas utiliser le théorème de Gauss, on a par exemple souvent une surface de Gauss sphérique alors qu'avant ils ont expliqué que les surfaces équipotentielles étaient des cylindres. Un cercle proposé comme surface équipotentielle est évidemment une réponse incorrecte. De trop nombreux candidats ne connaissent pas l'unité du champ électrique !
Q 19 : près de la moitié des candidats ne connaissent pas l'équation d'un cercle en coordonnées cartésiennes : ils ont souvent programmé la localisation de l'intérieur d'un carré mais pas d'un cercle !
Q 20,21 : rarement correctes, la notation `tab-f` n'a parfois pas été comprise (confusion entre un tableau global et l'argument d'une fonction).
Q 22 : les commentaires manquent de finesse en général, les documents et données à disposition n'étant pas examinés avec suffisamment de détails. Par exemple, le fait que le champ calculé au centre ne soit pas exactement nul a été très rarement commenté, tout comme la valeur plus élevée du champ au bord de l'enceinte.
Q 23 : l'idée de la répartition est souvent présentée, sans réelle justification. Les calculs de champs sont rarement faits.
Q 24 : la démonstration de l'expression est trop souvent approximative.
Q 25 : erreurs de signe, oubli d'unités répété.
Q 26 : conditions initiales parfois mal exploitées.
Q 27 : cette question est rarement traitée dans son intégralité : la détermination de l'équation de la tangente à une parabole semble être une question difficile !
Q 28 : cette question, évidente, est majoritairement ratée : la nullité de `rhos` échappe à beaucoup de candidats.
Q 30 : réponses très souvent incomplètes. Beaucoup ne jugent pas nécessaire d'écrire les développements limités permettant d'aboutir aux résultats et se contentent d'inventer la formule par « analogie ».
Q 31, 32, 33 : Ces trois questions, très abordables, auraient pu être correctement traitées par la quasi-totalité des candidats, ce qui n'a pas été le cas.
Q 35 : le tracé a très souvent été très mal reproduit sur la copie, et l'interprétation physique des résultats absente.

Conclusions

Toutes les parties du problème donnaient lieu à discussion /critiques et permettaient ainsi aux étudiants de mettre en avant les compétences et la démarche d'analyse scientifique qu'ils ont pu acquérir aux cours de deux ou trois années en CPGE.
La plupart des copies abordent toutes les parties, mais bien souvent avec une absence chronique de rigueur, surtout pénalisante dans les parties algorithmiques où l'on attend que les candidats écrivent le code dans un langage obéissant à une syntaxe spécifique, elle-même évaluée.
L'annexe sur Python, pourtant bien fournie, n'a sans doute pas été bien lue par certains candidats qui ont commis des erreurs simples comme ne pas utiliser la fonction `np.copy`.

Enfin, trop souvent, les questions ne sont pas lues de manière suffisamment attentive (notation choisie peu adaptée, écriture de fonctions qui ne renvoie pas les grandeurs demandées…). Ce sont autant de points qu'il est facile de ne pas perdre.

CONCOURS COMMUNS POLYTECHNIQUES

EPREUVE SPECIFIQUE - FILIERE PC

MODELISATION DE SYSTEMES PHYSIQUES OU CHIMIQUES

Jeudi 5 mai : 8 h - 12 h

N.B. : le candidat attachera la plus grande importance à la clarté, à la précision et à la concision de la rédaction. Si un candidat est amené à repérer ce qui peut lui sembler être une erreur d'énoncé, il le signalera sur sa copie et devra poursuivre sa composition en expliquant les raisons des initiatives qu'il a été amené à prendre.

Les calculatrices sont autorisées

Détermination du coefficient de transfert d'un polluant dans une colonne d'absorption

I. Présentation du problème

Les procédés d'absorption sont souvent utilisés pour la dépollution des gaz. Ces procédés reposent sur l'absorption préférentielle du gaz polluant par un solvant et sont la plupart du temps mis en œuvre dans des colonnes d'absorption (figure 1). Lors du contact entre les deux phases (gazeuse et liquide), le polluant est transféré du gaz vers le solvant. On récupère un gaz purifié en sortie haute de colonne et le solvant chargé du polluant en pied de colonne.

Un exemple courant est l'absorption du dioxyde de carbone présent dans les gaz de combustion à l'aide d'un solvant aminé pour éviter de rejeter ce gaz à effet de serre directement dans l'atmosphère.

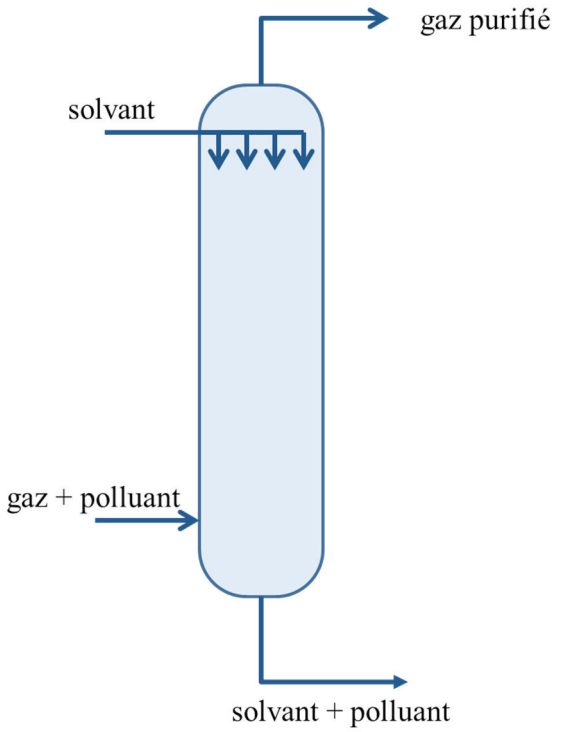


Figure 1 – Schéma d'une colonne d'absorption utilisée dans l'industrie pour la dépollution d'un gaz.

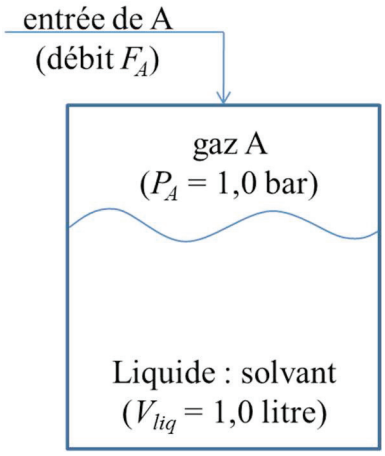
Pour dimensionner les colonnes d'absorption, on a besoin de connaître la conductance de transfert du polluant dans le solvant (notée k_L, unité : $m.s^{-1}$). Pour déterminer ce paramètre, on réalise une étude dans un réacteur de laboratoire dont les conditions de fonctionnement sont beaucoup plus simples et beaucoup mieux définies que dans une véritable colonne d'absorption. Le réacteur de laboratoire utilisé est un réacteur biphasique gaz-liquide.

Une des spécificités de ce réacteur est qu'il est alimenté par un débit variable de polluant qui est ajusté au cours du temps de manière à maintenir la pression de la phase gazeuse constante (figure 2, page 3). Au cours d'une expérience, on enregistre le débit molaire de polluant A qui entre dans le réacteur en fonction du temps. D'après la loi de Dalton, la pression totale est égale à la somme des pressions partielles des espèces dans la phase gazeuse. Comme A est seul dans la phase gazeuse, la pression totale P_{tot} est égale à la pression partielle de A, notée P_A. On suppose que le réacteur est isotherme.

La détermination directe de la conductance de transfert d'un polluant dans un solvant n'est pas aisée. On procède en deux étapes pour la déterminer :

- on détermine d'abord la valeur du produit $k_L \times a$, où a est l'aire interfaciale entre le liquide et le gaz par unité de volume de la phase liquide (unité de a : $m^2.m^{-3}$), grâce à une première expérience où le seul phénomène qui a lieu est l'absorption physique du polluant A dans le solvant ;

- on détermine ensuite les valeurs des deux paramètres k_L et a de manière indépendante grâce à une seconde expérience avec le même solvant mais qui contient cette fois-ci un réactif B qui réagit avec le gaz polluant A (absorption physique avec réaction chimique). On supposera que la réaction chimique en phase liquide s'écrit : $A + B \rightarrow produits$.



F_A : débit molaire de A qui entre dans le réacteur.

P_A : pression partielle de A dans la phase gazeuse.

V_{liq} : volume de la phase liquide (solvant).

Figure 2 – Schéma du réacteur de laboratoire utilisé pour la détermination de la conductance de transfert du polluant A.

On enregistre le débit molaire de A qui entre dans le réacteur en fonction du temps pour les deux expériences (sans réaction et avec réaction avec le réactif B). Les données se trouvent dans un fichier nommé « `data.txt` ». La première colonne correspond au temps (s). Les deuxième et troisième colonnes correspondent aux débits molaires de A enregistrés au cours des deux expériences (sans et avec réaction respectivement). Les colonnes sont séparées par des espaces (tableau 1, page 7).

II. Modélisation des phénomènes

Dans cette partie, on demande d'établir les modèles qui vont représenter l'évolution du débit molaire d'entrée F_A en fonction du temps. Les modèles sont obtenus en réalisant des bilans de matière.

II.1 Cas où le solvant ne contient pas le réactif B

Q1.1) On considère tout d'abord l'absorption du polluant A dans le solvant (cas où le solvant ne contient pas le réactif B). Le phénomène qui régit le transfert de A à l'interface entre les deux phases est la diffusion, phénomène localisé au niveau de la couche limite située à proximité de l'interface (zone délimitée par des pointillés sur la figure 3, page 4).

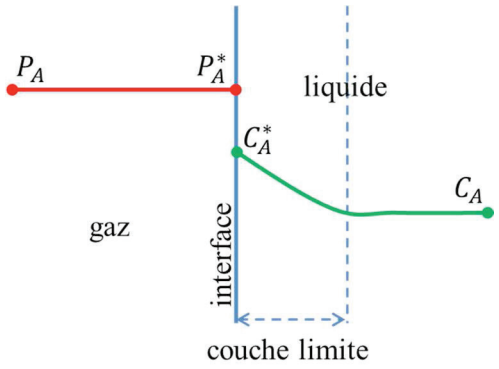


Figure 3 – Schéma représentant l'interface gaz – liquide.

La pression à l'interface P_A^* est égale à la pression partielle P_A dans la phase gazeuse car le polluant A est seul dans cette phase. La pression dans la phase gazeuse est maintenue constante ($P_A = P_A^* = 1{,}0$ bar). A l'interface, la loi de Henry permet de relier la pression P_A^* et la concentration C_A^*. D'après cette loi, la concentration de gaz dissous dans un liquide est proportionnelle à la pression partielle du gaz en contact avec le liquide à l'équilibre thermodynamique et à température constante.

Calculer la valeur de la concentration C_A^* en mol.m^{-3} à partir de la loi de Henry. On donne la constante de Henry pour A : $H = 26{,}0$ bar.L.mol^{-1}. On pourra s'appuyer sur l'analyse dimensionnelle pour écrire la loi de Henry.

Q1.2) Le débit molaire F_A^* qui traverse l'interface gaz-liquide vers le solvant s'écrit : $k_L \times a \times (C_A^* - C_A) \times V_{liq}$, où C_A^* est la concentration du polluant à l'interface, C_A est la concentration de A dans le solvant (figure 3) et V_{liq} le volume de la phase liquide.

Vérifier que l'expression $k_L \times a \times (C_A^* - C_A) \times V_{liq}$ est bien homogène à un débit molaire.

Q1.3) Sachant qu'à l'instant initial la concentration C_A de A dans la phase liquide est nulle, expliquer qualitativement comment C_A va évoluer au cours du temps. Préciser le signe du terme $k_L \times a \times (C_A^* - C_A) \times V_{liq}$.

Q1.4) On considère que la phase gazeuse se comporte comme un réacteur ouvert parfaitement agité en régime permanent (il n'y a donc pas d'accumulation de A dans cette phase) dans lequel il n'y a pas de réaction (le gaz ne fait que transiter dans cette partie du réacteur). On précise que les grandeurs intensives (température, pression, concentration) dans un réacteur ouvert parfaitement agité sont identiques en tout point.

Ecrire le bilan de matière sur le polluant A dans la phase gazeuse. Préciser le terme d'entrée et le terme de sortie intervenant dans le bilan de matière.

Q1.5) La phase liquide est modélisée par un réacteur semi-fermé parfaitement agité fonctionnant en régime transitoire. Ce type de réacteur peut être considéré comme un réacteur fermé parfaitement agité (contenant initialement le solvant dans le cas présent) possédant une entrée permettant l'ajout d'une espèce au cours du temps (le polluant A dans le cas présent).

Pour simplifier le problème, on suppose que le volume de la couche limite (zone à proximité de l'interface dans laquelle on observe un gradient de concentration du polluant dû à sa diffusion de l'interface vers la phase liquide, figure 3, page 4) est très faible et que la concentration de A dans la phase liquide est homogène. En d'autres termes, on considère que la concentration de A passe instantanément de C_A^*, à l'interface, à C_A dans la phase liquide comme indiqué sur le schéma de la figure 4.

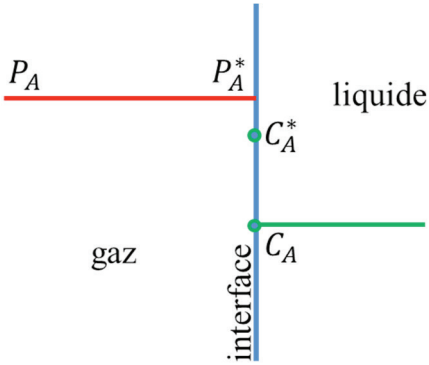


Figure 4 – Schéma simplifié de l'interface gaz–liquide.

Ecrire le bilan de matière sur le polluant A dans la phase liquide. Préciser le terme d'entrée et le terme d'accumulation.

Q1.6) Simplifier le bilan en considérant que le volume de la phase liquide reste constant au cours du temps.

Q1.7) Résoudre l'équation différentielle obtenue à la question précédente en considérant que la concentration initiale de A dans la phase liquide est nulle. On pourra effectuer un changement de variable.

Q1.8) Donner l'expression de la concentration de A en fonction du temps. Vérifier que C_A évolue bien de la manière prévue à la question **Q1.3)**, page 4.

Q1.9) Etablir l'expression du débit molaire F_A en fonction du temps. Vérifier l'homogénéité de la relation.

Q1.10) Préciser vers quelle valeur tend le débit molaire F_A pour des temps importants et indiquer vers quelle valeur tend la concentration C_A dans ce cas.

II.2 Cas où le réactif B est présent dans le solvant

Q2.1) On considère maintenant le cas où le réactif B est présent dans le solvant (à la concentration initiale C_{B0} de 1 000,0 mol.m^{-3}) et où il y a réaction entre A et B dans la phase liquide. Pour simplifier, on suppose que le réactif B ne passe pas dans la phase gazeuse (figure 5, page 6). On suppose également que la phase liquide est parfaitement agitée et que le volume reste constant. On précise que, comme dans le cas précédent, la concentration C_A de A dans la phase liquide est nulle à l'instant initial.

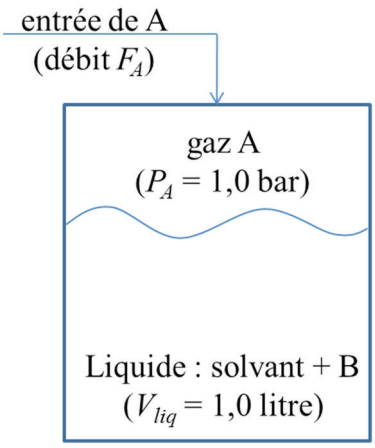


Figure 5 – Schéma du réacteur dans le cas de la deuxième expérience.

Le bilan de matière sur la phase gazeuse est-il modifié ? Justifier.

Q2.2) Des études préalables ont montré que la vitesse apparente de la réaction dans la phase liquide (notée r_{app}) est impactée par les phénomènes de diffusion de A dans la couche limite et qu'elle peut s'écrire sous la forme $r_{app} = a \times C_A^* \times \sqrt{D_A \times k \times C_B}$ pour une réaction d'ordre un par rapport à chacun des deux réactifs. D_A est la diffusivité de A dans la phase liquide ($1{,}83 \times 10^{-9}$ m^2.s^{-1}), k est la constante cinétique vraie de la réaction entre A et B (k = 0,10 m^3.mol^{-1}.s^{-1}) et C_B est la concentration de B dans la phase liquide.

Indiquer la dimension de r_{app}. Préciser son unité dans le système SI.

Q2.3) Ecrire le nouveau bilan de matière sur A dans la phase liquide en prenant en compte le nouveau terme correspondant à la réaction chimique (on fera particulièrement attention au signe).

Q2.4) Simplifier le bilan en considérant que le volume de la phase liquide reste constant.

Q2.5) Ecrire le bilan de matière sur B dans la phase liquide en considérant que la phase liquide est un réacteur fermé parfaitement agité.

Q2.6) Simplifier ce bilan en considérant que le volume de la phase liquide reste constant.

Q2.7) Donner le système de deux équations différentielles ordinaires qui régit l'évolution des concentrations de A et de B dans la phase liquide en fonction du temps.

III. Traitement numérique des données expérimentales

Dans cette partie, on propose de comparer les modèles établis dans la partie II aux résultats expérimentaux pour déterminer la valeur de l'aire interfaciale a et celle du coefficient de transfert k_L en utilisant des outils numériques.

Les portions de programme demandées au candidat peuvent être réalisées dans le langage Python ou dans le langage Scilab. Cependant, toutes les questions seront traitées dans le même langage. On veillera à apporter les commentaires suffisants à la compréhension du programme et utiliser des noms de variables explicites. Il est demandé de répondre précisément aux questions posées (par exemple, on écrira une fonction uniquement lorsque cela est explicitement demandé). Des annexes sont disponibles à la fin de l'énoncé.

III.1 Détermination de la valeur du produit $k_L \times a$

Dans cette partie, on va déterminer la valeur du produit $k_L \times a$ à partir des données expérimentales enregistrées en l'absence de réaction dans la phase liquide. Si l'expression du débit molaire de A demandée à la question **Q1.9)**, page 5, n'a pas été trouvée, on utilisera la notation formelle suivante pour écrire le code : $F_A = f(t, (k_L \times a))$.

Q3.1) Les données expérimentales sont disponibles sous forme d'un fichier texte nommé « `data.txt` » dans lequel les informations sont présentées sous forme de colonnes (tableau 1). La première colonne du fichier correspond au temps (unité : s), la deuxième au débit molaire de A (unité : mol.s^{-1}) entrant dans le réacteur enregistré au cours de la première expérience (cas de l'absorption dans le solvant sans réaction) et la troisième colonne au débit molaire de A (unité : mol.s^{-1}) entrant dans le réacteur enregistré au cours de la seconde expérience (en présence du réactif B).

```
00.0 1.9231E-04 1.9230E-04
10.0 1.8293E-04 1.9108E-04
20.0 1.7401E-04 1.8991E-04
30.0 1.6552E-04 1.8878E-04
40.0 1.5745E-04 1.8771E-04
50.0 1.4977E-04 1.8667E-04
```

Tableau 1 – Données expérimentales contenues dans le fichier « `data.txt` ».

Q3.1.a) Indiquer la syntaxe à utiliser pour charger le fichier « `data.txt` » qui contient les données expérimentales. Donner le code permettant de créer trois vecteurs t^{exp}, F_A^{exp1} et F_A^{exp2} correspondant respectivement aux données des première, deuxième et troisième colonnes. Donner également le code qui permet de déterminer le nombre de points expérimentaux n.

Q3.1.b) Donner la syntaxe permettant de tracer sur un graphe l'évolution du débit molaire de A en fonction du temps dans le cas où seul le phénomène d'absorption physique est étudié.

Q3.2) On souhaite déterminer la valeur du produit $k_L \times a$ grâce au modèle établi à la question **Q1.9)**, page 5, par une méthode d'optimisation numérique de paramètres. La méthode utilisée est celle des moindres carrés. Il s'agit d'une méthode qui permet de comparer des données expérimentales à un modèle mathématique la plupart du temps issu d'une théorie.

Dans le cas présent, le modèle mathématique correspond à l'expression du débit molaire F_A, que l'on notera F_A^{th1} par la suite, déterminé à la question **Q1.9)**, page 5. Ce débit molaire est une fonction du temps t et du produit $k_L \times a$ que l'on souhaite déterminer. La méthode des moindres carrés donne la valeur optimale du produit $k_L \times a$ qui permet de calculer un débit molaire théorique F_A^{th1} représentant le mieux le débit molaire expérimental F_A^{exp1}. Cette valeur optimale est celle qui permet de minimiser la somme quadratique des déviations des mesures aux prédictions. Cette somme, notée $S(k_L \times a)$, peut se mettre sous la forme suivante :

$$S(k_L \times a) = \sum_{i=1}^{n} \left(F_{Ai}^{exp1} - F_{Ai}^{th1}(k_L \times a) \right)^2$$

avec n le nombre de points expérimentaux, i l'indice correspondant au point expérimental enregistré au temps t_i^{exp} $(1 \leq i \leq n)$, F_{Ai}^{exp1} le débit molaire expérimental obtenu au temps t_i^{exp} et $F_{Ai}^{th1}(k_L \times a)$ le débit molaire théorique calculé au temps t_i^{exp} grâce au modèle établi à la question **Q1.9)**, page 5.

Ecrire une fonction `smc(kla,t_exp,Fa_exp1)` qui retourne la valeur de la quantité S. Cette fonction aura comme argument d'entrée le produit $k_L \times a$ et les vecteurs t^{exp} et F_A^{exp1} créés à la question **Q3.1.a)**, page 7.

Q3.3) Une des méthodes utilisées pour trouver le minimum d'une fonction f est celle proposée par Nelder-Mead. Il s'agit d'un algorithme d'optimisation non linéaire basé sur le concept de simplexe à $n+1$ sommets dans un espace à n dimensions. Dans le cas d'une fonction d'une variable $(n = 1)$, un simplexe est un segment.

L'algorithme de Nelder-Mead est une procédure itérative. Partant d'un segment initial $[MN]$, l'algorithme va générer une succession de segments par des transformations simples au cours des itérations : le segment se déplace et se réduit jusqu'à ce que ses extrémités se rapprochent d'un point où la fonction présente un minimum (ce minimum peut être un minimum local).

Soient x_M et x_N les abscisses des points M et N. Les transformations subies par le segment (illustrées à la figure 6, page 9) sont basées sur la comparaison des valeurs de la fonction f aux extrémités du segment.

- La première étape consiste à réindexer si nécessaire les deux extrémités du segment de manière à ce que $f(x_M) \leq f(x_N)$.
- L'extrémité N pour laquelle la fonction f est maximale est remplacée par une nouvelle extrémité. On introduit alors le point R, réflexion de N par rapport à M, tel que $x_R = x_M + (x_M - x_N)$.
- Si $f(x_R) < f(x_M)$, le segment est étiré dans cette direction (car la réflexion se rapproche du minimum). On introduit un point E (étirement du segment) tel que $x_E = x_M + 2(x_M - x_N)$ qui permet éventuellement de se rapprocher encore un peu plus du minimum. Le point N est substitué par le point R si $f(x_R) < f(x_E)$, sinon par E.
- Si $f(x_R) > f(x_M)$, le segment est réduit dans la direction opposée de la réflexion (car la réflexion s'éloigne du minimum). On introduit le point C_1 (contraction du segment) qui est défini par $x_{C_1} = x_N + 1/2\,(x_M - x_N)$. Si $f(x_{C_1}) < f(x_M)$, N est remplacé par C_1. Sinon N est remplacé par C_2, homothétie de rapport -1 et de centre M du point C_1 $(x_{C_2} = x_M + 1/2\,(x_M - x_N))$.

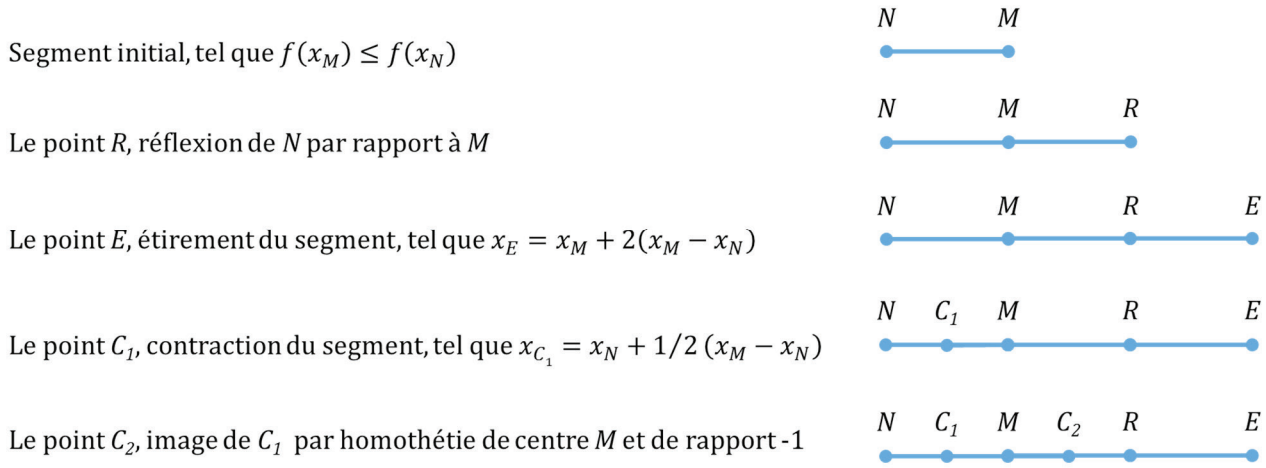


Figure 6 – Illustration des transformations subies par le segment initial $[MN]$ dans la méthode d'optimisation de Nelder-Mead.

Q3.3.a) Soit $f(x)$ une fonction dont on cherche le minimum. Soient x_M et x_N les abscisses des extrémités d'un segment $[MN]$. Ecrire le code permettant de réindexer les deux extrémités du segment de manière à ce que $f(x_M)$ soit inférieure ou égale à $f(x_N)$.

Q3.3.b) Partant du segment $[MN]$ réindexé, écrire le code permettant de transformer le segment $[MN]$ au cours d'une itération de la méthode de Nelder-Mead. Chaque étape du code devra être commentée.

Q3.3.c) Construire une boucle permettant de réaliser des itérations successives de la méthode de Nelder-Mead. On prendra $x_M^0 = 1$ et $x_N^0 = 10^{-5}$ pour les abscisses des deux extrémités du segment initial $[MN]$. Cette boucle sera interrompue lorsque l'écart relatif $abs\left(\frac{x_M - x_N}{x_M}\right)$ sera inférieur à 10^{-4} ou lorsque le nombre maximum d'itérations, fixé par l'utilisateur, sera atteint soit $Itmax = 1\,000$.

Q3.4) La méthode de Nelder-Mead appliquée à la fonction $S(k_L \times a)$ conduit à $(k_L \times a)_{opt} = 5{,}0 \times 10^{-3}$. Donner le code permettant de calculer le débit molaire théorique grâce à la valeur optimisée du produit $(k_L \times a)_{opt}$.

Q3.5) Indiquer la syntaxe à utiliser pour comparer sur un même graphe les débits molaires théorique et expérimental. Expliquer l'intérêt de comparer la courbe théorique et les points expérimentaux.

Q3.6) On souhaite développer un algorithme de tri permettant de réindexer les sommets d'un simplexe à $p + 1$ sommets ($X = \{x_1, x_2, \ldots x_{p+1}\}$) dans le cadre de la généralisation de la méthode de Nelder-Mead appliquée à une fonction quelconque $z(x)$ de p variables ($x \in \mathbb{R}^p$). On propose d'utiliser la méthode du tri par sélection pour réindexer les sommets du simplexe de sorte que les valeurs que prend la fonction $z(x)$ en chaque sommet du simplexe soient classées de manière croissante.

La méthode du tri par sélection repose sur la recherche du plus petit élément d'une liste. Une fois identifié, on échange cet élément avec le premier élément de la liste. Il s'agit d'une procédure itérative (figure 7). Lors de la première itération, on échange le premier élément ($i = 1$) avec le plus petit de la liste. Lors de la i^e itération, on échange le i^e élément par le plus petit élément en ne considérant que ceux à partir de la i^e position.

Vecteur initial	10	2	- 3	5	- 1	20	60	48	- 15	9
1re itération	**- 15**	2	- 3	5	- 1	20	60	48	**10**	9
2e itération	*- 15*	**- 3**	**2**	5	- 1	20	60	48	10	9
3e itération	*- 15*	*- 3*	**- 1**	5	**2**	20	60	48	10	9

Figure 7 – Représentation schématique des trois premières itérations de la méthode de tri par sélection.

Ecrire le code permettant de réindexer les sommets du simplexe $X = \{x_1, x_2, \ldots x_{p+1}\}$ de sorte que les valeurs que prend la fonction $z(x)$ en chaque sommet du simplexe soient classées de manière croissante. On supposera que la fonction $z(x)$ est déjà définie.

III.2 Détermination des valeurs de a et k_L

Dans cette partie, on souhaite déterminer la valeur de l'aire interfaciale a en comparant les débits molaires expérimentaux obtenus lors de la deuxième expérience (en présence du réactif B dans le solvant) avec le débit molaire théorique calculé à partir du modèle établi à la question **Q2.7)**, page 6. Ce modèle est un système de deux équations différentielles ordinaires que l'on peut écrire sous la forme suivante :

$$\begin{cases} \dfrac{dC_A}{dt} = g\big(t, C_A, C_B, a, (k_L \times a)\big) \\ \dfrac{dC_B}{dt} = h(t, C_A, C_B, a) \end{cases}$$

La fonction g est une fonction du produit $k_L \times a$ dont la valeur a été déterminée grâce à la première expérience ($(k_L \times a)_{opt} = 5{,}0 \times 10^{-3}$ s^{-1}), du temps t, des concentrations C_A et C_B de A et B dans le solvant et de l'aire interfaciale a. La fonction h est une fonction du temps t, des concentrations C_A et C_B de A et B dans le solvant et de l'aire interfaciale a.

On propose d'utiliser la méthode d'Euler pour résoudre le système d'équations différentielles.

Q3.7) Ecrire deux fonctions `G(t,CA,CB,A,kla)` et `H(t,CA,CB,A)` permettant de calculer les valeurs des fonctions $g\big(t, C_A, C_B, a, (k_L \times a)\big)$ et $h(t, C_A, C_B, a)$.

Q3.8) La méthode utilisée pour résoudre le système d'équations différentielles est la méthode d'Euler à pas fixe. On note t_0 le temps initial, t_f le temps final d'intégration et Δt le pas d'intégration.

Q3.8.a) Soient n le nombre d'expériences réalisées et Δt^{exp} l'intervalle de temps entre deux mesures expérimentales successives du débit molaire du polluant A. Pour réaliser la comparaison entre les débits molaires théoriques et expérimentaux, on ne s'intéresse qu'aux valeurs du débit molaire théorique F_{Ai}^{th2} aux instants particuliers $t_i^{exp} = \Delta t^{exp} \times i$ avec i variant de 1 à n.

La méthode d'Euler ne permet d'obtenir un résultat correct que si le pas d'intégration, Δt, est suffisamment faible. Par conséquent, on souhaite utiliser un pas d'intégration Δt cent fois plus petit que l'intervalle de temps entre deux mesures expérimentales Δt^{exp}. Le temps d'intégration est donc discrétisé en intervalles de durée Δt et les valeurs des concentrations C_A et C_B sont calculées aux instants particuliers $t_j = \Delta t \times j$ avec j variant de 0 à m.

Donner la syntaxe permettant de calculer le pas de temps Δt ainsi que le nombre d'intervalles m (m est un nombre entier) en fonction de t_0 (le premier élément du vecteur t^{exp}), t_f (le dernier élément du vecteur t^{exp}) et Δt^{exp} (l'intervalle de temps entre deux mesures expérimentales successives du débit molaire).

Q3.8.b) Donner une expression de $C_A(t + \Delta t)$ à l'ordre 1 ($o(\Delta t)$) en fonction de C_A et $\frac{dC_A}{dt}$ évaluées en t à l'aide d'un développement limité de la fonction $t \mapsto C_A(t)$.

Q3.8.c) En déduire une valeur approchée de $\left.\frac{dC_A}{dt}\right|_t$ à l'ordre 0 ($o(1)$) en fonction de $C_A(t)$, $C_A(t + \Delta t)$ et Δt. Faire de même pour donner une valeur approchée de $\left.\frac{dC_B}{dt}\right|_t$ à l'ordre 0 ($o(1)$) en fonction de $C_B(t)$, $C_B(t + \Delta t)$ et Δt.

Q3.8.d) On note C_{Aj} et C_{Bj} les concentrations $C_A(t_j)$ et $C_B(t_j)$. De même, on note C_{Aj+1} et C_{Bj+1} les concentrations $C_A(t_{j+1})$ et $C_B(t_{j+1})$. Donner une expression de la dérivée par rapport au temps de C_A évaluée à l'instant t_j, notée $\left.\frac{dC_A}{dt}\right|_{t_j}$, en fonction de Δt, C_{Aj} et C_{Aj+1}. De même, donner une expression de la dérivée par rapport au temps de C_B évaluée à l'instant t_j, notée $\left.\frac{dC_B}{dt}\right|_{t_j}$, en fonction de Δt, C_{Bj} et C_{Bj+1}.

Q3.8.e) Donner le système de deux équations permettant de calculer C_{Aj+1} et C_{Bj+1} en fonction de $C_{Aj}, C_{Bj}, \Delta t, g\left(t_j, C_{Aj}, C_{Bj}, a, (k_L \times a)\right)$ et $h(t_j, C_{Aj}, C_{Bj}, a)$.

Q3.8.f) Ecrire une fonction `Euler(G,H,t0,tf,CA0,CB0,Dt,m,a,kla)` qui retourne deux vecteurs : le vecteur temps et le vecteur correspondant à la concentration C_A calculée par la méthode d'Euler.

Q3.9) On souhaite utiliser la méthode des moindres carrés pour déterminer la valeur optimale de l'aire interfaciale a en utilisant une stratégie similaire à celle utilisée pour obtenir la valeur optimisée du produit $(k_L \times a)$. Pour cela, on a besoin de connaître les valeurs du débit molaire de A F_{Ai}^{th2} calculé aux instants particuliers $t_i^{exp} = \Delta t^{exp} \times i$ avec i variant de 1 à n.

Donner le code permettant de créer un vecteur F_A^{th2} contenant les valeurs du débit molaire de A calculé aux instants particuliers t_i^{exp} à partir des valeurs de la concentration de A (C_A^{th2}) calculées en utilisant la fonction `Euler` de la question **Q3.8.f)**.

Q3.10) La valeur optimale de l'aire interfaciale a est celle qui permet de minimiser la fonction $S_2(a)$ qui peut se mettre sous la forme suivante :

$$S_2(a) = \sum_{i=1}^{n} \left(F_{Ai}^{exp2} - F_{Ai}^{th2}(a) \right)^2$$

avec F_{Ai}^{exp2} le débit molaire expérimental obtenu au temps t_i^{exp}, $F_{Ai}^{th2}(a)$ le débit molaire théorique calculé au temps t_i^{exp} (question **Q3.9)**, page 11) et n le nombre de points expérimentaux.

Donner le code permettant de construire une fonction `smc2(a,t_exp,Fa_exp2)` qui retourne la valeur de la quantité S_2. Cette fonction aura comme argument d'entrée l'aire interfaciale a et les vecteurs t^{exp} et F_A^{exp2} créés à la question **Q3.1.a)**, page 7.

Q3.11) La méthode de Nelder-Mead appliquée à la fonction $S_2(a)$ conduit à a_{opt} = 10,0 $m^2.m^{-3}$. Indiquer le code pour calculer le coefficient de transfert k_L et afficher les valeurs de a_{opt} et k_L à l'écran.

III.3 Utilisation du modèle pour réaliser des simulations

Maintenant que l'on connaît les valeurs des paramètres a et k_L, on souhaite utiliser le modèle pour réaliser des prédictions. On propose d'étudier l'influence de la concentration initiale C_{B0} du réactif B dans la phase liquide sur le débit molaire en entrée du réacteur (F_A^{th2}).

Q3.12) Expliquer qualitativement comment doit varier le débit molaire en entrée du réacteur avec la concentration initiale du réactif B.

Q3.13) Justifier le concept « d'accélération du transfert par la réaction chimique ».

Q3.14) Ecrire le code qui permettrait de réaliser une prédiction du débit molaire F_A^{th} à l'aide du modèle et des paramètres optimisés (k_L et a) en prenant C_{B0} comme valeur pour la concentration initiale du réactif B, une valeur rentrée par l'utilisateur (unité : $mol.m^{-3}$). On prendra le même pas Δt que précédemment et comme durée d'intégration 1 000 s.

Q3.15) On souhaite calculer la quantité de matière de polluant A, n_A, à partir du débit molaire F_A^{th} obtenu à la question précédente par intégration sur l'intervalle 0 – 1 000 s. La méthode utilisée pour réaliser l'intégration est la méthode des trapèzes. Donner le code permettant de calculer n_A par cette méthode.

ANNEXE A : COMMANDES ET FONCTIONS USUELLES DE SCILAB

A=[a b c d;e f g h;i j k l]

Description : commande permettant de créer une matrice dont la première ligne contient les éléments a, b, c, d, la seconde ligne contient les éléments e, f, g, h et la troisième, les éléments i, j, k, l.

Exemple : A=[1 2 3 4 5;3 10 11 12 20;0 1 0 0 2]

⇨ 1. 2. 3. 4. 5.
3. 10. 11. 12. 20.
0. 1. 0. 0. 2.

A(i,j)

Description : fonction qui retourne l'élément (i, j) de la matrice A. Pour accéder à l'intégralité de la ligne i de la matrice A, on écrit $A(i,:)$. De même, pour obtenir toute la colonne j de la matrice A, on utilise la syntaxe $A(:,j)$.

Arguments d'entrée : les coordonnées de l'élément dans la matrice A.

Argument de sortie : l'élément (i, j) de la matrice A.

Exemple : A=[1 2 3 4 5;3 10 11 12 20;0 1 0 0 2]
A(2,4)
⇨ 12
A(2,:)
⇨ 3. 10. 11. 12. 20.
A(:,3)
⇨ 3.
11.
0.

x=[x1:Dx:x2]

Description : commande permettant de créer un vecteur dont les éléments sont espacés de Dx et dont le premier élément est x_1 et le dernier élément est le plus grand multiple de Dx inférieur ou égal à x_2.

ATTENTION : le vecteur ainsi créé est un vecteur ligne. Pour convertir un vecteur ligne en un vecteur colonne, on le transpose en utilisant l'apostrophe « ' » : x_trans=x'.

Exemple *:* x=[2:0.5:4.2]

⇨ 2. 2.5 3. 3.5 4.
x_trans=x'
⇨ 2.
2.5
3.
3.5
4.

zeros(n,m)

Description : fonction créant une matrice (tableau) de dimensions $n \times m$ dont tous les éléments sont nuls.

Arguments d'entrée : deux entiers n et m correspondant aux dimensions de la matrice à créer.

Argument de sortie : un tableau (matrice) d'éléments nuls.

Exemple : zeros(3,4)

⇨ 0. 0. 0. 0.
0. 0. 0. 0.
0. 0. 0. 0.

plot(x,y)

Description : fonction permettant de tracer sur un graphique n points dont les abscisses sont contenues dans le vecteur x et les ordonnées dans le vecteur y.
Arguments d'entrée : un vecteur d'abscisses x (tableau de dimension n) et un vecteur d'ordonnées y (tableau de dimension n).
Exemple :
```
x= [3:0.1:5]
y=sin(x)
plot(x,y)
```

read("nom_fichier",m,n)

Description : fonction permettant de lire les données sous forme de matrice dans un fichier texte et de les stocker dans une matrice.
Arguments d'entrée : un nom de fichier contenant des données sous forme de matrice de dimension (m,n) et les dimensions de la matrice m (nombre de lignes) et n (nombre de colonnes). On prend m = -1 si le nombre de lignes n'est pas connu a priori.
Exemple :
```
data= read("fichier.txt",−1,2)
//dans cet exemple, data est une matrice constituée des deux premières
//colonnes se trouvant dans le fichier nommé fichier.txt.
```

sum(x)

Description : fonction permettant de faire la somme des éléments d'un vecteur ou tableau
Arguments d'entrée : un vecteur ou un tableau de réels, entiers ou complexes.
Argument de sortie : un scalaire y qui est la somme des éléments de x.
Exemple :
```
y= sum(x)
//y retourne la somme des éléments de x.
```

ANNEXE B : FONCTIONS DE PYTHON

B.1. BIBLIOTHEQUE NUMPY DE PYTHON

Dans les exemples ci-dessous, la bibliothèque numpy a préalablement été importée à l'aide de la commande : **import numpy as np**

On peut alors utiliser les fonctions de la bibliothèque, dont voici quelques exemples :

np.array(liste)

Description : fonction permettant de créer une matrice (de type tableau) à partir d'une liste.
Argument d'entrée : une liste définissant un tableau à 1 dimension (vecteur) ou 2 dimensions (matrice).
Argument de sortie : un tableau (matrice).
Exemple :
```
np.array([4,3,2])
⇨ [4  3  2]
np.array([[5],[7],[1]])
⇨ [[5]
   [7]
   [1]]
np.array([[3,4,10],[1,8,7]])
⇨ [[3 4 10]
   [1 8 7]]
```

$A[i,j]$.

Description : fonction qui retourne l'élément $(i+1, j+1)$ de la matrice A. Pour accéder à l'intégralité de la ligne i+1 de la matrice A, on écrit $A[i,:]$. De même, pour obtenir toute la colonne j+1 de la matrice A, on utilise la syntaxe $A[:,j]$.
Arguments d'entrée : une liste contenant les coordonnées de l'élément dans le tableau A.
Argument de sortie : l'élément $(\mathrm{i}+1, \mathrm{j}+1)$ de la matrice A.
ATTENTION : en langage Python, les lignes d'un tableau A de dimension $n \times m$ sont numérotées de 0 à $n-1$ et les colonnes sont numérotées de 0 à $m-1$

```
Exemple :   A=np.array([[3,4,10],[1,8,7]])
            A[0,2]
            ⇨ 10
            A[1,:]
            ⇨ [1  8  7]
            A[:,2]
            ⇨ [10  7]
```

np.zeros((n,m))

Description : fonction créant une matrice (tableau) de dimensions $n \times m$ dont tous les éléments sont nuls.
Arguments d'entrée : un tuple de deux entiers correspondant aux dimensions de la matrice à créer.
Argument de sortie : un tableau (matrice) d'éléments nuls.

```
Exemple :   np.zeros((3,4))
            ⇨ [[0 0 0 0]
               [0 0 0 0]
               [0 0 0 0]]
```

np.linspace(Min,Max,nbElements)

Description : fonction créant un vecteur (tableau) de $nbElements$ nombres espacés régulièrement entre Min et Max. Le 1er élément est égal à Min, le dernier est égal à Max et les éléments sont espacés de $(Max - Min)/(nbElements - 1)$:
Arguments d'entrée : un tuple de 3 entiers.
Argument de sortie : un tableau (vecteur).

```
Exemple :   np.linspace(3,25,5)
            ⇨ [3  8.5  14  19.5  25]
```

np.loadtxt('nom_fichier',delimiter='string',usecols=[n])

Description : fonction permettant de lire les données sous forme de matrice dans un fichier texte et de les stocker sous forme de vecteurs.
Arguments d'entrée : le nom du fichier qui contient les données à charger, le type de caractère utilisé dans ce fichier pour séparer les données (par exemple un espace ou une virgule) et le numéro de la colonne à charger (ATTENTION, la première colonne porte le numéro 0).
Argument de sortie : un tableau.

```
Exemple :   data=np.loadtxt('fichier.txt',delimiter=' ',usecols=[0])
            #dans cette exemple data est un vecteur qui correspond à la première
            #colonne de la matrice contenue dans le fichier fichier.txt
```

B.2. BIBLIOTHEQUE MATPLOTLIB.PYPLOT DE PYTHON

Cette bibliothèque permet de tracer des graphiques. Dans les exemples ci-dessous, la bibliothèque matplotlib.pyplot a préalablement été importée à l'aide de la commande :
import matplotlib.pyplot as plt

On peut alors utiliser les fonctions de la bibliothèque, dont voici quelques exemples :

plt.plot(x,y)

Description : fonction permettant de tracer un graphique de n points dont les abscisses sont contenues dans le vecteur x et les ordonnées dans le vecteur y. Cette fonction doit être suivie de la fonction **plt.show()** pour que le graphique soit affiché.
Arguments d'entrée : un vecteur d'abscisses x (tableau de dimension n) et un vecteur d'ordonnées y (tableau de dimension n).
Argument de sortie : un graphique.
Exemple :

```
x= np.linspace(3,25,5)
y=sin(x)
plt.plot(x,y)
plt.title('titre_graphique')
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```

plt.title('titre')

Description : fonction permettant d'afficher le titre d'un graphique.
Argument d'entrée : une chaîne de caractères.

plt.xlabel('nom')

Description : fonction permettant d'afficher le contenu de nom en abscisse d'un graphique.
Argument d'entrée : une chaîne de caractères.

plt.ylabel('nom')

Description : fonction permettant d'afficher le contenu de nom en ordonnée d'un graphique.
Argument d'entrée : une chaîne de caractères.

plt.show()

Description : fonction réalisant l'affichage d'un graphe préalablement créé par la commande **plt.plot(x,y)**. Elle doit être appelée après la fonction plt.plot et après les fonctions plt.xlabel et plt.ylabel.

B.3. FONCTION INTRINSEQUE DE PYTHON

sum(x)

Description : fonction permettant de faire la somme des éléments d'un vecteur ou tableau.
Arguments d'entrée : un vecteur ou un tableau de réel, entier ou complexe.
Argument de sortie : un scalaire y qui est la somme des éléments de x.
Exemple :

```
y= sum(x)
 //y retourne la somme des éléments de x.
```

FIN

CONCOURS COMMUNS POLYTECHNIQUES

RAPPORT SUR L'EPREUVE ECRITE 2016 DE MODELISATION DE SYSTEMES PHYSIQUE OU CHIMIQUE DE LA FILIERE PC

1/ REMARQUES GENERALES

a) Présentation du sujet

Le sujet portait sur la détermination du coefficient de transfert d'un polluant dans une colonne d'absorption utilisée pour la dépollution des gaz.
La première partie donnait une description du problème posé : à savoir, la détermination de cette grandeur physico-chimique dans un réacteur de laboratoire avec deux phases (phase gazeuse et phase liquide) modélisées par des réacteurs idéaux simples et traités en cours (réacteurs ouvert et fermé). La détermination du coefficient de transfert d'un polluant se fait en deux étapes. Lors de la première, la phase liquide est uniquement constituée d'un solvant qui va absorber le polluant et l'exploitation des résultats (évolution du débit molaire de polluant absorbé en fonction du temps) permet de remonter à la valeur du produit du coefficient de transfert et de l'aire interfaciale. Lors de la deuxième étape, le solvant contient un réactif qui va réagir avec le polluant, ce qui a pour effet d'accélérer le transfert. Ceci permet de remonter à la valeur de l'aire interfaciale, puis à celle du coefficient de transfert du polluant.
La deuxième partie consistait en la mise en équation du problème par écriture de bilans de matière en réacteurs idéaux (réacteurs fermé et ouvert). En l'absence de réactif dans le solvant, on obtenait par intégration du bilan de matière l'expression théorique du débit molaire de polluant absorbé en fonction du temps. En présence du réactif, le modèle obtenu était un système de deux équations différentielles ordinaires.
La troisième partie portait sur le traitement numérique des données expérimentales (évolution du débit molaire de polluant absorbé en fonction du temps) à l'aide des bilans de matière établis dans la partie précédente pour déterminer les valeurs du produit du coefficient de transfert et de l'aire interfaciale. Dans un premier temps, la valeur du produit du coefficient de transfert et de l'aire interfaciale a été déterminée par une méthode d'optimisation numérique de paramètres (la méthode des moindres carrés) avec recherche du minimum de la somme quadratique des déviations des mesures aux prédictions grâce à l'algorithme de Nelder-Mead. Cette méthode faisait appel à un algorithme de tri. Dans un second temps, la valeur de l'aire interfaciale a été déterminée par une stratégie similaire (méthode des moindres carrés). Cette fois-ci, le modèle était un système de deux équations différentielles dont la résolution a été réalisée par la méthode d'Euler vue en cours. Dans un troisième temps, le modèle et les paramètres optimisés ont été utilisés pour réaliser des prédictions.

Le sujet était de difficulté moyenne et faisait appel à des notions transversales et complémentaires de chimie, de physique, de mathématiques et d'informatique. Les parties étaient rédigées de manière indépendante pour ne pas bloquer les candidats qui auraient pu être en difficultés sur l'une ou l'autre des parties.

Le niveau de difficulté des questions était varié, ce qui a permis de classer les candidats. La longueur du sujet était adéquate étant donné le nombre de candidats ayant pu aborder le sujet dans son intégralité.

Une annexe présentait les principales fonctions de Python et de Scilab utiles à la résolution de ce sujet, ce qui permettait d'aider les candidats ne se souvenant plus de la syntaxe exacte des fonctions à utiliser.

b) Prestation des candidats

Le langage informatique utilisé a été uniquement Python.

Les codes fournis dans les copies par les candidats étaient parfois difficiles à lire en raison de la présentation (mal écrit, code sur plusieurs pages, pas de couleur, pas de commentaires).

Les notations de l'énoncé ne sont pas toujours respectées, ce qui complique la correction.
Certains candidats font la confusion entre langage mathématique et langage informatique, voire mélangent les deux, ce qui complique la correction. Les indentations sont parfois oubliées.

Les consignes ne sont pas toujours respectées (emploi de fonction alors que cela n'est pas demandé et que ce n'est pas utile).

Certains candidats ont pris la liberté de donner un algorithme de tri qui était différent de celui demandé dans le sujet, ne répondant ainsi pas à la question posée.

Les dernières questions ont souvent été abordées dans l'esprit d'obtenir un maximum de points.

Les annexes ont été peu utilisées.

On peut classer les candidats en trois groupes :

- ceux qui ne sont ni à l'aise en physique/chimie, ni en informatique ;
- ceux qui se débrouillent en informatique mais qui ne maîtrisent pas la physique/chimie ;
- ceux qui ont les bases dans les deux domaines.

2/ REMARQUES SPECIFIQUES

- Partie II : Modélisation des phénomènes
  - Le passage d'une unité à l'autre (mol/L → mol/m^3) pose souvent problème.
  - Il y a une confusion entre dimension et unité.
  - Le bilan en réacteur ouvert en régime permanent n'est pas maîtrisé alors qu'il est conceptuellement plus simple que le bilan en réacteur fermé.
  - Beaucoup se lancent dans une démonstration complexe qui n'aboutit pas, bien que ce ne soit pas l'esprit de la question.
  - L'intégration de l'équation différentielle du premier ordre a parfois posé problème.
  - L'analyse dimensionnelle de la vitesse apparente a parfois posé problème.
  - Problème de signe pour le bilan sur B (accumulation forcément négative puisque B est consommé en réacteur fermé).
- Partie III.1 :
  - Confusion entre « read » du langage Scilab et « np.load.txt » du langage python.
  - Utilisation d'une boucle de 0 à len(t_exp) pour calculer len(t_exp).
  - Confusion entre « len » et « sum ».
  - Réindexation de f(xn) et f(xm) au lieu de xn et xm.
  - Syntaxe de réindexation fausse : oubli de la variable temporaire permettant de stocker une des valeurs pendant l'échange.
  - Il manque return à la fin des fonctions.
  - Utilisation de fonctions alors que ce n'était pas utile et pas demandé.
  - Confusion OR / AND dans la condition de la boucle WHILE.
  - Arguments peu probants pour expliquer l'utilité de la comparaison des courbes expérimentale et théorique (les algorithmes de recherche de minimum peuvent conduire à un minimum local).
- Partie III.2 :
  - Des erreurs sur les développements limités (non homogènes).
  - Les candidats repartent parfois de la formule discrétisée apprise par cœur pour redémontrer le développement limité de base.
  - Pour le calcul des concentrations par la méthode d'Euler, le calcul de CB est souvent manquant alors qu'il est nécessaire pour le calcul de CA.
  - Confusion entre « print » et « return ».
- Partie III.3 :
  - Les questions qualitatives ont souvent été bien traitées lorsque les candidats ont eu le temps.
  - La méthode des trapèzes, bien qu'elle figure au programme, a rarement été traitée et lorsqu'elle l'a été, avec plus ou moins de succès.

CONCOURS COMMUNS POLYTECHNIQUES

EPREUVE SPECIFIQUE - FILIERE PC

MODELISATION DE SYSTEMES PHYSIQUES OU CHIMIQUES

Durée : 4 heures

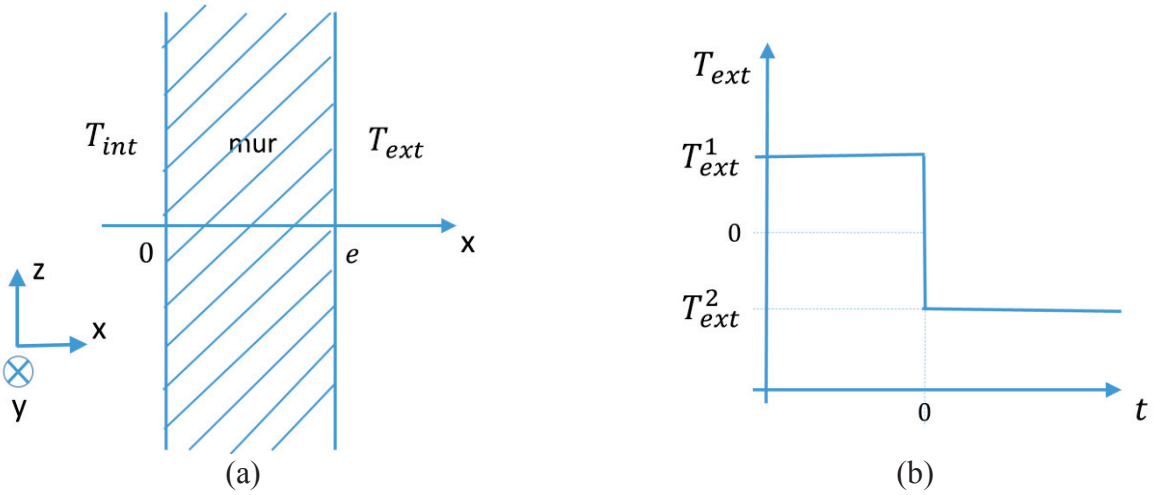
N.B. : le candidat attachera la plus grande importance à la clarté, à la précision et à la concision de la rédaction. Si un candidat est amené à repérer ce qui peut lui sembler être une erreur d'énoncé, il le signalera sur sa copie et devra poursuivre sa composition en expliquant les raisons des initiatives qu'il a été amené à prendre.

Les calculatrices sont interdites

Le sujet comporte deux parties indépendantes. Le candidat précisera au début de sa copie le langage de programmation (Python ou Scilab) qu'il a choisi et toutes les questions seront traitées dans le même langage. Un bonus sera accordé aux copies soignées avec des programmes bien commentés. Plusieurs fonctions du langage Scilab sont rappelées en annexe A. Les candidats choisissant le langage Python pourront utiliser les bibliothèques numpy et matplotlib.pyplot. Une documentation simplifiée de plusieurs fonctions de ces bibliothèques est présente en annexes B et C.

SIMULATION NUMERIQUE DU TRANSFERT THERMIQUE DANS UN MUR EN REGIME TRANSITOIRE

On étudie les transferts thermiques dans le mur d'une maison, figure 1(a). La température à l'intérieur de la maison est constante dans le temps et égale à $T_{int} = 20$ °C. Aux temps négatifs ($t < 0$), la température extérieure est égale à $T_{ext1} = 10$ °C. A $t = 0$, elle chute brusquement à $T_{ext2} = -10$ °C et elle reste égale à cette valeur aux temps positifs ($t > 0$), figure 1(b). On souhaite étudier l'évolution du profil de température dans le mur au cours du temps.



Figures 1 (a) - Schéma du mur étudié. (b) - Evolution de la température extérieure au cours du temps.

Le mur a une épaisseur $e = 40$ cm. Les propriétés physiques du mur sont constantes : conductivité thermique $\lambda = 1{,}65\ \mathrm{W.m^{-1}.K^{-1}}$, capacité thermique massique $c_p = 1\,000\ \mathrm{J.kg^{-1}.K^{-1}}$, masse volumique $\rho = 2\,150\ \mathrm{kg.m^{-3}}$.

PARTIE I : ETUDE PRELIMINAIRE

Dans cette partie, on établit l'équation gouvernant les variations de la température et on la résout en régime permanent.

I.A. Equation gouvernant la température

On suppose que la température dans le mur T ne dépend que du temps t et de la coordonnée x.

I.A.1. A quelle condition peut-on supposer que la température ne dépend pas des coordonnées y et z ?

I.A.2. Donner l'équation générale qui décrit le transport de chaleur dans un solide en l'absence de source d'énergie. Comment cette équation se simplifie-t-elle sous les hypothèses de la question I.A.1 ?

I.B. Conditions aux limites

On envisage plusieurs types de conditions aux limites.

(i) La température est imposée aux limites du système.

(ii) La paroi extérieure est isolée par un matériau de très faible conductivité.

I.B.1. Traduire chacune de ces conditions aux limites sur la fonction $T(x,t)$ et/ou sa dérivée.

Dans toute la suite, on adoptera des conditions aux limites de type température imposée.

I.C. Solutions en régime permanent

I.C.1. Résoudre l'équation obtenue à la question I.A.2. en régime permanent, avec les conditions aux limites de type températures imposées (question I.B.(i)) :

- pour un instant particulier négatif $t_1 < 0$,
- 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.

I.C.2. 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.

I.C.3. Sur le même graphique, tracer à la main qualitativement les profils intermédiaires à différents instants entre la variation brutale de la température extérieure ($t = 0$) et l'instant t_2 où le régime est de nouveau permanent.

PARTIE II : RESOLUTION NUMERIQUE

II.A. Equation à résoudre

On cherche à résoudre numériquement l'équation aux dérivées partielles :

$$\alpha \frac{\partial T}{\partial t} = \frac{\partial^2 T}{\partial x^2} \qquad (1)$$

où α est une constante. A l'équation (1) sont associées les conditions :

$$\begin{aligned} &T(0,t) = T_{int} && \text{pour tout } \ t > 0 \\ &T(e,t) = T_{ext2} && \text{pour tout } \ t > 0 \\ &T(x,0) = ax + b && \text{pour tout } \ x \in [0,e] \end{aligned}$$

II.A.1. Quelle est l'expression de α en fonction des paramètres physiques du mur ?

II.A.2. Exprimer a et b en fonction de T_{int}, T_{ext1} et e.

Pour effectuer la résolution de l'équation (1), nous utiliserons la méthode des différences finies présentée dans la partie II.B.

II.B. Méthode des différences finies

II.B.1. Discrétisation dans l'espace et dans le temps

On divise l'intervalle $[0, e]$, représentant l'épaisseur du mur, en $N + 2$ points, numérotés de 0 à $N + 1$, régulièrement espacés de Δx (figure 2, page suivante). Cette division est appelée « discrétisation ». La distance Δx est appelée le « pas d'espace ». A l'intérieur du mur (frontières intérieure et extérieure exclues) se trouvent donc N points. On cherche à obtenir la température en ces points particuliers à chaque instant.

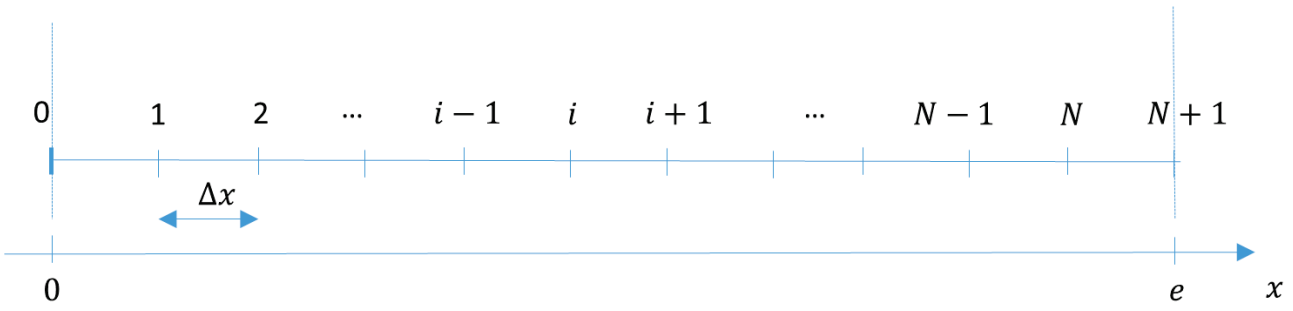


Figure 2 - Discrétisation spatiale dans la direction x.

II.B.1.a. Donner l'expression de Δx en fonction de N et de l'épaisseur du mur e.

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

Le temps est discrétisé en $ItMax$ intervalles de durée Δt et on ne s'intéresse au profil de température qu'aux instants particuliers $t_k = k.\Delta t$. L'intervalle élémentaire de temps Δt est appelé le « pas de temps ».

Pour résoudre l'équation (1), deux méthodes sont proposées :
- méthode utilisant un schéma explicite,
- méthode utilisant un schéma implicite.

II.B.2. Méthode utilisant un schéma explicite

II.B.2.a. A l'aide d'un développement limité de la fonction $x \longmapsto T(x,t)$, donner une expression de $T(x+\Delta x,t)$ à l'ordre 3 ($o(\Delta x^3)$) en fonction de T et de ses dérivées partielles par rapport à x évaluées en (x,t). De même, donner une expression de $T(x-\Delta x,t)$ à l'ordre 3.

II.B.2.b. En déduire une expression approchée à l'ordre 1 ($o(\Delta x)$) de $\left.\frac{\partial^2 T}{\partial x^2}\right|_{x,t}$ (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.

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 $T_{i-1}^k = T(x_i-\Delta x,t_k)$.

II.B.2.c. Déduire de la question précédente une expression approchée de $\left.\frac{\partial^2 T}{\partial x^2}\right|_{x_i,\,t_k}$ (dérivée partielle spatiale seconde de T évaluée en x_i à l'instant t_k) en fonction de T_i^k, T_{i+1}^k et T_{i-1}^k et Δx.

La dérivée partielle temporelle de l'équation (1) est maintenant approchée grâce à un développement limité.

II.B.2.d. A l'aide d'un développement limité de la fonction $t \longmapsto T(x,t)$, donner une expression de $T(x,t+\Delta t)$ à l'ordre 1 ($o(\Delta t)$) en fonction de T et de sa dérivée partielle par rapport à t évaluées en (x,t).

II.B.2.e. En déduire une valeur approchée de $\left.\frac{\partial T}{\partial t}\right|_{x,t}$ (dérivée partielle par rapport au temps de T évaluée au point x à l'instant t) à l'ordre 0 ($o(1)$) en fonction de $T(x,t+\Delta t)$, $T(x,t)$ et Δt.

II.B.2.f. Donner une expression de $\left.\frac{\partial T}{\partial t}\right|_{x_i,\, t_k}$ (dérivée partielle par rapport au temps de T évaluée en x_i à l'instant t_k) en fonction de Δt, T_i^k et T_i^{k+1}, avec $T_i^{k+1} = T(x_i, t_k + \Delta t)$.

L'équation (1) est valable en chaque point d'abscisse x_i et à chaque instant t_k.

II.B.2.g. Ecrire la forme approchée de cette équation au point i et à l'instant k en approchant $\left.\frac{\partial^2 T}{\partial x^2}\right|_{x,t}$ avec la formule obtenue à la question II.B.2.c. et en approchant $\left.\frac{\partial T}{\partial t}\right|_{x,t}$ avec la formule obtenue à la question II.B.2.f.

II.B.2.h. Montrer que l'équation obtenue à la question II.B.2.g peut s'écrire sous la forme :

$$T_i^{k+1} = rT_{i-1}^k + (1-2r)T_i^k + rT_{i+1}^k \qquad (2)$$

en précisant la valeur du paramètre r en fonction de $\Delta x, \Delta t$ et α.

L'équation (2) est appelée schéma numérique explicite. Si on connaît la température en tous les points $x_1, x_2, \ldots, x_{N-1}, x_N$ à l'instant t_k, on peut calculer grâce à elle la température en tous les points à l'instant ultérieur t_{k+1}.

II.B.2.i. L'équation (2) est-elle valable dans tout le domaine, c'est-à-dire pour toute valeur de $i,\ 0 \leq i \leq N+1$? Que valent T_0^k et T_{N+1}^k ?

II.B.2.j. Dans cette question, on élabore une fonction `schema_explicite` permettant de calculer la température en chaque point au cours du temps selon la formule (2). Parmi les variables d'entrée se trouvera un vecteur `T0` de dimension N, défini en dehors de la fonction, contenant les valeurs de la température aux points de discrétisation à l'instant initial. Au sein de la fonction, un algorithme calculera itérativement la température avec un nombre maximal d'itérations `ItMax`. En sortie de la fonction, on récupérera le nombre d'itérations réellement effectuées, `nbIter` et une matrice `T_tous_k`, de dimensions $N \times ItMax$. Chaque colonne de cette matrice contient le vecteur $\boldsymbol{T^k}$ dont les éléments sont les valeurs de la température aux N points $x_1, \ldots, x_N$ (points à l'intérieur du mur) à l'instant k :

$$\boldsymbol{T^k} = \begin{pmatrix} T_1^k \\ T_2^k \\ \ldots \\ T_{N-1}^k \\ T_N^k \end{pmatrix} \qquad \text{et} \qquad \texttt{T\_tous\_k} = \begin{pmatrix} T_1^1 & T_1^2 & \ldots & T_1^k & \ldots & T_1^{k-1} & T_1^k \\ T_2^1 & T_2^2 & & T_2^k & & T_2^{k-1} & T_2^k \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ T_{N-1}^1 & T_{N-1}^2 & \ldots & T_{N-1}^k & \ldots & T_{N-1}^{k-1} & T_{N-1}^k \\ T_N^1 & T_N^2 & & T_N^k & & T_N^{k-1} & T_N^k \end{pmatrix}.$$

On souhaite arrêter le calcul lorsque la température ne varie presque plus dans le temps. Dans ce but, on évaluera la norme 2 de $\boldsymbol{T^k - T^{k-1}}$ à chaque itération. La définition de la norme 2 est rappelée à la question II.B.2.j.*(vi)*.

II.B.2.j.*(i)* Ecrire l'en-tête de la fonction en précisant bien les paramètres d'entrée et de sortie.

II.B.2.j.*(ii)* Le schéma numérique (2) permet d'approcher avec succès la solution à la condition $r < 1/2$. Programmer un test qui avertit l'utilisateur si cette condition n'est pas respectée.

II.B.2.j.*(iii)* Affecter la valeur 2 000 à `ItMax`. Créer la matrice `T_tous_k` de dimensions $N \times ItMax$ en la remplissant de zéros.

II.B.2.j.*(iv)* Remplacer la première colonne de `T_tous_k` par le vecteur des valeurs initiales `T0`.

II.B.2.j.*(v)* Calculer le profil de température à l'instant $k = 1$ $(t = \Delta t)$, en distinguant le cas $i = 1$, le cas $2 \leq i \leq N - 1$ et le cas $i = N$. Affecter ces valeurs à la deuxième colonne de `T_tous_k`.

II.B.2.j.*(vi)* Ecrire une fonction `calc_norme` qui calcule la norme 2 d'un vecteur. On rappelle que la norme 2 d'un vecteur $\boldsymbol{V}$ s'écrit :

$$\|\boldsymbol{V}\|_2 = \sqrt{\sum_{i=1}^{N} V_i^2} \quad \text{avec} \quad \boldsymbol{V} = \begin{pmatrix} V_1 \\ \vdots \\ V_i \\ \vdots \\ V_N \end{pmatrix}.$$

II.B.2.j.*(vii)* Elaborer une boucle permettant de calculer itérativement le profil de température aux instants $t_k = k.\Delta t$ avec $k \geq 2$. Cette boucle sera interrompue lorsque la norme 2 du vecteur $\boldsymbol{T^k - T^{k-1}}$ deviendra inférieure à 10^{-2} ou lorsque le nombre d'itérations atteindra la valeur $ItMax$ (prévoir les deux cas). Utiliser, pour cela, la fonction `calc_norme` définie à la question II.B.2.j.*(vi)*.

II.B.2.j.*(viii)* Ecrire la fin de la fonction afin de renvoyer tous les arguments de sortie définis au début de la question II.B.2.j.

II.B.3. Méthode utilisant un schéma implicite

Le schéma explicite (2) ne converge que si le pas de temps Δt est suffisamment faible par rapport au pas d'espace Δx. Si l'on souhaite effectuer un calcul pour un temps physique long, beaucoup d'itérations seront nécessaires et le temps de calcul sera très long. C'est pourquoi on préfère d'autres types de schémas appelés schémas implicites.

Dans cette partie, la dérivée partielle seconde par rapport à x de la température apparaissant dans l'équation (1) est évaluée au point d'abscisse x_i et à l'instant $k + 1$:

$$\left.\frac{\partial^2 T}{\partial x^2}\right|_{x,t} \approx \left.\frac{\partial^2 T}{\partial x^2}\right|_{x_i, t_{k+1}}$$

et la dérivée partielle par rapport à t est évaluée au point d'abscisse x_i et à l'instant k :

$$\left.\frac{\partial T}{\partial t}\right|_{x,t} \approx \left.\frac{\partial T}{\partial t}\right|_{x_i, t_k}.$$

II.B.3.a. Donner la nouvelle expression approchée de l'équation (1) définie en page 3.

II.B.3.b. Montrer que l'équation obtenue à la question II.B.3.a. peut être mise sous la forme

$$T_i^k = -rT_{i-1}^{k+1} + (1 + 2r)T_i^{k+1} - rT_{i+1}^{k+1} \quad . \qquad (3)$$

L'équation (3) est appelée schéma implicite car la température à l'instant t_k est exprimée en fonction de la température à l'instant ultérieur t_{k+1}.

Le système d'équations ainsi obtenu peut être écrit sous la forme :

$$M\boldsymbol{T}^{k+1} = \boldsymbol{T}^{k} + r\,\boldsymbol{v} \qquad (4)$$

où M est une matrice carrée $N \times N$ et $\boldsymbol{T}^{k}$ et $\boldsymbol{T}^{k+1}$ sont les vecteurs de dimension N définis par :

$$\boldsymbol{T}^{k} = \begin{pmatrix} T_1^k \\ T_2^k \\ \dots \\ T_{N-1}^k \\ T_N^k \end{pmatrix} \quad \text{et} \quad \boldsymbol{T}^{k+1} = \begin{pmatrix} T_1^{k+1} \\ T_2^{k+1} \\ \dots \\ T_{N-1}^{k+1} \\ T_N^{k+1} \end{pmatrix}$$

et $\boldsymbol{v}$ est un vecteur de taille N faisant intervenir les conditions aux limites.

II.B.3.c. Préciser l'expression de la matrice M et l'expression du vecteur $\boldsymbol{v}$.

A chaque pas de temps, il faut inverser le système matriciel :

$$M\boldsymbol{T}^{k+1} = \boldsymbol{T}^{k} + r\,\boldsymbol{v}$$

pour obtenir $\boldsymbol{T}^{k+1}$à partir de $\boldsymbol{T}^{k}$.

II.B.3.d. Le but de cette question est d'écrire une fonction `CalcTkp1` qui permet de résoudre un système matriciel tridiagonal en utilisant l'algorithme de Thomas présenté ci-dessous.

Algorithme de Thomas :
On cherche à résoudre un système matriciel tridiagonal de la forme :

$$M\,\boldsymbol{u} = \boldsymbol{d} \qquad (5)$$

où M est une matrice de dimensions $N \times N$ tridiagonale, c'est-à-dire une matrice dont tous les éléments sont nuls, sauf sur la diagonale principale, la diagonale supérieure et la diagonale inférieure

$$M = \begin{pmatrix} b_1 & c_1 & & & & \\ a_2 & b_2 & c_2 & & 0 & \\ & a_3 & b_3 & c_3 & & \\ & & \ddots & \ddots & \ddots & \\ & 0 & & a_{N-1} & b_{N-1} & c_{N-1} \\ & & & & a_N & b_N \end{pmatrix}$$

et où les vecteurs $\boldsymbol{u}$ et $\boldsymbol{d}$, de dimension N, s'écrivent **:**

$$\boldsymbol{u} = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_{N-1} \\ u_N \end{pmatrix} \quad \text{et} \quad \boldsymbol{d} = \begin{pmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ d_{N-1} \\ d_N \end{pmatrix}$$

Dans cet algorithme, on calcule d'abord les coefficients suivants :

$$c'_1 = \frac{c_1}{b_1}$$
$$c'_i = \frac{c_i}{b_i - a_i c'_{i-1}} \quad \text{pour} \quad i = 2, 3, \dots, N-1$$

et

$$d'_1 = \frac{d_1}{b_1}$$
$$d'_i = \frac{d_i - a_i d'_{i-1}}{b_i - a_i c'_{i-1}} \quad \text{pour} \quad i = 2, 3, \dots, N\,.$$

Les inconnues $u_1, u_2, \dots, u_N$ sont alors obtenues par les formules :

$$u_N = d'_N$$
$$u_i = d'_i - c'_i u_{i+1} \quad \text{pour} \quad i = N-1, N-2, \dots, 2, 1\ .$$

II.B.3.d.*(i)* En utilisant l'algorithme de Thomas, écrire une fonction `CalcTkp1` qui permet de calculer le vecteur $\boldsymbol{u}$, solution du système matriciel (5), à partir de la matrice *M* et du vecteur $\boldsymbol{d}$.

II.B.3.e. Dans cette question, une fonction `schema_implicite` est élaborée avec les mêmes arguments d'entrée et de sortie que la fonction `schema_explicite` (définis à la question II.B.2.j.**)** et qui utilise les mêmes critères d'arrêt (définis à la question II.B.2.j.*(vii)*).

II.B.3.e.*(i)* Ecrire l'en-tête de la fonction en précisant les paramètres d'entrée et de sortie.

II.B.3.e.*(ii)* Affecter la valeur 2 000 à `ItMax`. Créer la matrice `T_tous_k` dont les dimensions sont $N \times ItMax$ en la remplissant de zéros.

II.B.3.e.*(iii)* Remplacer la 1re colonne de `T_tous_k` par le vecteur des valeurs initiales `T0`.

II.B.3.e.*(iv)* Définir la matrice `M` et le vecteur `v` qui interviennent dans l'équation (4).

II.B.3.e.*(v)* Calculer le profil de température à l'instant $k = 1$ $(t = \Delta t)$. Affecter ces valeurs à la deuxième colonne de `T_tous_k`.

II.B.3.e.*(vi)* Ecrire une boucle permettant de calculer itérativement le profil de température aux instants ultérieurs $t_k = k \times \Delta t$ avec $k \geq 2$, en prévoyant un arrêt lorsque la norme 2 du vecteur $\boldsymbol{T^k - T^{k-1}}$ devient inférieure à 10^{-2} ou lorsque le nombre d'itérations atteint la valeur `ItMax` (prévoir les deux cas). Utiliser pour cela la fonction `calc_norme` définie à la question II.B.2.j.*(vi)*.

II.B.3.e.*(vii)* Ecrire la fin de la fonction afin de renvoyer tous les arguments de sortie définis au début de la question II.B.2.j.

II.C. Programme principal

II.C.1. Début du programme

II.C.1.a. Définir les variables `epais` (épaisseur du mur), `conduc` (conductivité thermique), `rho` (masse volumique), `Cp` (capacité thermique massique), `Tint` (température intérieure), `Text1`

(température extérieure pour les instants $t < 0$), `Text2` (température extérieure pour les instants $t > 0$), `N` (nombre de points de calcul **à l'intérieur du mur**) et `Dt` (intervalle de temps élémentaire) et leur affecter les valeurs correspondant au problème physique défini au début de l'énoncé. On prendra un nombre de points de discrétisation $N = 60$ et un pas de temps Δt de 25 secondes.

II.C.1.b. Calculer les coefficients `a` et `b` avec la formule trouvée à la question II.A.2.

II.C.1.c. Créer un vecteur `x` dont les éléments $x_1, x_2, \ldots, x_N$ sont définis à la question II.B.1.b.

II.C.1.d. Calculer le vecteur des températures initiales `T0`.

II.C.1.e. Calculer `alpha` selon la formule trouvée à la question II.A.1. Calculer `r` en utilisant la formule calculée à la question II.B.2.h.

II.C.2. Calcul des températures

II.C.2.a. Ecrire un morceau de programme qui demande à l'utilisateur quel schéma (explicite ou implicite) il souhaite utiliser et qui appelle la fonction correspondante.

II.C.3. Analyse du résultat

II.C.3.a. Ecrire un morceau de programme permettant de tracer sur un même graphique le profil de température en fonction de `x` tous les 100 pas de temps.

II.C.3.b. Faire afficher le temps en heures au bout duquel le régime permanent est établi.

Fin de l'énoncé

ANNEXE A : COMMANDES ET FONCTIONS USUELLES DE SCILAB

A=[a b c d;e f g h;i j k l]

Description : commande permettant de créer une matrice dont la première ligne contient les éléments a, b, c, d, la seconde ligne contient les éléments e, f, g, h et la troisième, les éléments i, j, k, l.

Exemple : A=[1 2 3 4 5;3 10 11 12 20;0 1 0 0 2]

⇨
```
 1.   2.   3.   4.   5.
 3.   10.  11.  12.  20.
 0.   1.   0.   0.   2.
```

A(i,j)

Arguments d'entrée : les coordonnées de l'élément dans le tableau A.

Argument de sortie : l'élément (i, j) de la matrice A.

Description : fonction qui retourne l'élément (i, j) de la matrice A. Pour obtenir toute la colonne j de la matrice A, on utilise la syntaxe $A(:, j)$. De même, pour accéder à l'intégralité de la ligne i de la matrice A, on écrit $A(i, :)$.

Exemple :
```
A=[1 2 3 4 5;3 10 11 12 20;0 1 0 0 2]
A(2,4)
⇨ 12
A(:,3)
⇨  3.
   11.
   0.
A(2,:)
⇨ 3.   10.   11.   12.   20.
```

x=[x1:Dx:x2]

Description : commande permettant de créer un vecteur dont les éléments sont espacés de Dx et dont le premier élément est x_1 et le dernier élément est le plus grand multiple de Dx inférieur ou égal à x_2.

ATTENTION : le vecteur ainsi créé est un vecteur ligne. Pour convertir un vecteur ligne en un vecteur colonne, on le transpose en utilisant l'apostrophe « ' » : x_trans=x'.

Exemple *:*
```
x=[2:0.5:6.3]
⇨ 2.   2.5   3.   3.5   4.   4.5   5.   5.5   6.
x_trans=x'
⇨ 2.
  2.5
  3.
  3.5
  4.
  4.5
  5.
  5.5
  6.
```

zeros(n,m)

Arguments d'entrée : deux entiers n et m correspondant aux dimensions de la matrice à créer.

Argument de sortie : un tableau (matrice) d'éléments nuls.

Description : fonction créant une matrice (tableau) de dimensions $n \times m$ dont tous les éléments sont nuls.

Exemple : zeros(3,4)

⇨ 0. 0. 0. 0.
0. 0. 0. 0.
0. 0. 0. 0.

plot(x,y)

Arguments d'entrée : un vecteur d'abscisses x (tableau de dimension n) et un vecteur d'ordonnées y (tableau de dimension n).

Description : fonction permettant de tracer sur un graphique n points dont les abscisses sont contenues dans le vecteur x et les ordonnées dans le vecteur y.

Exemple :
```
x= [3:0.1:5]
y=sin(x)
plot(x,y)
```

ANNEXE B : BIBLIOTHEQUE NUMPY DE PYTHON

Dans les exemples ci-dessous, la bibliothèque numpy a préalablement été importée à l'aide de la commande : **import numpy as np**

On peut alors utiliser les fonctions de la bibliothèque, dont voici quelques exemples :

np.array(liste)

Argument d'entrée : une liste définissant un tableau à 1 dimension (vecteur) ou 2 dimensions (matrice).

Argument de sortie : un tableau (matrice).

Description : fonction permettant de créer une matrice (de type tableau) à partir d'une liste.

Exemple :
```
np.array([4,3,2])
⇨ [4 3 2]
np.array([[5],[7],[1]])
⇨ [[5]
   [7]
   [1]]
np.array([[3,4,10],[1,8,7]])
⇨ [[3 4 10]
   [1 8 7]]
```

$\boldsymbol{A[i,j]}$.

Arguments d'entrée : un tuple contenant les coordonnées de l'élément dans le tableau A.

Argument de sortie : l'élément $(i+1, j+1)$ de la matrice A.

Description : fonction qui retourne l'élément $(i+1, j+1)$ de la matrice A. Pour obtenir toute la colonne j+1 de la matrice A, on utilise la syntaxe $A[:, j]$. De même, pour accéder à l'intégralité de la ligne i+1 de la matrice A, on écrit $A[i, :]$.

ATTENTION : en langage Python, les lignes d'un A de dimension $n \times m$ sont numérotées de 0 à $n-1$ et les colonnes sont numérotées de 0 à $m-1$

Exemple :
```
A=np.array([[3,4,10],[1,8,7]])
A[0,2]
⇨ 10
A[:,2]
⇨ [10 7]
A[1,:]
⇨ [1 8 7]
```

np.zeros((n,m))

Arguments d'entrée : un tuple de deux entiers correspondant aux dimensions de la matrice à créer.
Argument de sortie : un tableau (matrice) d'éléments nuls.
Description : fonction créant une matrice (tableau) de dimensions $n \times m$ dont tous les éléments sont nuls.
Exemple : np.zeros((3,4))

⇨ [[0 0 0 0]
[0 0 0 0]
[0 0 0 0]]

np.linspace(Min,Max,nbElements)

Arguments d'entrée : un tuple de 3 entiers.
Argument de sortie : un tableau (vecteur).
Description : fonction créant un vecteur (tableau) de $nbElements$ nombres espacés régulièrement entre Min et Max. Le 1er élément est égal à Min, le dernier est égal à Max et les éléments sont espacés de $(Max - Min)/(nbElements - 1)$:
Exemple : np.linspace(3,25,5)

⇨ [3 8.5 14 19.5 25]

ANNEXE C : BIBLIOTHEQUE MATPLOTLIB.PYPLOT DE PYTHON

Cette bibliothèque permet de tracer des graphiques. Dans les exemples ci-dessous, la bibliothèque matplotlib.pyplot a préalablement été importée à l'aide de la commande :
import matplotlib.pyplot as plt

On peut alors utiliser les fonctions de la bibliothèque, dont voici quelques exemples :

plt.plot(x,y)

Arguments d'entrée : un vecteur d'abscisses x (tableau de dimension n) et un vecteur d'ordonnées y (tableau de dimension n).
Description : fonction permettant de tracer sur un graphique de n points dont les abscisses sont contenues dans le vecteur x et les ordonnées dans le vecteur y. Cette fonction doit être suivie de la fonction **plt.show()** pour que le graphique soit affiché.
Exemple :

```
x= np.linspace(3,25,5)
y=sin(x)
plt.plot(x,y)
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```

plt.xlabel(nom)

Argument d'entrée : une chaîne de caractères.
Description : fonction permettant d'afficher le contenu de nom en abscisse d'un graphique.

plt.ylabel(nom)

Argument d'entrée : une chaîne de caractères.
Description : fonction permettant d'afficher le contenu de nom en ordonnée d'un graphique.

plt.show()

Description : fonction réalisant l'affichage d'un graphe préalablement créé par la commande **plt.plot(x,y)**. Elle doit être appelée après la fonction plt.plot et après les fonctions plt.xlabel et plt.ylabel.

CONCOURS COMMUNS POLYTECHNIQUES

RAPPORT SUR L'EPREUVE ECRITE 2015 DE MODELISATION DE SYSTEMES PHYSIQUES OU CHIMIQUES DE LA FILIERE PC

1/ CONSIGNES GENERALES :

a) Présentation du sujet

Le sujet portait sur la résolution de l'équation de la chaleur par des méthodes numériques. La première partie physique permettait d'établir les équations et les conditions aux limites sur lesquelles la seconde partie allait travailler pour trouver une solution approchée par des méthodes numériques. Deux méthodes classiques de résolution étaient proposées : une par méthode explicite, assez facile à mettre en œuvre mais nécessitant un pas de temps faible et une par méthode explicite, un peu plus délicate mais plus stable.

Le sujet était de difficulté moyenne et alliait des questions de physique, de mathématiques et de programmation. Un des objectifs du sujet était de faire réaliser par le candidat le programme dans son intégralité, morceau par morceau. En conséquence, certaines questions étaient redondantes.

L'énoncé était suffisamment clair puisque la plupart des candidats ont bien compris ce qui leur était demandé. Il était découpé en parties indépendantes et certains résultats intermédiaires étaient donnés, ce qui permettait aux candidats de ne pas être bloqués.

Le niveau de difficulté des questions était varié, ce qui a permis de classer les candidats.

Une annexe présentait les principales fonctions de Python et de Scilab utiles à la résolution de ce sujet, ce qui permettait d'aider les candidats aux connaissances informatiques fragiles et permettait de ne pas défavoriser les élèves 5/2.

b) Problèmes constatés par les correcteurs

- De nombreux candidats ne savent pas faire correctement un développement limité.

- Très peu de candidats utilisent correctement l'instruction « print ».

- Beaucoup d'erreurs d'indices.

- Beaucoup d'étudiants oublient d'initialiser les matrices.

- Certains étudiants confondent les signes « == » et « = ».

- De nombreux étudiants font des fonctions sans nécessité.

- Certaines questions de programmation pouvaient être traitées de différentes manières, ce qui rendait la notation difficile.

- La plupart des candidats ont abordé le sujet avec sérieux, sauf quelques rares candidats qui n'ont pas du tout traité les questions de programmation.

Un nombre non négligeable de candidats ne font pas encore la différence entre l'écriture mathématique d'une expression et son écriture en langage de programmation (grandeurs ou fonctions non définies, non-respect de la syntaxe pour les opérations de multiplication, division, élévation à la puissance, etc.), ce qui a été difficile à corriger et à sanctionner, surtout quand la réponse était correcte.

Dans la plupart des copies, les indentations sont clairement visibles, soit en respectant un décalage constant et clair, soit en indiquant directement les indentations avec des lignes verticales (objets au même niveau d'indentation) ou horizontales (nombre d'indentation à chaque ligne). Toutefois, certaines copies présentaient des indentations « fluctuantes ». Cela nuit fortement à la lecture du code.

Les copies étaient globalement propres à de rares exceptions près. Il est important que les candidats mettent en valeur les résultats à l'issue de leur raisonnement et indique clairement la question qu'ils sont en train de traiter.

Les commentaires étaient trop peu présents. Il est important que le correcteur puisse comprendre la démarche des candidats surtout lorsque plusieurs solutions sont envisageables. Certains candidats ont utilisé des couleurs différentes pour les commentaires, c'est une très bonne idée pour les faire ressortir.

Les candidats ont pour beaucoup utilisé les syntaxes de liste plutôt que la syntaxe tableau de numpy. Ceci a induit, pour une partie des copies, des programmations 'lourdes' et des erreurs d'affectation dans les tableaux. Certains ont alors programmé des opérations telles que la multiplication par un scalaire, l'addition et la différence, ce qui alourdissait inutilement la programmation.

Lors de l'écriture d'un « while », surtout avec plusieurs conditions d'arrêt, les candidats seraient bien avisés d'écrire la négation mathématique. « Je veux m'arrêter si P » correspond à « je continue tant que non(P) » ; cela éviterait peut-être des erreurs du type « or » au lieu de « and » et « < » au lieu de « >= ».

Des élèves ont fait un usage inutile de la commande « break ».

Des affectations de variables sont réalisées dans le mauvais sens (2000=nbIter).

2/ REMARQUES SPECIFIQUES :

PARTIE I : ETUDE PRELIMINAIRE

I.A. Equation gouvernant la température
I.A.1. Question souvent mal traitée par les candidats.

I.A.2. Question bien traitée. L'équation de la diffusion thermique est connue ainsi que l'expression du coefficient de diffusion en fonction des paramètres du solide.

I.B. Conditions aux limites
La condition aux limites de type flux nul a été rarement énoncée correctement.

I.C. Solutions en régime permanent
La résolution en régime permanent et le tracé des solutions sont dans l'ensemble très bien traités. Le tracé des profils intermédiaires est souvent bien réalisé : l'erreur la plus classique consiste en un tracé de droites où la température en $x = e$ varie au cours du temps.

PARTIE II : RESOLUTION NUMERIQUE

II.A. Equation à résoudre

II.A.1. Quelques erreurs.

II.A.2. Certains utilisent Text2 au lieu de Text1, mais ils n'ont pas été pénalisés.

II.B. Méthode des différences finies

II.B.1.a. et b. Questions bien réussies.

II.B.2. Méthode utilisant un schéma explicite.

II.B.2.a. Beaucoup d'erreurs dans le développement limité DL : les coefficients étaient faux, le DL était fait au mauvais ordre, oubli du o(Dx3).

II.B.2.b. et c. Questions plutôt bien réussies. Le candidat a obtenu des points pour le raisonnement même s'il a fait des erreurs aux questions précédentes.

II.B.2.d. Question bien réussie. Quelques oublis du o(Dt).

II.B.2.e. et f. Questions bien réussies.

II.B.2.g. et h. Questions plutôt bien réussies. Le candidat a obtenu des points pour le raisonnement même s'il a fait des erreurs aux questions précédentes.

II.B.2.i. Question bien réussie.

II.B.2.j.(i) Beaucoup oublient les variables "Tint" et "Text" en argument d'entrée.

II.B.2.j.(ii) Beaucoup d'élèves n'avertissent pas l'utilisateur de la mauvaise valeur de « r », contrairement à ce qui était demandé dans l'énoncé. Beaucoup ne semblent pas connaître la commande « print ».

II.B.2.j.(iii) Question bien traitée mais oubli fréquent de définir la valeur de « N ». Parfois, affectation inversée « 2000=NbIter ».

II.B.2.j.(iv) Question généralement bien traitée.

II.B.2.j.(v) Des erreurs d'indice.

II.B.2.j.(vi) Question très bien réussie dans l'ensemble.

II.B.2.j.(vii) Des erreurs dans la boucle « while » (utilisation de « or » au lieu de « and », erreurs dans les signes d'inégalité). Des erreurs d'indice.

II.B.2.j.(viii) Certains oublient de donner une valeur à « nbIter » avant de la renvoyer.

II.B.3. Méthode utilisant un schéma implicite.

II.B.3.a. et b. Questions plutôt bien réussies. Le candidat a obtenu des points pour le raisonnement même s'il a fait des erreurs aux questions précédentes.

II.B.3.c. Question assez bien réussie. Matrice M globalement bien identifiée. Beaucoup d'erreurs sur le vecteur v.

I.B.3.d.(i) Beaucoup d'erreur d'indices. Les vecteurs n'ont souvent pas le bon nombre d'éléments. Dernière boucle rarement écrite correctement.

I.B.3.e.(i) Beaucoup oublient « Tint » et « Text » en argument d'entrée.

I.B.3.e.(ii) Question bien traitée mais oubli fréquent de définir la valeur de « N ». Parfois, affectation inversée « 2000=NbIter ».

I.B.3.e.(iii) Question généralement bien traitée.

I.B.3.e.(iv) Beaucoup d'erreur d'indice dans les boucles et mauvaises définitions des éléments aux bords de la matrice.

I.B.3.e.(v) Peu de candidats ont répondu à cette question. Question plutôt bien traitée mais certains oublient de rajouter « r*v » au deuxième argument d'entrée de « CalcTkp1. »

I.B.3.e.(vi) Des erreurs dans la boucle « while » (utilisation de « or » au lieu de « and », erreurs dans les signes d'inégalité). Des erreurs d'indice. Certains élèves n'appellent pas la fonction « CalcTkp1 ».

I.B.3.e.(vii) Certains oublient de donner une valeur à « nbIter » avant de la renvoyer.

II.C. Programme principal

II.C.1.a. Certains confondent écriture mathématique et programmation et utilisent d'autres noms de variables que ceux donnés dans l'énoncé, avec des symboles grecs par exemple ou des indices/exposants dans les noms de variables. Certains se compliquent la tâche en utilisant des fonctions.

II.C.1.b. Question plutôt bien traitée sauf quand « Text2 » a été utilisé au lieu de « Text1 ».

II.C.1.c. Beaucoup d'erreurs dans cette question. Le vecteur n'a souvent pas la bonne taille ou ses indices sont faux.

II.C.1.d. Question bien réussie.

II.C.1.e. Certains confondent écriture mathématique et programmation.

II.C.2. Certains ne connaissent pas la commande « input ». Non renvoi de « T_tous_k » et « nbIter ».

II.C.3. Analyse du résultat

II.C.3.a. Les courbes étaient généralement présentes sur un graphique avec des labels aux axes, toutefois de trop nombreux candidats ont voulu tracer T(t) alors que T(x) était demandé. Certains oublient de faire afficher le graphique avec « pl.show() ».

II.C.3.b. La question a été très peu abordée alors qu'il ne s'agissait que d'une simple conversion nombre de pas de temps vers heure en utilisant Dt. Certains candidats ont oublié de diviser le temps par 3 600 pour obtenir un résultat en heures. Tous les candidats ne connaissent pas la commande « print ».

3/ CONCLUSIONS :

Une immense majorité des candidats a utilisé le langage Python. Moins de 10 élèves ont traité l'épreuve en langage Scilab.

De nombreuses questions étaient très faciles, en particulier dans la partie physique et dans la résolution numérique par schéma implicite. Quelques questions, en particulier pour le schéma implicite, étaient plus délicates. Le sujet a donc permis de trier les candidats.

Les étudiants ayant fait preuve de sérieux dans les différentes matières ont rencontré peu de difficulté.