Enseignant : Roger Sauser
ICS - CMS
14 juin 2023
Durée : 105 minutes

1

RS

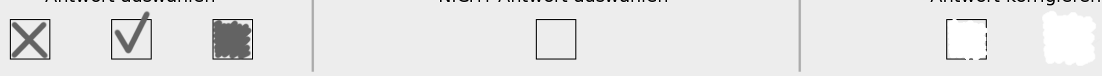
SCIPER : **22**

Attendez le début de l'épreuve avant de tourner la page. Ce document est imprimé recto-verso. Il contient 20 pages et 7 questions. Ne pas dégrafer.

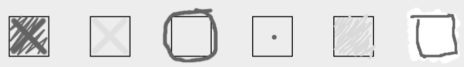
- Posez **votre carte d'étudiant** sur la table.
- **Aucun** document n'est autorisé.
- L'utilisation d'une **calculatrice** et de tout outil électronique est interdite pendant l'épreuve.
- Pour les questions à **choix unique** ("multiple choice"), on comptera :
  les points indiqués si la réponse est correcte,
  0 point s'il n'y a aucune ou plus d'une réponse inscrite,
  0 point si la réponse est incorrecte.
- Utilisez un **stylo** à encre **noire ou bleu foncé** et effacez proprement avec du **correcteur blanc** si nécessaire. Toute réponse doit être rédigée en utilisant la place réservée à cet effet à la suite de la question. N'écrivez **pas dans les marges !**
- Veuillez vous conformer aux indications suivantes pour les sujets qui demandent d'**écrire du code Python** (avec du papier et un stylo) :
  - respectez la syntaxe Python (parenthèses, crochets, accolades, deux points, mots-clés, etc.) ;
  - mettez en forme votre code pour qu'il soit formaté exactement comme si vous le tapiez en vue d'une exécution sans erreur ;
  - respectez les indentations (en sachant que la taille de l'indentation n'importe pas en soi, mais elle doit permettre d'identifier vos blocs de code de manière claire et immédiate) ;
  - votre réponse doit comporter uniquement du code exécutable, à l'exception de quelques commentaires (au format habituel) si ceux-ci sont véritablement nécessaires et aident à la compréhension.

Respectez les consignes suivantes | Observe this guidelines | Beachten Sie bitte die unten stehenden Richtlinien

choisir une réponse | select an answer Antwort auswählen	ne PAS choisir une réponse | NOT select an answer NICHT Antwort auswählen	Corriger une réponse | Correct an answer Antwort korrigieren



ce qu'il ne faut **PAS** faire | what should **NOT** be done | was man **NICHT** tun sollte



Première partie, trois questions de "type ouvert"

Répondez dans l'espace dédié. Laissez libres les cases à cocher : elles sont réservées au correcteur.
Cette première partie comprend un **total de 24 points** (**22 points + 2 points de bonus**).

Question 1: *Cette question est notée sur 12 points.*

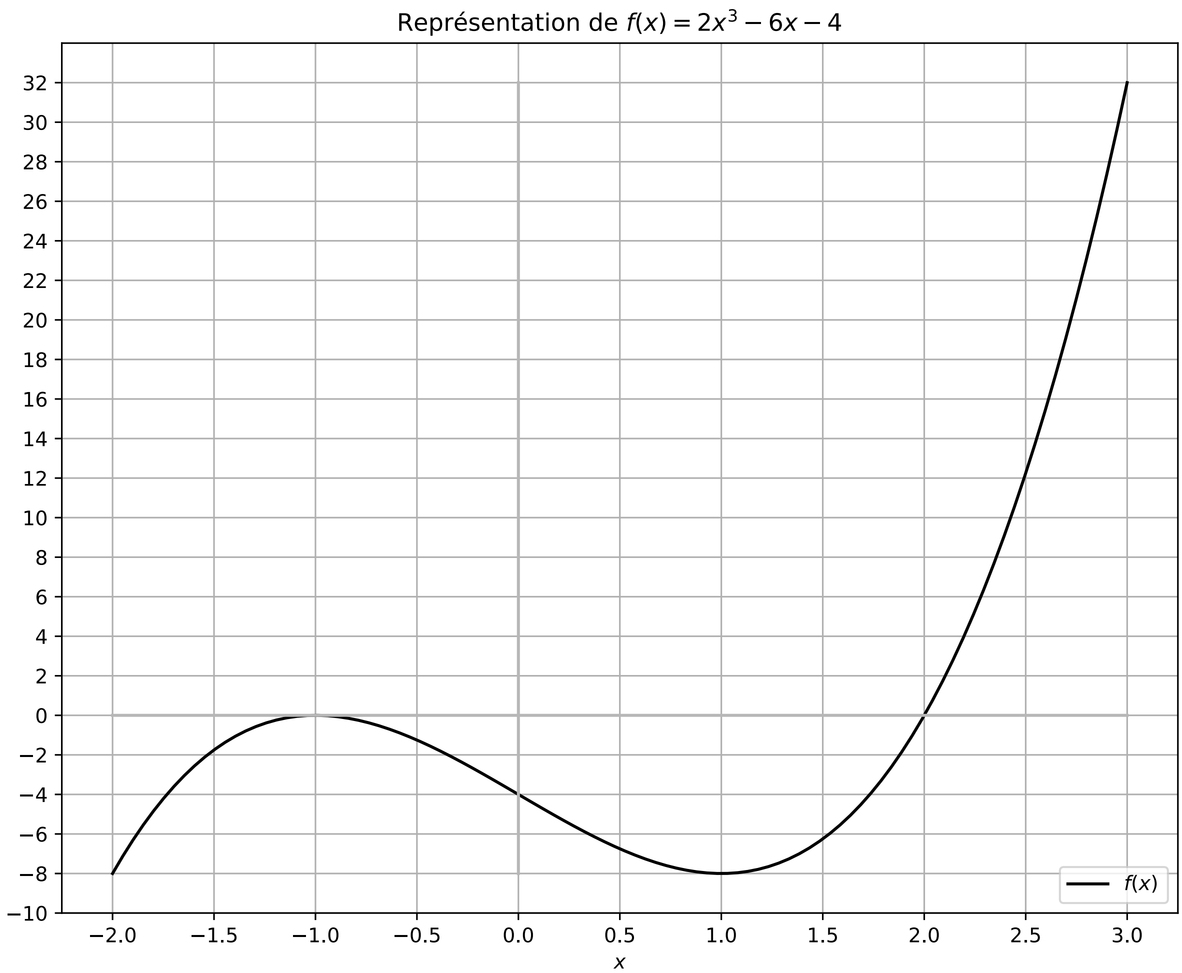
.5 .5 .5 .5 .5 .5 .5 .5 .5 .5 .5 .5
0 1 2 3 4 5 6 7 8 9 10 11 12

Soit la fonction réelle d'une variable réelle $f : \mathbb{R} \to \mathbb{R}$ définie par

$$f(x) = 2(x+1)^2(x-2)\,.$$

On donne ci-dessous la représentation graphique de cette fonction sur l'intervalle $[-2, 3]$ et on considère l'intégrale définie

$$I = \int_{-2}^{3} f(x)\, dx\,.$$



Les parties (a) et (b) en pages 3 et 4 peuvent être résolues de manière indépendante. Pour comprendre la partie (b), il faut toutefois avoir lu la partie (a).

(a) Complétez le code Python en page suivante de manière à ce qu'il permette de déterminer une approximation numérique J de l'intégrale définie I grâce à l'une des deux méthodes de quadrature composite suivantes :

- méthode basée sur la **formule dite du point de gauche** ;
- méthode basée sur la **formule de Simpson**.

Plus précisément, le code doit définir une fonction `integration` admettant comme arguments :

- la fonction `f` à intégrer ;
- les bornes `a` et `b` de l'intervalle d'intégration ;
- le nombre `n` de sous-intervalles à considérer ;
- la méthode choisie : `gauche` pour la méthode du point de gauche ou `simpson` pour la méthode de Simpson.

Cette fonction `integration` doit :

- vérifier que la borne `b` est bien plus grande que la borne `a` ; la fonction doit afficher le message "`L'intervalle [a,b] est mal défini.`" et retourner "`None`" si ce n'est pas le cas ;
- vérifier que la méthode donnée en argument à la fonction `integration` est soit `gauche` soit `simpson` ; si ce n'est pas le cas, la fonction doit afficher le message "`La méthode demandée n'est pas implémentée.`" et retourner "`None`" ;
- calculer le pas `dx` qui sera utilisé par la méthode composite ;
- déterminer la valeur `J` de l'approximation en implémentant la méthode composite choisie ;
- retourner la valeur `J` .

(le code à compléter se trouve en page suivante)

Code à compléter pour la partie (a) de la Question 1 :

```
# DEBUT du code à compléter
# (veuillez respecter les notations spécifiées dans l'énoncé)
#
# définition de la fonction Python integration
def

    if                                                                    # test de la cohérence de
                                                                          # l'intervalle et affichage
                                                                          # d'un message d'erreur
                                                                          #
                                                                          # test de la méthode
                                                                          # demandée et affichage
        print("La méthode demandée n'est pas implémentée.")              # d'un message d'erreur
                                                                          #


    else:
        J = 0.0
        dx =                                                              # calcul du pas dx


        x_gauche = a                  # initialisation des bornes des sous-intervalles
        x_droite = a + dx             # utilisées ensuite dans l'implémentation de la
                                      # méthode d'intégration numérique choisie

        for _ in range                         # implémentation de la méthode précisée
                                               # en argument
            if















        return J

# FIN du code à compléter
```

(b) Donnez l'affichage qui sera alors produit par la suite d'instructions suivante :

```
def f(x):
    return 2*(x+1)**2*(x-2)

print("J_gauche =",integration(f,-2,3,5,'gauche'))
print("J_simpson =",integration(f,-2,3,1,'simpson'))
```

Votre réponse doit être soigneusement justifiée, par exemple en vous appuyant sur la représentation graphique de la fonction f donnée en page 2. Comparez les valeurs approchées obtenues à la valeur exacte de I (sans forcément faire le calcul de la valeur exacte de I, mais en justifiant votre conclusion).

(réponse en page suivante)

Solution

(a)

```
# DEBUT du code à compléter
# (veuillez respecter les notations spécifiées dans l'énoncé)
#
# définition de la fonction Python integration
def integration(f,a,b,n,methode):

    if a >= b:                                              # test de la cohérence de
        print("L'intervalle [a,b] est mal défini.")         # l'intervalle affichage
                                                            # d'un message d'erreur
        return None                                         # (ligne facultative)

    elif (methode != 'gauche') and (methode != 'simpson'):  # test de la méthode
        print("La méthode demandée n'est pas implémentée.") # demandée affichage
                                                            # d'un message d'erreur
        return None                                         # (ligne facultative)

    else:
        J = 0.0
        dx = (b-a)/n                                        # calcul du pas dx

        x_gauche = a                                        # initialisation des bornes
        x_droite = a + dx                                   # des sous-intervalles
                                                            # utilisées ensuite dans
                                                            # l'implémentation de la
                                                            # méthode d'intégration
                                                            # numérique choisie

        for _ in range(1,n+1):                              # implémentation de la méthode
                                                            # précisée en argument
            if methode == 'gauche':
                J += f(x_gauche)*dx
            elif methode == 'simpson':
                J += (f(x_gauche) + 4*f((x_gauche + x_droite)/2)+ f(x_droite))*dx/6
            x_gauche = x_droite
            x_droite = x_gauche + dx

        return J
# FIN du code à compléter
```

(b)

```
J_gauche = -20.0
J_simpson = -2.5
```

Approximation avec la méthode du point de gauche (avec 5 sous-intervalles) :

$$J_{gauche} = -8\cdot 1 + 0\cdot 1 - 4\cdot 1 - 8\cdot 1 + 0\cdot 1 = -8 - 4 - 8 = -20\,.$$

Approximation avec la méthode de Simpson (avec 1 seul sous-intervalle) :

$$J_{simpson} = 5\cdot\frac{1}{6}\left(-8 + 4f(0.5) + 32\right) = 5\cdot\frac{1}{6}\left(-8 + 4(\frac{1}{4} - 7) + 32\right) = 5\cdot\frac{1}{6}\left(-8 + 1 - 28 + 32\right) = -\frac{5}{2} = -2.5\,.$$

Comparaison des approximations obtenues avec la valeur exacte de l'intégrale définie :

$$I = \int_{-2}^{3}\big(2(x+1)^2(x-2)\big)\,dx\int_{-2}^{3}\big(2x^3 - 6x - 4\big)\,dx = \left(\frac{1}{2}x^4 - 3x^2 - 4x\right)\Big|_{-2}^{3} = -2.5\,,$$

et

$$I = J_{simpson} > J_{gauche}\,.$$

Ce résultat n'est pas surprenant, dans la mesure où la méthode de Simpson est exacte pour les polynômes de degré 3.

Question 2: *Cette question est notée sur 3 points.*

.5 .5 .5

0 1 2 3

Soient $m+1$ noeuds (points) de quadrature distincts $-1 \leq t_0 < t_1 < \ldots < t_m \leq 1$ et $\{\varphi_0, \varphi_1, \ldots, \varphi_j, \ldots, \varphi_m\}$ la base de Lagrange de l'espace vectoriel $\mathcal{P}_m$ (espace vectoriel des polynômes de degré inférieur ou égal à m) associée à ces points de quadrature.

Soit la formule de quadrature

$$J(g) = \sum_{j=0}^{m} \mu_j g(t_j) \,.$$

Montrez que si les poids de quadrature μ_j, $\forall j \in \{0, 1, \ldots, m\}$, sont donnés par

$$\mu_j = \int_{-1}^{1} \varphi_j(t) dt \,,$$

alors le degré d'exactitude de la formule de quadrature ci-dessus est (au moins) m.

Solution
On suppose que les poids de quadrature sont donnés par les formules

$$\mu_j = \int_{-1}^{1} \varphi_j(t) dt \,.$$

On considère maintenant un polynôme quelconque $p \in \mathcal{P}_m$. Etant donné que la base de Lagrange est une base de $\mathcal{P}_m$, on peut écrire l'expression du polynôme p dans cette base:

$$p(t) = \sum_{j=0}^{m} p(t_j) \varphi_j(t) \,,$$

et on calcule

$$\begin{aligned}
\int_{-1}^{1} p(t) dt &= \int_{-1}^{1} \left(\sum_{j=0}^{m} p(t_j) \varphi_j(t) \right) dt \\
&= \sum_{j=0}^{m} p(t_j) \int_{-1}^{1} \varphi_j(t) dt \\
&= \sum_{j=0}^{m} p(t_j) \mu_j = J(p) \,.
\end{aligned}$$

Question 3: *Cette question est notée sur 9 points.*

.5 .5 .5 .5 .5 .5 .5 .5 .5
0 1 2 3 4 5 6 7 8 9

Soit $c(t)$ une fonction du temps t correspondant à la concentration d'une substance dans le sang d'un être vivant. On suppose que l'évolution de c est décrite par l'équation différentielle suivante :

$$c'(t) = 2\exp(-t) - (c(t))^2\,,$$

où le temps t est donné en heures et la concentration c en g/l. Cette équation différentielle est complétée par la condition initiale (à $t_0 = 0\,\text{h}$) :

$$c(0) = 0\,\text{g/l}\,.$$

Les parties (a) et (b) ci-dessous peuvent être résolues de manière indépendante. Pour comprendre la partie (b), il faut toutefois avoir lu la partie (a). La partie (b) est une question bonus valant 2 points sur les 9 points de la question.

(a) Complétez le code Python donné en page suivante de manière à ce qu'il permette de déterminer l'évolution de la concentration c de $t_0 = 0\,\text{h}$ à $T = 10\,\text{h}$.

Plus précisément, le code doit :

- importer la librairie NumPy ;
- définir le problème de Cauchy que l'on se propose de résoudre dans cette question en définissant la fonction $f(t, y)$ ainsi que la condition initiale ;
- créer un "vecteur" (`ndarray`) temps `t` avec 51 instants régulièrement espacés entre 0 et 10 heures ;
- créer un "vecteur" (`ndarray`) concentration `c` avec 51 éléments dont le premier élément correspond à la condition initiale et dont les autres éléments seront déterminés par la suite ;
- déterminer l'évolution de la concentration en utilisant la méthode de Crank-Nicolson (pour la partition régulière mentionnée) et mettre à jour au fur et à mesure les valeurs des éléments du "vecteur" `c` ; l'équation implicite intervenant dans la méthode de Crank-Nicolson doit être résolue à l'aide de la méthode de Picard (méthode du point fixe) en prenant comme point de départ la valeur obtenue en appliquant le schéma d'Euler progressif, et en choisissant une tolérance de 10^{-8} et un nombre maximum d'itérations de 20 ; la librairie SciPy ne doit pas être utilisée ;
- afficher la valeur approchée de la concentation après 10 heures.

(le code à compléter se trouve en page suivante)

Code à compléter pour la partie (a) de la Question 3 :

```
# DEBUT du code à compléter
# (veuillez respecter les notations spécifiées dans l'énoncé)
#
                                                     # importation de la librairie NumPy

# définition du problème de Cauchy :
# * équation différentielle à résoudre (membre de droite)
def



# * condition initiale (valeur de la concentration c en t=0)
y_0 = 0

t_0 = 0        # instant initial
T = 10         # instant final
N = 51         # nombre d'instants

tol =          # tolérance

i_max = 20     # nombre maximum d'itérations

t =

c =

c[0] = y_0

h = T/(N-1)

for n in range(N-1):
    fn = f(t[n], c[n])
    # calcul du point de départ avec Euler progressif


# FIN du code à compléter
```

(b) **Cette partie (b) est une question bonus valant 2 points.**

En représentant les résultats obtenus en fonction du temps (c'est-à-dire en représentant le "vecteur" `c` en fonction du "vecteur" `t`), on observe que la concentration commence par augmenter avant de diminuer.

En supposant avoir exécuté le code de la partie (a), donnez ci-dessous les quelques lignes de code supplémentaires permettant d'estimer après combien d'heures la concentration est redescendue à la moitié de la valeur maximale qu'elle a atteinte.

(réponse en page suivante)

Solution

(a)

```
# DEBUT du code à compléter
# (veuillez respecter les notations spécifiées dans l'énoncé)
#
import numpy as np        # importation de la librairie NumPy

# définition du problème de Cauchy :
# * équation différentielle à résoudre (membre de droite)
def f(t,y):
    return 2*np.exp(-t) - y**2
# * condition initiale (valeur de la concentration c en t=0)
y_0 = 0

t_0 = 0      # instant initial
T = 10       # instant final
N = 51       # nombre d'instants

tol = 1e-8   # tolérance
i_max = 20   # nombre maximum d'itérations

t = np.linspace(t_0, T, N)
c = np.empty(N)

c[0] = y_0
h = T/(N-1)

for n in range(N-1):
    fn = f(t[n], c[n])
    # calcul du point de départ avec Euler progressif
    c_old = c[n]
    c_new = c[n] + h*fn
    i = 1
    while abs(c_new - c_old) > tol and i < i_max:
        c_old = c_new
        c_new = c[n] + (t[n+1]-t[n]) / 2 * (fn + f(t[n+1], c_old))
        i += 1
    c[n+1] = c_new

print(c[-1])

# FIN du code à compléter
```

(b)

```
i = 0
while c[i+1]>c[i]:
    i+=1
while c[i] > max(c)/2 :
    i+=1
print(t[i])
```

Seconde partie, quatre questions à choix unique

Pour chaque question, marquez la case correspondante à la réponse correcte sans faire de ratures. Il n'y a qu'**une seule réponse correcte** par question. Cette seconde partie comprend un **total de 8 points**.

Question 4 (à 2 points)

On cherche à intégrer sur un intervalle $[a, b]$ une fonction d'une variable réelle suffisamment régulière (plus précisément, une fonction dont les dérivées sont continues). Pour ce faire, on utilise une formule de quadrature composite basée sur une méthode de quadrature non composite de degré d'exactitude 3.
En passant d'une partition régulière comprenant n sous-intervalles à une partition régulière plus fine comprenant $3n$ sous-intervalles, on observe que l'erreur absolue commise est divisée par un certain facteur k. Quel est, en bonne approximation, ce facteur ?

- [ ] $k = 8$
- [ ] $k = 9$
- [ ] $k = 64$
- [ ] $k = 16$
- [x] $k = 81$
- [ ] $k = 3$

Question 5 (à 2 points)

On considère le polynôme de Lagrange $p(t)$ associé aux cinq points suivants :

$$P_0 = (-2, 1),\ P_1 = (-1, 0),\ P_2 = (0, -1),\ P_3 = (1, -2) \text{ et } P_4 = (2, 21)\,.$$

Parmi les affirmations suivantes, laquelle est fausse ?

- [ ] Le polynôme $p(t)$ a pour expression : $p(t) = t^4 + 2t^3 - t^2 - 3t - 1$.
- [ ] Le polynôme $p(t)$ est de degré 4 .
- [x] Le polynôme $p(t)$ peut s'écrire sous la forme d'une combinaison linéaire : $p(t) = \sum_{j=0}^{3} p_j \varphi_j(t)$, où les polynômes $\varphi_j(t)$, $j \in \{0, 1, 2, 3\}$, forment une base de l'espace vectoriel des polynômes de degré inférieur ou égal à 4 .
- [ ] Le polynôme $p(t)$ est l'unique polynôme d'interpolation pour les cinq points P_0, P_1, P_2, P_3 et P_4.
- [ ] Le polynôme $p(t)$ passe par les cinq points P_0, P_1, P_2, P_3 et P_4 .

Question 6 (à 2 points)

Soit le problème de Cauchy suivant :

$$\begin{cases} y'(t) &= t^2 \sin(y(t)), \ \forall t \in I = [t_0, T], \\ y(t_0) &= \dfrac{\pi}{2}. \end{cases}$$

Vers quelle valeur réelle va tendre la solution $y(t)$ lorsque l'on considère des temps t grands (c'est-à-dire lorsque l'on considère un intervalle I avec T grand) ?

- ☐ -2π
- ☐ 0
- ☐ $-\pi/2$
- ☐ 2π
- ■ π
- ☐ $-\pi$

Question 7 (à 2 points)

A laquelle des équations différentielles suivantes correspond le champ de directions représenté ci-dessous ?

- ☐ $y'(t) = y(t) + \exp(t)$
- ☐ $y'(t) = y(t) - \exp(t)$
- ☐ $y'(t) = -y(t) - \exp(-t)$
- ■ $y'(t) = -y(t) + \exp(t)$