Chapitre[2] Intégration numérique - simple et multiple

SAMIR KENOUCHE - DÉPARTEMENT DES SCIENCES DE LA MATIÈRE - UMKB

MÉTHODES NUMÉRIQUES ET PROGRAMMATION

Résumé

Ce chapitre aborde les différentes méthodes d'intégration numérique permettant de calculer une approximation de l'intégrale d'une fonction. Nous nous bornerons aux méthodes d'intégration usuelles (point milieu, trapèze et Simpson) utilisées en sciences expérimentales pendant les premières années du cycle universitaire. Ce cours est destiné aux étudiants (es) en deuxième année des filières Physique et Chimie. Par ailleurs, afin de tester et de consolider les concepts théoriques développés, l'étudiant (e) est amené (e) à suivre assidument les séances de travaux pratiques adossés à ce module. La dernière section est donnée à cet effet. La mise en pratique de ces méthodes d'intégration sera conduite par le biais du logiciel Matlab®.

Mots clés

Méthode du point milieu, méthode du trapèze, méthode de Simpson, intégrales multiples, scripts Matlab®.

Table des matières

I. INTRODUCTION

Très souvent le calcul explicite de l'intégrale, d'une fonction f continue sur $[a, b]$ dans $\mathbb{R}$, définie par $I(f) = \int_a^b f(x)\,dx$ peut se révéler très laborieux et rébarbatif, ou tout simplement impossible à atteindre. Par conséquent, on fait appel à des méthodes numériques afin de calculer une approximation de $I(f)$. Pour ces méthodes numériques, la fonction f, est remplacée par une somme finie constituée de n sous-intervalles selon :

$$\int_a^b f(x)\,dx = \frac{(b-a)}{n} \times \sum_{k=1}^{n} f(x_k) \tag{1}$$

S. Kenouche est docteur en Physique de l'Université des Sciences et Techniques de Montpellier et docteur en Chimie de l'Université A. Mira de Béjaia.
Site web : voir http://www.sites.univ-biskra.dz/kenouche
Document corrigé, amélioré et actualisé le 19.09.2019.

Selon ce type d'évaluation, on calcule forcément une approximation (passage d'une intégrale à une somme) de la vraie valeur. La méthode d'intégration mise en œuvre est dite d'ordre p si l'erreur d'intégration :

$$Err(f) = \left| \int_a^b f(x)\,dx - \frac{(b-a)}{n} \times \sum_{k=1}^{n} f(x_k) \right| \tag{2}$$

est nulle quand f est un polynôme de degré inférieur ou égal à p et non nulle pour au moins un polynôme de degré supérieur ou égal à $p+1$, soit $f \in \mathbb{C}^{p+1}([a,b])$. Dans ce chapitre, nous allons étudier et construire quelques méthodes d'intégration usuelles dites *composites* dans lesquelles la fonction à intégrer est substituée par un polynôme d'interpolation sur chaque intervalle élémentaire $\Delta x = \dfrac{(b-a)}{n}$. Nous formaliserons cette notion d'intégrale au moyen du *théorème* ci-dessous.

a) ***Théorème fondamental de l'analyse*** *(intégral et dérivée):* soit $f \in \mathcal{D} \subset \mathbb{R} \longmapsto \mathbb{R}$ une fonction continue et $a \in \mathcal{D}$ alors la dérivée de l'intégrale de f est la fonction elle-même, ie :

$$\left[\int_a^x f(x)\,dx \right]' = f(x) \tag{3}$$

Nous pouvons trouver plusieurs formulations équivalentes pour ce *théorème*. Il se démontre comme suit :

$$\begin{aligned}
\left[\int_a^x f(x)\,dx \right]' &= \frac{1}{h} \left[\int_a^{x+h} f(x)\,dx - \int_a^x f(x)\,dx \right] \\
&= \frac{1}{h} \left[\int_x^a f(x)\,dx + \int_a^{x+h} f(x)\,dx \right] \\
&= \frac{1}{h} \left[\int_x^{x+h} f(x)\,dx \right] = \frac{1}{h} \times h \times f(x) \\
&= f(x)
\end{aligned}$$

Comme f est continue en x, nous pouvons écrire $\forall u \in [x-h\,,\,x+h] \Rightarrow f(u) \in [f(x)-\epsilon\,,\,f(x)+\epsilon]$ nous en déduisons l'encadrement suivant :

$$\frac{1}{h} \int_x^{x+h} [f(x)-\epsilon]\,du \leq \frac{1}{h} \int_x^{x+h} f(u)\,du \leq \frac{1}{h} \int_x^{x+h} [f(x)+\epsilon]\,du$$

Après intégration, nous obtenu :

$$f(x) - \epsilon \leq \frac{1}{h} \int_x^{x+h} f(u)\,du \leq f(x) + \epsilon$$

Cette inégalité est équivalente à l'écriture : $\forall \epsilon > 0, \exists \delta > 0, \forall h \in]0\,,\,\delta[\Rightarrow$ la distance :

$$\| \frac{1}{h} \left[\int_a^{x+h} f(x)\,dx - \int_a^x f(x)\,dx \right] - f(x) \| \leq \epsilon$$

Qui se lit littéralement que la dérivée de l'intégrale de f est ϵ proche de f.

II. INTÉGRALES SIMPLES

A des fins d'organisation, nous présenterons d'abord le fondement théorique de chaque méthode d'intégration avant d'écrire les scripts Matlab® correspondants.

A. *Méthode du point milieu*

Soit f une fonction continue sur l'intervalle $[a\,;\,b] \in \mathbb{R}$. On se propose dans cette section d'évaluer l'intégrale $I(f)$ par la méthode du point milieu. De façon générale nous avons :

$$I(f) = \int_a^b f(x)\,dx \tag{4}$$

Signalons que l'expression analytique de $f(x)$ peut être connue comme elle peut être inconnue.

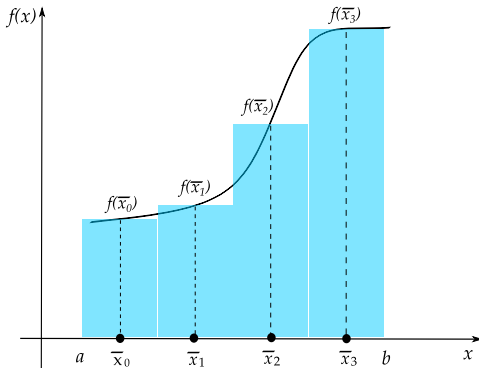

FIGURE 1: Formule du point milieu composite représentée sur 4 sous-intervalles

L'idée de base de cette méthode est de subdiviser l'intervalle [a, b] en n sous-intervalles $[x_k, x_{k+1}]$. Dans le cas où les sous-intervalles sont équidistants, on écrira $\Delta x = (b-a)/n$. Ainsi, le schéma numérique de cette méthode s'écrira comme :

$$I(f) = \sum_{k=1}^{n} \int_{I_k} f(x)\,dx \tag{5}$$

$$I(f) = \Delta x \times \sum_{k=1}^{n} f(\bar{x}_k) \qquad \text{avec} \qquad \bar{x} = \frac{x_{k+1} + x_k}{2} \tag{6}$$

Comme illustré par l'Éq. (6), pour chaque $\Delta x = [x_k, x_{k+1}]$ on prend le point central de l'ordonnée $f(\bar{x})$, valeur médiane entre $f(x_k)$ et $f(x_{k+1})$. En effet, $I(f)$, comme montré sur la figure ci-dessus, représente l'aire comprise entre la courbe $y = f(x)$ et l'axe des abscisses entre les droites $x = a$ et $x = b$. A titre illustratif, pour les quatre premiers sous-intervalles, on obtient :

$$I_1(f) \approx \Delta x \times f(\bar{x}_0)$$
$$I_2(f) \approx \Delta x \times f(\bar{x}_1)$$
$$I_3(f) \approx \Delta x \times f(\bar{x}_2)$$
$$I_4(f) \approx \Delta x \times f(\bar{x}_3)$$

Il existe plusieurs façons de mettre en œuvre la méthode des rectangles. Ainsi, on peut prendre la borne inférieure ou la borne supérieure sur chaque intervalle $[x_k, x_{k+1}]$. La méthode du point milieu est d'ordre zéro et son erreur d'intégration est majorée par :

$$\left| \int_a^b f(x)\,dx - \Delta x \times \sum_{k=1}^{n} f(\bar{x}_k) \right| \leq \frac{1}{2} \frac{(b-a)^2}{n} \max_{x \in [a,b]} \left| f^{(1)}(x) \right| \tag{7}$$

B. *Méthode du trapèze*

Cette méthode est basée sur l'interpolation, de chaque intervalle $[x_k, x_{k+1}]$, par un polynôme de degré un. En d'autres mots, sur chaque intervalle $[x_k, x_{k+1}]$, la fonction f continue et dérivable sur $[a, b]$, est substituée par la droite joignant les points $(x_k, f(x_k))$ et $(x_{k+1}, f(x_{k+1}))$. Le schéma numérique de la méthode du trapèze est donné par :

$$I(f) \approx \frac{\Delta x}{2}\left[f(a) + f(b)\right] + \Delta x \times \sum_{k=1}^{n-1} f(x_k) \tag{8}$$

Sur la figure ci-dessous, nous avons implémenté la formule du trapèze sur quatre sous-intervalles. Chaque trapèze est obtenu en remplaçant la fonction à intégrer par son polynôme de *Lagrange* de degré un.

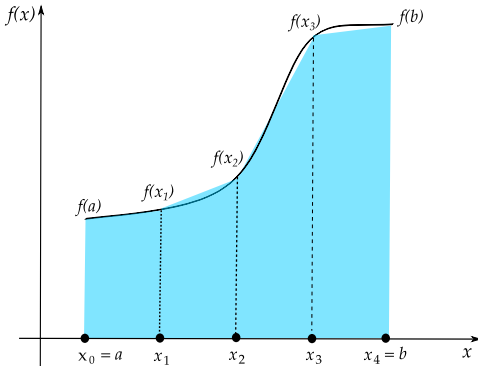

FIGURE 2: Formule du Trapèze composite représentée sur 4 sous-intervalles

Par exemple pour ces quatre trapèzes on obtient :

$$I_1(f) \approx \frac{x_1 - x_0}{2} \times \left[f(x_0) + f(x_1)\right]$$

$$I_2(f) \approx \frac{x_2 - x_1}{2} \times \left[f(x_1) + f(x_2)\right]$$

$$I_3(f) \approx \frac{x_3 - x_2}{2} \times \left[f(x_2) + f(x_3)\right]$$

$$I_4(f) \approx \frac{x_4 - x_3}{2} \times \left[f(x_3) + f(x_4)\right]$$

Tous les nœuds d'interpolation ont le même poids $\dfrac{\Delta x}{2}$. La méthode du trapèze est d'ordre deux $(O(h^2))$ et fournit une bien meilleure précision que la méthode du point milieu $(O(h))$. Il existe une version améliorée de la méthode du trapèze, dite méthode de *Poncelet* dont le schéma numérique est donné par :

$$I(f) \approx \frac{\Delta x}{4}(f(x_0) + f(x_{2n}) + 7 \times (f(x_1) + f(x_{2n-1})) + 8 \sum_{k=1}^{n-2} f(x_{2k+1})) \tag{9}$$

L'erreur d'intégration de la méthode du trapèze est majorée par :

$$\underbrace{\left|\int_a^b f(x)\,dx - I(f)\right|}_{Err(I(f))} \leq \frac{1}{12}\frac{(b-a)^3}{n^2} \max_{x\in[a,b]} \left|f^{(2)}(x)\right| \tag{10}$$

C. *Méthode de Simpson*

Cette méthode est basée sur l'interpolation, de chaque intervalle $[x_k, x_{k+1}]$, par un polynôme de degré deux. Ainsi, la fonction f est substituée par ce polynôme du second degré qui définit donc un arc de parabole passant par les points d'ordonnées $f(x_k)$, $f(x_{k+1})$ et $f(x_{k+2})$. Le schéma numérique est donné par :

$$I(f) \approx \frac{\Delta x}{6}[f(a) + f(b) + 2 \times \sum_{k=1}^{n-1} f(x_k) + 4 \times \sum_{k=1}^{n-1} f\left(\frac{x_{k+1} - x_k}{2}\right)] \tag{11}$$

Sur la figure ci-dessous, nous avons implémenté la formule ds *Simpson* sur quatre sous-intervalles. Ainsi, chaque sous-intervalle est interpolé par son polynôme de *Lagrange* de degré deux sur trois nœuds.

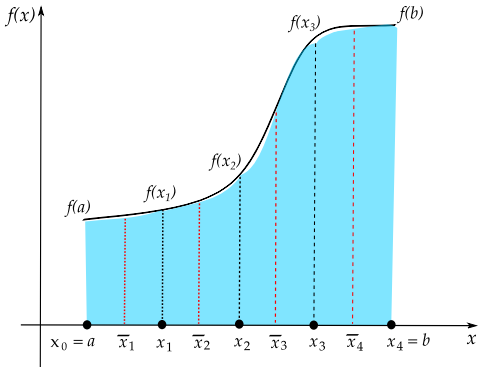

FIGURE 3: Formule de Simpson composite représentée sur 4 sous-intervalles

Par exemple pour les quatre premiers sous-intervalles on obtient :

$$I_1(f) \approx \frac{x_1 - x_0}{6} \times \left[f(x_0) + 4\, f\left(\frac{x_1 + x_0}{2}\right) + f(x_1)\right]$$

$$I_2(f) \approx \frac{x_2 - x_1}{6} \times \left[f(x_1) + 4\, f\left(\frac{x_2 + x_1}{2}\right) + f(x_2)\right]$$

$$I_3(f) \approx \frac{x_3 - x_2}{6} \times \left[f(x_2) + 4\, f\left(\frac{x_3 + x_2}{2}\right) + f(x_3)\right]$$

$$I_4(f) \approx \frac{x_4 - x_3}{6} \times \left[f(x_3) + 4\, f\left(\frac{x_4 + x_3}{2}\right) + f(x_4)\right]$$

Les nœuds d'interpolation situés aux extrémités sont pondérés en $\frac{\Delta x}{6}$. En revanche, les nœuds centraux sont pondérés en $2\,\frac{\Delta x}{3}$. La méthode de Simpson est d'ordre quatre $(O(h^4))$ et fait mieux que celle du trapèze. Ceci provient du fait qu'elle pondère plus le point central. L'erreur d'intégration de la méthode de Simpson est majorée par :

$$\left|\int_a^b f(x)\, dx - I(f)\right| \leq \frac{1}{2880}\, \frac{(b-a)^5}{n^4} \max_{x \in [a,b]} \left|f^{(5)}(x)\right| \tag{12}$$

Exercice ❶ ☞ ®

Quand on chauffe un corps, celui-ci dégage de la chaleur. Cela veut dire que le corps rayonne de l'énergie électromagnétique (EM) infrarouge même s'il n'émet aucune lumière visible. D'après la théorie du corps noir, un corps chauffé émet un rayonnement EM continu caractéristique de sa température. Planck établit la loi du rayonnement thermique sous la forme :

$$\rho(\lambda, T) = \frac{8\,\pi\,h\,c}{\lambda^5}\,\frac{1}{exp\left(\dfrac{h\,c}{k_b\,T\,\lambda}\right) - 1} \tag{13}$$

Cette loi postulait que l'énergie était émise et absorbée d'une manière fragmentée. Cette quantité étant toujours un multiple entier du quantum $h\nu$. Avec cette formule, la constante h apparaissait pour la première fois dans la science. A la découverte de cette formule, Max Planck disait :

"Après quelques semaines qui furent certes remplies par le travail le plus acharné de ma vie, un éclair se fit dans l'obscurité où je me débattais et des perspectives insoupçonnées s'ouvrirent à moi".

Cf. Max Planck
Initiation à la physique, trad., p.73.

Cette quantification s'est avérée indispensable à la compréhension des phénomènes énergétiques de l'atome. L'avènement de la constante de Planck a marqué le commencement de la *théorie quantique*. Dans le même sillage Louis de Broglie disait :

"Malgré l'importance et l'étendue des progrès accomplis par la physique dans les derniers siècles, tant que les physiciens ont ignoré l'existence des quanta, ils ne pouvaient rien comprendre à la nature intime et profonde des phénomènes physiques car, sans quanta, il n'y aurait ni lumière, ni matière et s'il est permis de paraphraser un teste évangélique, on peut dire que rien de ce qui a été fait n'a été fait sans eux "

Cf. Louis de Broglie
La physique nouvelle et les quanta, p.6.

Dans cet exercice, on désire quantifier l'énergie émise par un corps noir, à $T = 215\,K$, dans la gamme des longueurs d'onde comprises entre $3\,\mu m$ et $14\,\mu m$.

1) Évaluer numériquement cette intégrale par les méthodes du point milieu, du trapèze et de Simpson avec $n = 3$ sous-intervalles. Prendre $A = 2.40\,10^{-11}$ et $B = 1.43$.
2) Écrire le script Matlab® pour ces trois méthodes d'intégration.
3) Déterminer le nombre de sous-intervalles permettant d'atteindre une erreur d'intégration inférieure à 10^{-4}.

Exercice ❷ ☞ ®

1) Soit l'intégrale :

$$I(f) = \int_0^{2\,\pi} x\,exp(-x)\,cos(2\,x)\,dx \simeq -0.12212$$

2) Évaluer numériquement cette intégrale par les méthodes du point milieu, du trapèze et de Simpson avec $n = 3$ sous-intervalles. Conclure.
3) Écrire le script Matlab® pour ces trois méthodes d'intégration.
4) Tracer l'aire de l'intégrale pour $n = 150$ sous-intervalles.
5) Étudier l'influence du nombre de sous-intervalles (n) sur l'erreur d'intégration.
6) Appliquez les mêmes étapes pour l'intégrale :

$$I(f) = \int_0^1 exp\left(-\frac{x^2}{2}\right)\, dx \simeq +0.85608$$

Exercice ❸ ☞ Ⓡ

Considérons l'intégrale définie sur $[1,\ 3]$ par $f(x) = 1 + \log(x)$. Cette fonction est de classe $\mathcal{C}^2(x \in [1, 3])$.

– Déterminer le nombre de sous-intervalles permettant d'atteindre une erreur d'intégration inférieure à 10^{-3}.

Les scripts Matlab® des exercices 1 et 2 sont détaillés à la dernière page du document.

Exercice ❹ ☞ Ⓢ

Soient les intégrales définies par :

$$\begin{cases} f_1(x) = x - \log(x) \quad \text{avec} \quad x \in [1, 2] \\ f_2(x) = \cos(x)\, exp(x) \quad \text{avec} \quad x \in [0, \pi] \\ f_3(x) = \dfrac{1}{1 + (x - \pi)^2} \quad \text{avec} \quad x \in [0, 5] \end{cases} \tag{14}$$

– Déterminer par les méthodes du point milieu et du trapèze, le nombre de sous-intervalles permettant d'atteindre une erreur d'intégration inférieure à 10^{-5}.

III. Intégrales multiples

Dans cette section nous étendons la notion d'intégration aux cas des fonctions double et triple. En Physique et en Chimie, ces intégrales servent notamment à calculer des aires et des volumes.

A. *Intégrale double*

Formellement, une intégrale multiple, par exemple double, d'une fonction $f(x, y)$ est définie par :

$$I(f) = \int \int_{(x,y)\in\Omega} f(x, y)\, dx\, dy \tag{15}$$

Où Ω est le domaine suivant $\Omega = \{(x, y) \in \mathbb{R}^2,\ a \leq x \leq b, c \leq y \leq d\}$. Les bornes $(a < b, c < d)$ forment un rectangle ou un carré fermé du plan $\mathbb{R}^2$ dont les côtés sont parallèles aux axes de coordonnées. Soit une fonction f : $\mathbb{R}^2 \longrightarrow \mathbb{R}$ continue par morceaux sur $[a, b] \times [c, d]$, nous avons :

$$\int_a^b \int_c^d f(x, y)\, dx\, dy = \int_a^b \left(\int_c^d f(x, y)\, dy\right) dx = \int_c^d \left(\int_a^b f(x, y)\, dx\right) dy \tag{16}$$

C'est l'un des corolaires du *théorème de Fubini*. Ce théorème illustre la possibilité d'intégrer la fonction séparément sur l'intervalle horizontal et vertical. On peut commencer l'intégration soit avec la variable x ou bien la variable y, le résultat est le même. Toutefois, le niveau de difficulté du calcul de l'intégrale peut être sensiblement différent. Rappelons aussi que l'ordinateur ne sait résoudre que des problèmes discrets. Ainsi, l'idée de base de la résolutions numérique de ces intégrales, consiste à convertir un système initialement continu (intégrale) en un système discret (somme). Dans ce cas, on subdivise alors le domaine Ω au moyen de rectangles élémentaires de côtés $\Delta x_i = x_{i+1} - x_i$ et $\Delta y_i = y_{i+1} - y_i$ ensuite on prend dans chacun de ces rectangles le point moyen $(\bar{x}_i, \bar{y}_i)$ pour lequel on évalue la fonction $f(\bar{x}_i, \bar{y}_i)$. Où, $i = a_1 + \Delta x, a_1 + 2\Delta x, ..., a_1 + n\Delta x = n_x = a_2$ et $j = b_1 + \Delta y, b_1 + 2\Delta y, ..., b_1 + n\Delta y = n_y = b_2$. Avec, a_1, a_2, b_1 et b_2 sont respectivement les bornes d'intégration suivant les dimensions x et y.

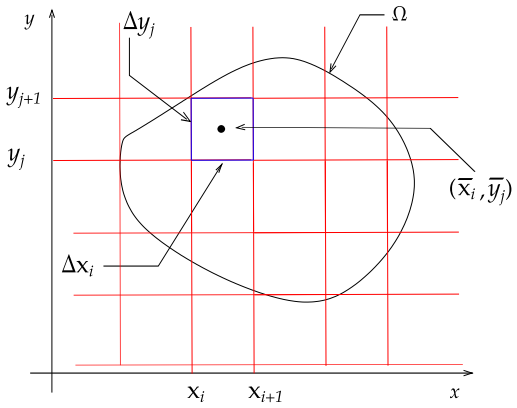

FIGURE 4: Discrétisation du domaine Ω

On aura ainsi la somme :

$$S_{n_x,n_y} = \sum_{i=1}^{n_x-1} \sum_{j=1}^{n_y-1} f(\bar{x}_i, \bar{y}_j)\, \Delta x_i\, \Delta y_i \tag{17}$$

Par définition

$$\int\int_{(x,y)\in\Omega} f(x,y)\, dx\, dy = \lim_{\substack{n_x\to\infty \\ n_y\to\infty}} S_{n_x,n_y} \tag{18}$$

De façon à ce que les surfaces définies par les rectangles $\Delta x_i\, \Delta y_j$ tendent vers zéro. La surface du domaine Ω s'obtient avec :

$$\int\int_{(x,y)\in\Omega} dx\, dy \tag{19}$$

On pourra définir, entre autre, la valeur moyenne de la fonction f, notée $< f(x,y) >$, dans le domaine Ω par :

$$< f(x,y) > = \frac{\displaystyle\int\int_{(x,y)\in\Omega} f(x,y)\, dx\, dy}{\displaystyle\int\int_{(x,y)\in\Omega} dx\, dy} \tag{20}$$

Notons par ailleurs que souvent un changement de variable s'impose. Il est d'usage de procéder à un changement de variable par le biais des coordonnées polaires. Dans ce cas, l'élément de surface $dx\, dy$ se transforme en $r\ drd\theta$. La formule de changement de variable s'écrit alors :

$$\int\int_{(x,y)\in\Omega} f(x,y)\, dx\, dy = \int\int_{(r,\theta)\in\Omega} f(r\ \cos(\theta), r\ \sin(\theta))\, r\, dr\, d\theta \tag{21}$$

Exercice ❺ ☞ ®

1) Soient les intégrales :

$$I_1 = \int_0^1 \int_0^1 (1 + x\,y + x^2 + y^2)\,dx\,dy = 1.916666666666667 \tag{22}$$

$$I_2 = \int_\pi^{2\,\pi} \int_0^\pi y^2\,\sin(x) + x^2\,\cos(y)\,dx\,dy = -20.670851083718247 \tag{23}$$

$$I_3 = \int_0^1 \int_0^2 (4 - x^2 - y^2)\,dx\,dy = 4.666666666666667 \tag{24}$$

2) Calculer analytiquement et numériquement l'intégrale (22).
3) Tracer le graphe de la fonction $f_2(x, y)$, $x \times y \in [\pi, 2\pi] \times [0, \pi]$.
4) Approcher numériquement, par deux approches différentes (de façon algorithmique et par des fonctions Matlab® prédéfinies), les intégrales (23) et (24).
5) Calculer numériquement la valeur moyenne de la fonction $f(x, y) = \exp(x - y)$ sur le domaine Ω défini par $\Omega = \{(x, y) \in \mathbb{R}^2,\, 0 \leq x \leq 2, 0 \leq y \leq 1\}$.

On commencera, dans un premier temps, par résoudre analytiquement l'intégrale I_1. D'après le *théorème de Fubini*, on peut écrire :

$$I_1 = \int_0^1 \int_0^1 (1 + x\,y + x^2 + y^2)\,dx\,dy = \int_0^1 \left(\int_0^1 (1 + x\,y + x^2 + y^2)\,dy \right) dx$$

$$= \int_0^1 \left[y + x\,\frac{y^2}{2} + x^2\,y + \frac{y^3}{3} \right]_0^1 dx = \int_0^1 (x^2 + \frac{x}{2} + \frac{4}{3})\,dx$$

$$\left[\frac{x^3}{3} + \frac{x^2}{4} + \frac{4\,x}{3} \right]_0^1 = \frac{1}{3} + \frac{1}{4} + \frac{4}{3} = 1.9167$$

De façon algorithmique :

```
clc ; clear all ;

% RESOLUTION ALGORITHMIQUE DE L'INTEGRALE DOUBLE I1
% LE 05/09/2019 - Samir KENOUCHE

lowerBound1 = 0 ; upperBound1 = 1  ;  % BORNES D'INTEGRATION SELON X
lowerBound2 = 0 ; upperBound2 = 1  ;  % BORNES D'INTEGRATION SELON Y
n = 800 ;                             % NOMBRE DE SOUS-INTERVALLES
dx = (upperBound1 - lowerBound1)/n ;  % PAS DE DISCRETISATION SELON L'AXE X
dy = (upperBound2 - lowerBound2)/n ;  % PAS DE DISCRETISATION SELON L'AXE Y
xi = lowerBound1 :dx: upperBound1  ;  % DISCRETISATION SELON L'AXE X
yi = lowerBound2 :dy: upperBound2  ;  % DISCRETISATION SELON L'AXE Y

fun = @(x,y) (1 + x*y + x.^2 + y.^2) ; nx = length(xi) ; ny = length(yi) ;

int = 0 ;  % INITIALISATION DE LA SOMME
```

```
for i = 1:nx-1
    for j = 1:ny-1

        xbar = (xi(i+1) + xi(i))/2 ;
        ybar = (yi(j+1) + yi(j))/2 ;

        int = int + fun(xbar,ybar)*dx*dy ;
    end
end

disp(strcat('LA VALEUR DE L''INTEGRALE I1 Vaut : ', num2str(int)))
```

La valeur de l'intégrale I_1 renvoyée est :

```
LA VALEUR DE L'INTEGRALE I1 Vaut :1.9167
```

Le script Matlab® ci-dessous, expose l'affichage de la fonction $f_2(x, y)$, le calcul de son intégrale I_2 formellement et au moyen de la fonction Matlab® prédéfinie `dblquad`.

```
clc ; close all ; clear all ;
%%%%%%%%%%%%%%%%%%%%%%%%%%%% INTEGRALES DOUBLES %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% LE 05/09/2019 Samir KENOUCHE

lowerBound1 = pi ; upperBound1 = 2*pi ; lowerBound2 = 0 ;
upperBound2 = pi ; n = 800 ; dx = (upperBound1 - lowerBound1)/n ;
dy = (upperBound2 - lowerBound2)/n ; xi = lowerBound1:dx:upperBound1;
yi = lowerBound2 :dy: upperBound2 ;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% GRAPHE DE f2 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[u, v] = meshgrid(xi,yi) ; w = v.^2.*sin(u)+ u.^2.*cos(u) ;

figure('color',[1 1 1]) ;
surf(u,v,w,'MarkerFaceColor','none') ; colormap(jet) ; shading interp
xlabel('x','FontSize',12) ; ylabel('y','FontSize',12) ;
zlabel('y^2 sin(x) + x^2 cos(y)','FontSize',12) ; grid off ;
axis auto ; view(-24,28)

%%%%%%%%%%%%%%%%%%%%%%%%%% INTEGRALE I2 : 1ERE APPROCHE %%%%%%%%%%%%%%%%%%%%%%%%

fun = @(x,y) y.^2*sin(x)+x.^2*cos(y) ; tol = 1e-08 ;
int1 = dblquad(fun, lowerBound1, upperBound1,lowerBound2, upperBound2, tol);

%%%%%%%%%%%%%%%%%%%%%%%%%% INTEGRALE I2 : 2EMME APPROCHE %%%%%%%%%%%%%%%%%%%%%%%%

nx = length(xi) ; ny = length(yi) ; int2 = 0;

for i = 1:nx-1
    for j = 1:ny-1
```

```
        xbar = (xi(i+1) + xi(i))/2 ;
        ybar = (yi(j+1) + yi(j))/2 ;

        int2 = int2 + fun(xbar,ybar)*dx*dy ;
    end
end


disp(strcat('LA VALEUR DE L''INTEGRALE int1 Vaut : '...
         , num2str(int1)))
disp(strcat('LA VALEUR DE L''INTEGRALE int2 Vaut : '...
         , num2str(int2)))
```

Les résultats renvoyés par le script sont affichés somme suit :

```
LA VALEUR DE L'INTEGRALE int1 Vaut :-20.6709
LA VALEUR DE L'INTEGRALE int2 Vaut :-20.6709
```

Les deux approches affichent strictement le même résultat de l'intégrale. La fonction `dblquad` permet de calculer numériquement une intégrale double. Sa syntaxe usuelle est donnée par :

```
[int, evalFun] = dblquad(fun, xmin, xmax, ymin, ymax, tol, method)
```

L'argument de sortie `int` constitue la valeur de l'intégrale calculée. Le deuxième argument de sortie `evalFun` désigne le nombre d'évaluation de la fonction. La fonction `dblquad` évalue l'intégrale double `fun(x,y)` sur le rectangle définit par `xmin <= x <= xmax` et `ymin <= y <= ymax` avec la tolérance considérée `tol`. Par défaut, la valeur de cette tolérance vaut : 10^{-6}. L'argument `fun` est une *fonction handle*. L'argument `method` indique la méthode utilisée pour résoudre l'intégrale. Il admet deux valeurs, à savoir : `method = @quad` (*adaptive Simpson quadrature*) prise par défaut ou bien `method = @quadl` (*adaptive Lobatto quadrature*).

Pour le calcul de la valeur moyenne, voici le script Matlab® :

```
clc ; close all ; clear all ; format short
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% LE 05/09/2019 Samir KENOUCHE

lowerBound1 = 0 ; upperBound1 = 2 ; lowerBound2 = 0 ;
upperBound2 = 1 ; n = 800 ; dx = (upperBound1 - lowerBound1)/n ;
dy = (upperBound2 - lowerBound2)/n ; xi = lowerBound1:dx:upperBound1;
yi = lowerBound2 :dy: upperBound2 ; fun = @(x,y) exp(x - y);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% MOYENNE %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
nx = length(xi) ; ny = length(yi) ; numer = 0 ; denum = 0 ;

for i = 1:nx
    for j = 1:ny

        numer = numer + fun(xi(i),yi(j))*dx*dy ;
        denum = denum + dx*dy ;
    end
```

```
end

moyVal = numer/denum ;

disp(strcat('LA VALEUR MOYENNE DE LA FONCTION Vaut :',...
    num2str(moyVal)))
```

L'affichage renvoyé est :

```
LA VALEUR MOYENNE DE LA FONCTION Vaut : 2.0203
```

Exercice ❺ ☞ Ⓢ

1) Calculer les intégrales doubles suivantes :

$$I(f) = \int\int_{(x,y)\in\Omega} f(x,y)\,dx\,dy \tag{25}$$

$$(a) \quad \Omega = \{(x,y) \in \mathbb{R}^2,\ 0 \leq x \leq 1, 1 \leq y \leq 2\}.$$

$$f(x,y) = x\,y\,\exp(-x-y) \tag{26}$$

$$(b) \quad \Omega = \{(x,y) \in \mathbb{R}^2,\ |x| < 1, |y| < 1\}.$$

$$f(x,y) = (x+y)\,\exp(x+y) \tag{27}$$

$$(c) \quad \Omega = \{(x,y) \in \mathbb{R}^2,\ 0 \leq x \leq y, 0 \leq y \leq 2\}.$$

$$f(x,y) = (x^2+y^2) \tag{28}$$

$$(d) \quad \Omega = \{(x,y) \in \mathbb{R}^2,\ 0 \leq x \leq 1, 0 \leq y \leq 1, x^2+y^2 \geq 1\}.$$

$$f(x,y) = \frac{x\,y}{1+x^2+y^2} \tag{29}$$

$$(e) \quad \Omega = \{(x,y) \in \mathbb{R}^2,\ 1 \leq x \leq 2, 0 \leq x\,y \leq \pi/2\}.$$

$$f(x,y) = \cos(x\,y) \tag{30}$$

$$(f) \quad \Omega = \{(x,y) \in \mathbb{R}^2,\ 0 \leq x \leq 2, 0 \leq y \leq 2\}.$$

$$f(x,y) = (x-y)\,\exp(x+5\,y) \tag{31}$$

$$(i) \quad \Omega = \{(x,y) \in \mathbb{R}^2,\ 0 \leq x \leq 1, 0 \leq y \leq 2\}.$$

$$f(x,y) = 1 - x^2\,y \tag{32}$$

B. *Intégrale triple*

Le principe de résolution des intégrales triples est analogue à celui des intégrales doubles. Il s'agit de déterminer sur quel domaine les variables varient et d'intégrer successivement par rapport aux trois variables d'intégration.

Soit une fonction f : $\mathbb{R}^3 \longrightarrow \mathbb{R}$ continue par morceaux sur $[a,b] \times [c,d] \times [e,f]$, de façon similaire que précédemment, le *théorème de Fubini* nous autorise à écrire :

$$\int_a^b \int_c^d \int_a^b f(x,y,z)\,dx\,dy\,dz = \int_a^b \left(\int_c^d \left(\int_e^f f(x,y,z)\,dz \right) dy \right) dx \tag{33}$$

La séquence d'intégration choisie est susceptible d'affecter le degré de difficulté du calcul. De la même manière que dans le cas des intégrales doubles, la discrétisation d'une intégrale triple se fait selon :

$$S_{n_x,n_y,n_z} = \sum_{i=1}^{n_x-1} \sum_{j=1}^{n_y-1} \sum_{k=1}^{n_z-1} f(\bar{x}_i, \bar{y}_j, \bar{z}_k)\,\Delta x_i\,\Delta y_i\,\Delta z_i \tag{34}$$

Par définition

$$\int\int\int_{(x,y,z)\in\Omega} f(x,y,z)\,dx\,dy\,dz = \lim_{\substack{n_x\to\infty \\ n_y\to\infty \\ n_z\to\infty}} S_{n_x,n_y,n_z} \tag{35}$$

De façon à ce que les volumes élémentaires $\Delta x_i\,\Delta y_j\,\Delta z_k$ de chaque parallélépipède tendent vers zéro. Le volume du domaine Ω s'obtient avec :

$$\int\int\int_{(x,y,z)\in\Omega} f(x,y,z)\,dx\,dy\,dz \tag{36}$$

D'autres systèmes de coordonnées, à l'instar des coordonnées cylindrique et sphérique, sont également utilisées pour résoudre ce type d'intégrale. En coordonnées cylindrique le volume élémentaire devient $dv = \rho d\rho d\theta dz$, ce qui donne :

$$\int\int\int_{(x,y,z)\in\Omega} f(x,y,z)\,dx\,dy\,dz = \int\int\int_{(\rho,\theta,z)\in\Omega} f(\rho\cos(\theta), \rho\,\sin(\theta), z)\,\rho\,d\rho\,d\theta\,dz \tag{37}$$

En coordonnées sphérique le volume élémentaire devient $dv = r^2\,\sin(\theta)\,dr\,d\phi\,d\theta$, ce qui donne :

$$\int\int\int_{(x,y,z)\in\Omega} f(x,y,z)\,dx\,dy\,dz = \int\int\int_{(r,\phi,\theta)\in\Omega} f(r\cos(\phi)\sin(\theta), r\sin(\phi)\,\sin(\theta), r\,\cos(\theta))\,r^2\,\sin(\theta)\,dr\,d\phi\,d\theta \tag{38}$$

Exercice ❻ ☞ ®

1) Soient les intégrales triples suivantes :

$$I_1 = \int_0^1 \int_1^2 \int_0^2 y^2\,x^2\,\sin(z)\,dx\,dy\,dz = 1.296296296296296 \tag{39}$$

$$I_2 = \int_0^{\pi}\int_0^{\pi}\int_{-1}^{1} z^2\,\sin(x) + z^2\,\cos(y)\,dx\,dy\,dz = 4.188790204291397 \tag{40}$$

$$I_3 = \int_0^{1}\int_0^{1}\int_0^{1} x^2\,y\,\exp(x\,y\,z)\,dx\,dy\,dz = 0.210000000000000 \tag{41}$$

2) Calculer analytiquement et numériquement l'intégrale I_1.
3) Déterminer l'erreur d'intégration.
4) Approcher numériquement, par deux approches différentes (de façon algorithmique et avec la fonction prédéfinie `triplequad`), les intégrales I_1, I_2 et I_3.

Commençons par résoudre analytiquement l'intégrale I_1 :

$$\int_0^{1}\int_1^{2}\int_0^{2} y^2\,x^2\,\sin(z)\,dx\,dy\,dz = \int_0^{1}\left(\int_1^{2}\left(\int_0^{2} y^2\,x^2\,\sin(z)\,dz\right)dy\right)dx \tag{42}$$

$$\int_0^{1}\left(\int_1^{2}\left(\left[-y^2\,x^2 cos(z)\right]_0^2\right)dy\right)dx = \int_0^{1}\left(\int_1^{2} 0.6\,y^2\,x^2 dy\right)dx$$

$$\frac{1}{1.8}\int_0^{1} 8\,x^2 - x^2 dx = \frac{1}{1.8}\left[\frac{8\,x^3}{3} - \frac{x^3}{3}\right]_0^1 = 1.2962$$

Pour la résolution numérique, voici le script Matlab® :

```
clc ; close all ; clear all ;
%%%%%%%%%%%%%%%%%%%%%%%%%% INTEGRALES TRIPLES %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% LE 05/09/2019 Samir KENOUCHE
lowerBound1 = 0 ; upperBound1 = 1 ; lowerBound2 = 1 ;
upperBound2 = 2 ; lowerBound3 = 0 ; upperBound3 = 2 ;

n = 200 ; dx = (upperBound1 - lowerBound1)/n ;
dy = (upperBound2 - lowerBound2)/n;dz=(upperBound3 - lowerBound3)/n ;
xi = lowerBound1 :dx: upperBound1 ;yi =lowerBound2 :dy: upperBound2 ;
zi = lowerBound3 :dz: upperBound3 ; fun = @(x,y,z) y.^2*x.^2*sin(z) ;

%%%%%%%%%%%%%%%%%%%%%%%%%% INTEGRALE I1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

nx = length(xi) ; ny = length(yi) ; nz = length(zi) ; int1 = 0;

for i = 1:nx - 1
    for j = 1:ny - 1
        for k = 1 : nz - 1

          xbar = (xi(i+1) + xi(i))/2 ;
          ybar = (yi(j+1) + yi(j))/2 ;
          zbar = (zi(k+1) + zi(k))/2 ;

        int1 = int1 + fun(xbar,ybar,zbar)*dx*dy*dz ;

        end
```

```
    end
end

disp(strcat('LA VALEUR DE L''INTEGRALE int1 Vaut : '...
         , num2str(int1)))
  intexact = 1.2962 ; err = abs(intexact - int1)/ intexact ;
```

Les sorties renvoyées par ce script sont :

```
LA VALEUR DE L'INTEGRALE int1 Vaut : 1.1014
>> err =
        0.1503
```

L'erreur d'intégration est de 15%, on peut bien entendu diminuer cette erreur en augmentant le nombre de points de discrétisation, mais dans ce cas on va accroitre le temps de calcul. Ainsi, un compromis doit être trouvé entre un temps de calcul raisonnable et une erreur d'intégration acceptable. Le calcul de l'intégrale I_2 par les deux approches est présenté suivant le script ci-dessous :

```
clc ; close all ; clear all ;
%%%%%%%%%%%%%%%%%%%%%%%%%%%% INTEGRALES TRIPLES %%%%%%%%%%%%%%%%%%%%%%%%%%%%
% LE 05/09/2019 Samir KENOUCHE

lowerBound1 = 0 ; upperBound1 = pi ; lowerBound2 = 0 ;
upperBound2 = pi ; lowerBound3 = -1 ; upperBound3 = 1 ;

n = 200 ; dx = (upperBound1 - lowerBound1)/n ;
dy = (upperBound2 - lowerBound2)/n ; dz = (upperBound3 - lowerBound3)/n ;
xi = lowerBound1 :dx: upperBound1 ;  yi = lowerBound2 :dy: upperBound2 ; zi =
   lowerBound3 :dz: upperBound3 ;

%%%%%%%%%%%%%%%%%%%%%%%% INTEGRALE I2 : 1ERE APPROCHE %%%%%%%%%%%%%%%%%%%%%%%%

fun = @(x,y,z) z.^2.*sin(x) + z.^2.*cos(y) ; tol = 1e-08 ;
int1 = triplequad(fun, lowerBound1, upperBound1,lowerBound2, upperBound2,
   lowerBound3, upperBound3, tol);

%%%%%%%%%%%%%%%%%%%%%%%% INTEGRALE I2 : 2EMME APPROCHE %%%%%%%%%%%%%%%%%%%%%%%%

nx = length(xi) ; ny = length(yi) ; nz = length(zi) ; int2 = 0;

for i = 1:nx - 1
    for j = 1:ny - 1
        for k = 1 : nz - 1

          xbar = (xi(i+1) + xi(i))/2 ;
          ybar = (yi(j+1) + yi(j))/2 ;
          zbar = (zi(k+1) + zi(k))/2 ;

        int2 = int2 + fun(xbar,ybar,zbar)*dx*dy*dz ;
```

```
        end
    end
end

disp(strcat('LA VALEUR DE L''INTEGRALE int1 Vaut : '...
         , num2str(int1)))

disp(strcat('LA VALEUR DE L''INTEGRALE int2 Vaut : '...
         , num2str(int2)))
```

Les résultats renvoyés par ce script sont affichés somme suit :

```
LA VALEUR DE L'INTEGRALE int1 Vaut :4.1888
LA VALEUR DE L'INTEGRALE int2 Vaut :4.1887
```

Il est très recommandé d'indenter son script, afin d'en améliorer sa lisibilité et ainsi détecter facilement d'éventuelles erreurs, comme par exemple, l'oubli d'un `end` dans une boucle. Ceci peut se révéler très utile notamment pour des scripts impliquant de nombreuses boucles. La syntaxe usuelle de la fonction `triplequad` est :

```
int = triplequad(fun,xmin,xmax,ymin,ymax,zmin,zmax,tol,method)
```

Elle évalue numériquement l'intégrale triple `fun(x,y,z)` sur un rectangle tridimensionnel définit par `xmin <= x <= xmax`, `ymin <= y <= ymax` et `zmin <= z <= zmay` avec la tolérance considérée `tol`. L'argument `fun` est une *fonction handle*. Comme pour les intégrales doubles, l'entrée `method` admet comme valeur : `method = @quad` prise par défaut ou bien `method = @quadl`. Signalons par ailleurs, que ces intégrales peuvent également être résolues en se servant de la boite à outil `Symbolic Math Toolbox`, suivant les instructions :

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clc ; clear all ;
syms x y z real

fun1 = y^2*sin(x) + x^2*cos(y) ; % INTEGRALE DOUBLE
int2Val = int(int(fun1, x,pi,2*pi), y, 0, pi) ;
int2Val = double(int2Val)

fun2 = z^2*sin(x) + z^2*cos(y) ; % INTEGRALE TRIPLE
int3Val = int(int(int(fun2, x, 0, pi), y, 0, pi), z, -1, 1) ;
int3Val = double(int3Val)
```

Ce qui donne :

```
>> int2Val =
            -20.6709 % integrale double
>> int3Val =
              4.1888 % integrale triple
```

Exercice ❼ ☞ Ⓢ

1) Calculer numériquement l'intégrale triple :

$$I(f) = \int\int\int_{(x,y,z)\in\Omega} f(x,y,z)\,dx\,dy\,dz$$

$$(a) \quad \Omega = \{(r,y,z) \in \mathbb{R}^3,\ 0 \le x \le 1, 0 \le y \le 1, 0 \le z \le 1\}.$$
$$f(x,y,z) = x^2 + y^2 + z^2$$

$$(b) \quad \Omega = \{(r,\theta,\phi) \in \mathbb{R}^3,\ 0 \le r \le R, 0 \le \theta \le 2\pi, -\pi/2 \le \phi \le \pi/2\}.$$
$$f(r,\theta,\phi) = r^2\ \cos(\phi)\,dr\,d\theta\,d\phi$$

$$(c) \quad \Omega = \{(r,y,z) \in \mathbb{R}^3,\ x \ge 0, y \ge 0, z \ge 0, x + y + z \le 1\}.$$
$$f(x,y,z) = 1 - x\,y\,z$$

IV. TRAVAUX PRATIQUES AVEC DES FONCTIONS MATLAB PRÉDÉFINIES

Il existe des fonctions Matlab® prédéfinies permettant de résoudre numériquement des intégrales simple, double et triple. Ces fonctions sont les suivantes : `quad`, `quadl`, `quadgk` et `quadv`. Les fonctions prédéfinies relatives au calcul des intégrales double et triple seront abordées lors des séances de travaux pratiques. La syntaxe usuelle de la fonction `quad` est la suivante :

```
[q, fcnt] = quad(fun, lowerB, upperB, Tol)
```

Ainsi, `quad` calcule numériquement l'aire (intégrale) sous la courbe de la *fonction anonyme* `fun`. Cette dernière peut également être écrite comme une *fonction M-file*. L'algorithme d'intégration de cette fonction prédéfinie est basé sur la méthode de *Simpson adaptative* (adaptive Simpson quadrature). Les arguments en entrée `lowerB` et `upperB` désignent respectivement les bornes inférieure et supérieure de l'intégrale. L'argument `Tol` désigne la tolérance considérée relative à l'erreur d'intégration. Par défaut, `Tol` vaut `1e-06`. L'argument de sortie `q` représente la valeur de l'intégrale calculée. La sortie `fcnt` exprime le nombre d'évaluation de la fonction à intégrer. Les fonctions `quadl` (adaptive Lobatto quadrature) et `quadv` (Vectorized quadrature) présentent une syntaxe similaire à celle de `quad`. La fonctions prédéfinie `quadgk` (adaptive Gauss-Kronrod quadrature) accepte d'autres arguments optionnels selon la syntaxe :

```
[q, errbnd] = quadgk(fun, lowerB, upperB, 'param1', val1, 'param2', val2,
  ....)
```

La sortie `errorbnd` exprime l'erreur absolue d'intégration $|Q - I|$, avec Q et I sont respectivement les valeurs approchée et exacte de l'intégrale. Parmi les arguments optionnels, on citera, `'AbsTol'` (Absolute error tolerance) dont la valeur par défaut vaut `1e-10` et `'RelTol'` (Relative error tolerance) dont la valeur par défaut vaut `1.e-6`. Voir aussi `'MaxIntervalCount'` (Maximum number of intervals allowed) et `'Waypoints'` (Vector of integration waypoints). Nous donnons dans l'exercice ci-dessous, un exemple d'utilisation de ces fonctions pour le calcul des intégrales simples :

Exercice ❶ ☞ ®

1) Au moyen des fonctions prédéfinies `quad`, `quadl` et `quadgk`, évaluer l'intégrale :

$$I(f) = \int_0^{2\,\pi} x\, exp(-x)\, cos(2\,x)\, dx \simeq -0.12212$$

2) Se servant de la fonction `quadv`, évaluer l'intégrale :

$$\int_0^{2\,\pi} x\, exp(-x)\, cos(k\,x)\, dx$$

pour différentes valeurs du paramètre $k = 2 : 24$.

3) Tracer la courbe représentant la valeur de l'intégrale en fonction du paramètre k.

A. *Supplément*

Il est également possible d'approcher une intégrale en y substituant la fonction à intégrer par sa série de Taylor. Ensuite on intègre chaque terme de la série.

Exemple 1

La formule de Simpson donne

$$\int_0^1 e^{-x^2/2}\, dx = 0.856$$

En procédant à un développement en série de Taylor

$$\begin{aligned}\int_0^1 e^{-x^2/2}\, dx &= \int_0^1 \left(1 - \frac{x^2}{2} + \frac{x^4}{8} - \frac{x^6}{48} + ...\right) dx \\ &= \left(x - \frac{x^3}{6} + \frac{x^5}{40} - \frac{x^7}{336} + ...\right)_{x=0}^{x=1} \\ &= 1 - 0.1666 + 0.0250 - 0.0029 + ... \\ &\simeq 0.854\end{aligned}$$

Soit une erreur inférieure à 1%

Exemple 2

Soit l'intégrale définie par :

$$\int_0^1 \sin(2\,x)\, dx$$

Puisque $\sin(2\,x) = 2\,\sin(x)\,\cos(x)$ on obtient la série en multipliant celle pour $\sin(x)$ par celle pour $\cos(x)$. Ensuite nous regroupons les termes suivant les puissances croissantes de x.

$$\begin{aligned}\sin(2\,x) &= 2\left(x - \frac{x^3}{6} + \frac{x^5}{120} - ...\right)\left(1 - \frac{x^2}{2} + \frac{x^4}{24} - ...\right) \\ &= 2\int_0^1 \left(x - \frac{2\,x^3}{3} + \frac{2\,x^5}{15} - ...\right) dx \\ &= 2\left(\frac{x^2}{2} - \frac{2\,x^4}{12} + \frac{2\,x^6}{90} - ...\right)_0^1 \\ &\simeq 0.711\end{aligned}$$

ANNEXE A
INTÉGRALES TRANSFORMÉES EN UNE SOMME D'UNE SÉRIE

Un certain nombre d'intégrales classiques trouvent des applications en Physique et en Chimie Quantique :

$$\int_0^{\infty} x^n\, e^{-x}\, dx = n!$$

$$\int_1^{\infty} x^n\, e^{-a\,x}\, dx = \frac{n!\, e^{-a}}{a^{n+1}}$$

$$\int_0^{\infty} \frac{x^n}{e^x - 1}\, dx = \sum_{k=0}^{\infty} \frac{n!}{(k+1)^4}$$

$$\int_0^{\infty} x^n\, e^{-a\,x}\, dx = \frac{n!}{a^{n+1}} \sum_{k=0}^{n} \frac{a^k}{k!} = A_n(a)$$

$$\int_{-1}^{+1} x^n\, e^{-a\,x}\, dx = (-1)^{n+1}\, A_n(-a) - A_n(a)$$

$$\int_0^{\pi} \cos(4\,x)\, \cos(\sin(x))\, dx = \pi \left(\frac{3}{2}\right)^4 \sum_{k=0}^{\infty} \frac{(-9/4)^k}{k!\,(k+4)!}$$

ANNEXE B
SÉRIES DE TAYLOR

Séries de Taylor de quelques fonctions élémentaires usuelles :

$$e^x = \sum_{k=0}^{\infty} \frac{x^k}{k!} \quad -\infty < x < +\infty$$

$$\log(1+x) = \sum_{k=1}^{\infty} \frac{(-1)^{k+1}}{k} x^k \quad |x| < 1$$

$$\cos(x) = \sum_{k=0}^{\infty} (-1)^k\, \frac{x^{2\,k}}{(2\,k)!} \quad -\infty < x < +\infty$$

$$\sin(x) = \sum_{k=0}^{\infty} (-1)^k\, \frac{x^{2\,k+1}}{(2\,k+1)!} \quad -\infty < x < +\infty$$

$$(1+x)^{\alpha} = \sum_{k=0}^{\infty} \frac{\alpha(\alpha-1)...(\alpha-k+1)}{k!}\, x^k \quad |x| < 1$$

Un peu d'histoire !

Pour la "petite" histoire, concernant le premier exercice du document. La "catastrophe ultraviolette", découle de la densité d'énergie infinie pour les grandes fréquences émises par un corps noir. Ce problème vient de la formule théorique développée par Rayleigh-Jeans, et qui contredisait l'observation. Max Planck en tentant de résoudre ce problème est parvenu à la formule :

$$\rho(\nu, T) = \alpha(\nu/c) \times \frac{1}{exp\left(\frac{\beta\,\nu}{T}\right) - 1}$$

Cette formule, publiée le 19 Octobre 1900, collait parfaitement avec les données spectrales d'un corps noir. Le mot noir vient d'une propriété de l'objet et ne traduit pas l'aspect visuel du corps. Un four par exemple est un corps noir. Pour interpréter cette formule, Planck partira d'une étude statistique de l'entropie pour chaque atome

constituant les parois du corps noir. Après plusieurs semaines de travail acharné, il tombe sur le même profil théorique où les constantes α et β sont remplacées par la constante h :

$$\rho(\lambda, T) = \frac{8\,\pi\,h\,c}{\lambda^5}\,\frac{1}{exp\left(\dfrac{h\,c}{k_b\,T\,\lambda}\right) - 1}$$

Le nom de cette constante vient de la première lettre du mot "au secours" en Allemand, publiée le 14 Décembre 1900. Il a utilisé ce mot car il était contraint d'admettre les arguments de Boltzmann sur le fait que l'entropie d'un système isolé est une propriété secondaire car elle dérive de mouvement des atomes. Planck défendait l'idée selon laquelle l'entropie est une propriété fondamentale qui ne peut donc émerger d'une autre propriété.

Dans un calcul intermédiaire entre les deux équations ci-dessus, il posa $\epsilon = h\,\nu$. Cette dernière stipule que les échanges d'énergies entre la matière et le rayonnement ne sont pas quelconques mais quantifiés. A l'époque Planck n'avait pas conscience de la portée de cette découverte qui allait mettre en déroute la physique classique. Il attribué la quantité $\epsilon = h\,\nu$ à un artéfact de calcul permettant l'annihilation de la "catastrophe ultraviolette". Cette découverte passait presque inaperçue pendant cinq ans, jusqu'au 1905 où Einstein citait cette formule dans un article et stipulait que non seulement les échanges sont quantifiés mais le rayonnement lui-même est de nature corpusculaire ...

Le développement de l'énergétisme quantique de Planck a démontré que les phénomènes se produisaient quand un système passait d'un état énergétique à un autre en émettant ou en absorbant de l'énergie.

Script Matlab® de l'exercice ❷

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%% METHODE DU POINT MILEU %%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  LE 05/09/2019 - Samir KENOUCHE :

a = 0 ; b = 2*pi ; n = 100 ; dx = (b - a)/n ; x = a :dx: b ;

fun =  x.*exp(-x).*cos(2.*x) ; int = 0 ;

for ik = 1:length(x)-1

    xbar = (x(ik) + x(ik+1))/2 ; func = eval('fun', xbar) ;

    int =  int + dx*func(ik) ;
end
disp(strcat('L''INTEGRALE PAR LA METHODE DU POINT MILIEU VAUT:',num2str(int)))

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% METHODE DU TRAPEZE %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

a = 0 ; b = 2*pi ; n = 1000 ;  dx = (b - a)/n ; x = a :dx: b ;

f =  x.*exp(-x).*cos(2.*x) ; int = 0 ; init = (f(1) + f(end))*(dx/2) ;

for ik = 1:length(x)-1

       int = int + dx*f(ik) ;
end

int = init + int
```

```
disp(strcat('L''INTEGRALE PAR LA METHODE DU TRAPEZE VAUT :',num2str(int)))

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% METHODE DE SIMPSON %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

a = 0 ; b = 2*pi ; n = 1000 ; dx = (b - a)/n ; x = a :dx: b ;

fun =  x.*exp(-x).*cos(2.*x) ; som1 = 0 ; som2 = 0 ;

for ik = 1:n-1

    som1 = som1 + fun(ik) ;  xbar = (x(ik+1) - x(ik))/2 ;

    fun2 = eval('fun',xbar) ; som2 = som2 + fun2(ik) ;

end

int = (dx/6)*(fun(1) + fun(end) + 2*som1 + 4*som2)

disp(strcat('L''INTEGRALE, PAR LA METHODE DE SIMPSON VAUT :',num2str(int)))
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% AIRE DE L'INTEGRALE %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

figure('color',[1 1 1]) ;
plot(x, fun,'LineWidth', 1) ; hold on

for ik = 1 : numel(x)

 plot([x(ik), x(ik)],[0, fun(ik)],'r','LineWidth', 1)

end

ih = gca ;
str(1) = {'Integral area of:'};
str(2) = {['$$\int_{0}^{2\pi}x \,\exp(-x)\, \cos(2 \, x)\, dx = $$',num2str(
   int)]} ;

set(gcf,'CurrentAxes', ih) ;

text('Interpreter','latex', 'String', str,'Position',[3 -0.2],'FontSize',12) ;
xlabel('x') ; ylabel('f(x)') ;
```

```
clear all ; clc ; close all ; format long
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% ERREUR D'INTEGRATION %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Samir Kenouche - le 05/09/2019

a = 0 ; b = 2*pi ; n = 150 ; ih = 0 ;
intexact = - 0.12212260461896 ;
```

```
for n = 60 : 20 : 600

      dx = (b - a)/n ; x = a :dx: b ; fun = x.*exp(-x).*cos(2.*x) ;

      som1 = 0 ; som2 = 0 ; ih = ih + 1 ;

    for ik = 1:n-1

     som1 = som1 + fun(ik) ; xbar = (x(ik+1) - x(ik))/2;

     fun2 = eval('fun',xbar) ; som2 = som2 + fun2(ik);

    end

    int(ih) = (dx/6)*(fun(1) + fun(end) + 2*som1 + 4*som2) ;
    err(ih) = abs((int(ih) - intexact)/intexact) ; % ERREUR RELATIVE

plot(n, err(ih), '+','MarkerSize',8, 'LineWidth', 1/2) ; hold on ;
xlabel('NOMBRE DE SOUS-INTERVALLES') ;
ylabel('ERREUR RELATIVE D''INTEGRATION (x 100 \%)') ;
end
```

Script Matlab® de l'exercice ❸

```
clc ; clear all ; close all ;

% LE 05/09/2019 - Samir kenouche

lB = 1 ; uB = 3 ; n = 10 ; dx = (uB - lB)/n ; xi = lB :dx: uB ;

syms x real

fun = 1 + log(x) ; dfun = diff(fun) ;  ddfun = diff(dfun) ;

pretty(dfun) ; pretty(ddfun) ;

ddfun_real = subs(ddfun, x, xi) ; % 2eme derivee

[maxi, idx] = max(ddfun_real) ;   % max(|f''(x)|) = maxi ==> maxi = 0.1111

ezplot(ddfun, [1 3])              % graphe de f''
```

Ainsi, afin d'atteindre une erreur $|Err(I(f))| < 10^{-3} \Leftrightarrow \dfrac{(3-1)}{12\,n^2} \times 0.1111 < 10^{-3} \Longrightarrow n^2 > 18.5 \Longrightarrow n > 4.30$. Il en résulte qu'à partir de cinq sous-intervalles, on atteint une erreur de quadrature inférieure à 10^{-3}.