INTÉGRATION NUMÉRIQUE

CHOKRI BEKKEY ET ZOUHAIER HELALI

Ce document présente quelques méthodes classiques de calcul numérique d'intégrales. Il est destiné à des étudiants de licence.

TABLE DES MATIÈRES

1. INTRODUCTION

1.1. **Problème étudié.** Soit une fonction $f : [a, b] \longrightarrow \mathbb{R}$ intégrable. Nous nous intéressons au calcul de son intégrale sur $[a, b]$:

$$I(f) = \int_a^b f(x)\, dx.$$

[1]Ce travail a été réalisé à l'occasion d'un projet Tempus, action JEP-31147-2003, impliquant d'une part l'université Paris-Sud, l'université de Lille (USTL) et l'université de Delft (TU Delft) et d'autre part l'université de Monastir (ISM et FSM) et l'université de Sousse (ISITC)

Dans ce chapitre on présente la théorie des quelques méthodes classiques de calcul numérique de $I(f)$. Ces méthodes sont appelées *méthodes de quadrature* . Pour chaque méthode, on s'intéresse à son ordre, à l'étude de sa convergence et à l'étude de son erreur de convergence. On développe aussi quelques idées nécessaires à l'écriture d'un programme numérique pour le calcul de $I(f)$.

1.2. **Notations et définitions.** Soit $f : [a,\, b] \longrightarrow \mathbb{R}$ bornée et soit

$$\sigma = \{a = x_0 < x_1 < \cdots < x_n = b\}$$

une subdivision de $[a,\, b]$ de pas

$$|\sigma| = \sup_{0\leq i\leq n} |x_{i+1} - x_i|$$

On pose :

$$m_i = \inf_{x\in]x_i,\ x_{i+1}[} f(x), \quad M_i = \sup_{x\in]x_i,\, x_{i+1}[} f(x)$$

$$s_\sigma(f) = \sum_{i=0}^{n-1} (x_{i+1} - x_i) m_i, \quad S_\sigma(f) = \sum_{i=0}^{n-1} (x_{i+1} - x_i) M_i$$

$$I_+(f) = \inf_\sigma S_\sigma(f), \quad I_-(f) = \sup_\sigma s_\sigma(f)$$

Définition 1 (Intégrale de Riemann)**.** La fonction f est dite *Riemann intégrable* si $I_+(f) = I_-(f)$. Dans ce cas, on note $I(f) = \displaystyle\int_a^b f(x)\, dx$ le réel $I_+(f) = I_-(f)$ et on l'appelle *l'intégrale de Riemann* associée à f.

Remarque 1. (1) Toute fonction continue par morceaux est Riemann intégrable.

(2) Toute fonction monotone est Riemann intégrable.

1.3. **Résultats fondamentaux.**

Proposition 1. *Si $f : [a,\, b] \longrightarrow \mathbb{R}$ est Riemann intégrable, alors*

$$\int_a^b f(x)\, dx = \lim_{|\sigma|\longrightarrow 0} S_\sigma(f) = \lim_{|\sigma|\longrightarrow 0} s_\sigma(f)$$

ou d'une manière équivalente

$$\forall \varepsilon > 0,\ \exists \eta > 0 \ \ tq \ \ \forall \sigma,\ |\sigma| < \eta \Longrightarrow \begin{array}{rcl} |I(f) - S_\sigma(f)| & \leq & \varepsilon \\ |I(f) - s_\sigma(f)| & \leq & \varepsilon \end{array}$$

Remarque 2. Si $f : [a,\, b] \longrightarrow \mathbb{R}$ est continue alors

$$\inf_{x\in]x_i,\, x_{i+1}[} f(x) = \inf_{x\in [x_i,\, x_{i+1}]} f(x) \ \text{ et } \ \sup_{x\in]x_i,\, x_{i+1}[} f(x) = \sup_{x\in [x_i,\, x_{i+1}]} f(x)$$

Théorème 1. *Si $f : [a,\, b] \longrightarrow \mathbb{R}$ est continue alors*

$$\int_a^b f(x)\, dx = \lim_{n\longrightarrow +\infty} \frac{b-a}{n} \sum_{i=0}^{n-1} m_i = \lim_{n\longrightarrow +\infty} \frac{b-a}{n} \sum_{i=0}^{n-1} M_i$$

et d'une façon plus générale

$$\int_a^b f(x)\, dx = \lim_{n\longrightarrow +\infty} \frac{b-a}{n} \sum_{i=0}^{n-1} f(c_i) \ \ avec \ \ c_i \in [x_i,\, x_{i+1}]$$

2. FORMULES DE QUADRATURE ET LEUR ORDRE

2.1. **Idée de base.** La plupart des algorithmes numériques procèdent comme suit : on subdivise l'intervalle $[a,\ b]$ en plusieurs sous-intervalles $\sigma = \{a = x_0 < x_1 < \cdots < x_n = b\}$ et on utilise le fait que

$$\int_a^b f(x)\,dx = \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} f(x)\,dx$$

De cette manière, on est amené au calcul de plusieurs intégrales pour lesquelles la longueur de l'intervalle d'intégration est relativement petite. Prenons une de ces intégrales, notons $h_i = x_{i+1} - x_i$ la longueur de l'intervalle et $g(t) = f(x_i + t h_i)$. Un changement de variable nous donne alors :

$$\int_{x_i}^{x_{i+1}} f(x)\,dx = h_i \int_0^1 g(t)\,dt$$

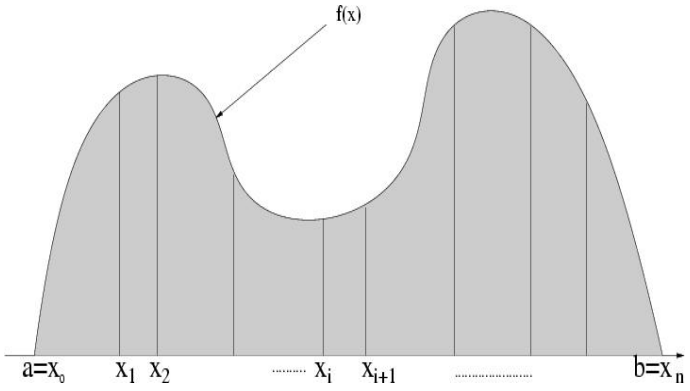


FIG. 1. Une subdivision d'un intervalle en sous-intervalles

Il reste alors à calculer une approximation de

$$\int_0^1 g(t)dt$$

2.2. **Méthode des rectangles à gauche.**

$$\int_0^1 g(t)dt \approx g(0)$$

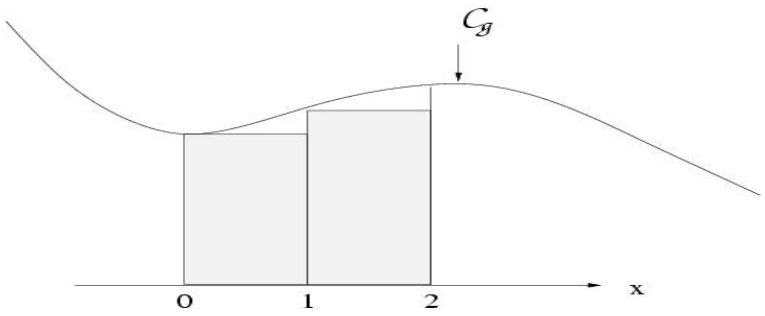


FIG. 2. L'aire du domaine limité par les droites $x = 0,\ x = 1$, l'axe Ox et $\mathcal{C}_g$ est approchée par l'aire du rectangle de base $[0,\ 1]$

Exercice 1. WIMS : Méthode du rectangle

2.3. **Méthode des rectangles à droite.**

$$\int_0^1 g(t)dt \approx g(1)$$

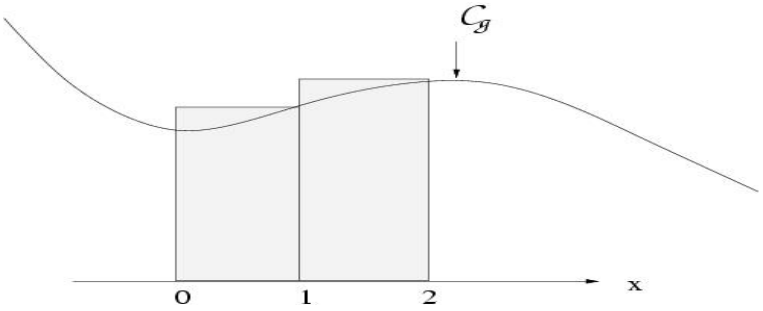


FIG. 3. L'aire du domaine limité par les droites $x = 0$, $x = 1$, l'axe Ox et $\mathcal{C}_g$ est approchée par l'aire du rectangle de base $[0,\ 1]$

2.4. **Méthode du point milieu.**

$$\int_0^1 g(t)dt \approx g(1/2)$$

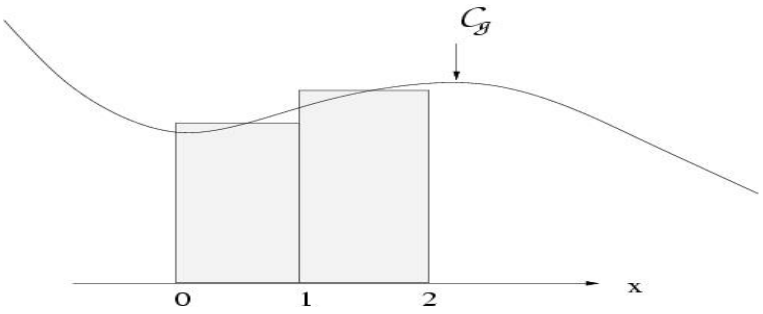


FIG. 4. L'aire du domaine limité par les droites $x = 0$, $x = 1$, l'axe Ox et $\mathcal{C}_g$ est approchée par l'aire du rectangle de base $[0,\ 1]$

Exercice 2. WIMS : Méthode du point milieu

2.5. **Méthode du trapèze.**

$$\int_0^1 g(t)dt \approx 1/2(g(0)+g(1))$$

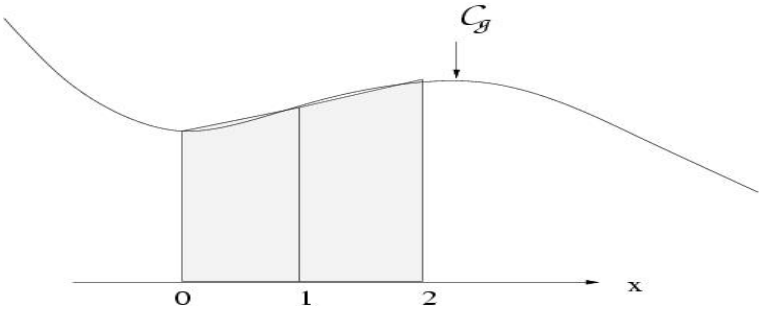


FIG. 5. L'aire du domaine limité par les droites $x = 0$, $x = 1$, l'axe Ox et $\mathcal{C}_g$ est approchée par l'aire du trapèze de base $[0,\ 1]$

2.6. **Méthode de Simpson.** Traçons la parabole passant par les trois points $(0,\ g(0))$, $(1/2,\ g(1/2))$, $(1,\ g(1))$. En approchant l'intégrale par l'aire sous la parabole, on obtient la *formule de Simpson* :

$$\int_0^1 g(t)dt \approx 1/6(g(0)+4g(1/2)+g(1))$$

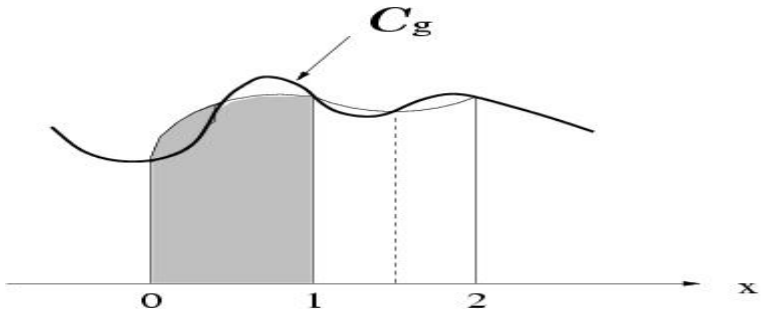


FIG. 6. L'aire du domaine limité par les droites $x = 0$, $x = 1$, l'axe Ox et $\mathcal{C}_g$ est approchée par l'aire grisée

2.7. **Méthode de Newton-Cotes.** On peut généraliser la méthode de Simpson en approchant l'intégrale par l'aire sous un polynôme de degré $s-1$ passant par les s points équidistants

$$\left(\left(\frac{i}{s-1}, g(\frac{i}{s-1})\right), i = 0, \cdots, s-1\right),$$

on obtient des formules de quadrature appelées formules de *Newton-Cotes* données par la définition.

Définition 2. Une formule de quadrature est dite *de Newton-Cotes* à *s étages* si elle est de la forme :

$$\int_0^1 g(t)dt \approx \sum_{i=1}^{s} b_i g(c_i). \tag{1}$$

Les c_i sont les *noeuds* de la formule de quadrature et les b_i sont les *poids*.

2.8. **Ordre.**

Définition 3. On dit que l'*ordre ordre d'une formule de quadrature* de la formule de quadrature 1 est p si elle est exacte pour tous les polynômes de degré inférieur ou égal à $p-1$, c'est-à-dire : pour g polynôme de degré $\leq p-1$,

$$\int_0^1 g(t)dt = \sum_{i=1}^{s} b_i g(c_i) \tag{2}$$

On voit que les formules du point milieu et des trapèzes sont d'ordre 2. La formule de Newton-Cotes à s étages a un ordre p supérieur ou égal à s.

Le tableau suivant résume l'ordre ainsi que les poids des différentes méthodes de quadrature pour $s \leq 7$.

s	ordre	b_1	b_2	b_3	b_4	b_5	b_6	b_7	nom
1	1	1							rectangle
1	2	1							pt. milieu
2	2	$\frac{1}{2}$	$\frac{1}{2}$						trapèze
3	4	$\frac{1}{6}$	$\frac{4}{6}$	$\frac{1}{6}$					Simpson
4	4	$\frac{1}{8}$	$\frac{3}{8}$	$\frac{3}{8}$	$\frac{1}{8}$				Newton
5	6	$\frac{7}{90}$	$\frac{32}{90}$	$\frac{12}{90}$	$\frac{32}{90}$	$\frac{7}{90}$			Boole
6	6	$\frac{19}{288}$	$\frac{75}{288}$	$\frac{50}{288}$	$\frac{50}{288}$	$\frac{75}{288}$	$\frac{19}{288}$		Boole
7	8	$\frac{41}{840}$	$\frac{216}{840}$	$\frac{27}{840}$	$\frac{272}{840}$	$\frac{27}{840}$	$\frac{216}{840}$	$\frac{41}{840}$	Weddle

Exercice 3. WIMS : Ordre d'une méthode de quadrature

2.8.1. *Condition nécessaire et suffisante.*

Théorème 2. *La formule de quadrature 1 est d'ordre p si et seulement si :*

$$\sum_{i=1}^{s} b_i c_i^{q-1} = \frac{1}{q}, \text{ pour } q = 1, \cdots, p \tag{3}$$

Preuve 1. *La nécessité de l'équivalence 3) est une conséquence de la formule 2) si l'on pose* $g(t) = t^{q-1}$. *Pour en montrer la suffisance, on utilise le fait qu'un polynôme de degré* $p-1$ *est une combinaison linéaire de* $1,\ t,\ \cdots,\ t^{p-1}$ *et que l'intégrale* $\int_0^1 g(t)\,dt$ *ainsi que l'expression* $\sum_{i=1}^{s} b_i g(c_i)$ *sont linéaires en g.*

Remarque 3. (1) En fixant les noeuds $c_1,\ c_2,\ \cdots,\ c_s$ (distincts), la condition 3 avec $p = s$ représente un système linéaire pour $b_1,\ b_2,\ \cdots,\ b_s$

$$\begin{pmatrix} 1 & 1 & \cdots & 1 \\ c_1 & c_2 & \cdots & c_s \\ \vdots & \vdots & & \vdots \\ c_1^{s-1} & c_2^{s-1} & \cdots & c_s^{s-1} \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_s \end{pmatrix} = \begin{pmatrix} 1 \\ \frac{1}{2} \\ \vdots \\ \frac{1}{s} \end{pmatrix} \tag{4}$$

Comme la matrice dans la formule 4 est inversible (matrice de Vandermonde), la résolution de ce système nous donne une formule de quadrature d'ordre $p = s$.

(2) Si l'on vérifie les conditions 3 pour la formule de Simpson, on fait une observation intéressante : par définition, il est évident que la condition 3 est satisfaite pour $q = 1,\ 2,\ 3$, mais on remarque qu'elle est aussi satisfaite pour $q = 4$. En effet :

$$\frac{1}{6}0^3 + \frac{4}{6}\left(\frac{1}{2}\right)^3 + \frac{1}{6}1^3 = \frac{1}{4}$$
$$\frac{1}{6}0^4 + \frac{4}{6}\left(\frac{1}{2}\right)^4 + \frac{1}{6}1^4 = \frac{5}{24} \neq \frac{1}{5}.$$

Elle est donc d'ordre 4. Par conséquent, elle n'est pas seulement exacte pour les polynômes de degré 2 mais aussi pour les polynômes de degré 3. Ceci est est une propriété qui peut être généralisée aux formules de quadrature *symétriques* (c'est-à-dire $c_i = 1 - c_{s+1-i},\ b_i = b_{s+1-i},\ \forall\ 1 \leq i \leq s$).

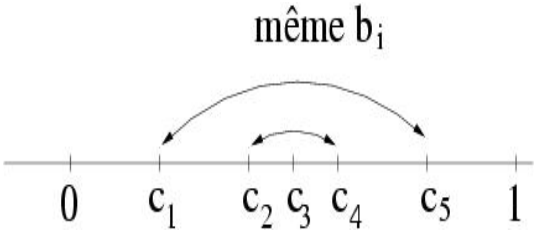


FIG. 7. Coefficients et noeuds d'une formule de quadrature symétrique

2.8.2. *Cas symétrique.*

Théorème 3. *Une formule de quadrature symétrique a toujours un ordre pair : si elle est exacte pour les polynômes de degré* $\leq 2m-2$, *elle est exacte pour les polynômes de degré* $\leq 2m-1$.

Preuve 2. *Chaque polynôme* $g(t)$ *de degré* $\leq 2m-1$ *peut être écrit sous la forme*

$$g(t) = c(t - \frac{1}{2})^{2m-1} + h(t)$$

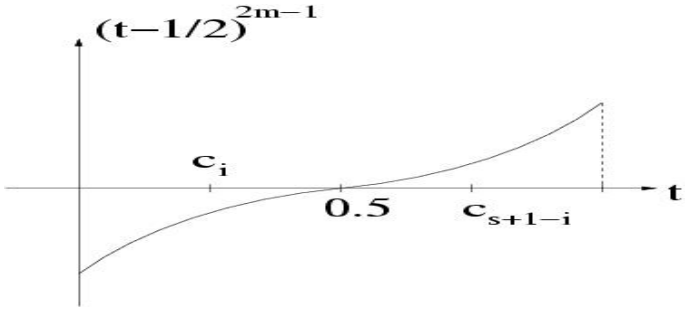
où $h(t)$ *est un polynôme de degré* $\leq 2m-2$ *et où c est une constante. Il suffit alors de montrer que la formule symétrique est exacte pour* $(t - \frac{1}{2})^{2m-1}$. *A cause de la symétrie de cette fonction, la valeur exacte vaut*

$$\int_0^1 (t - \frac{1}{2})^{2m-1} dt = 0$$

Pour une formule de quadrature symétrique on a

$$b_i(c_i - \frac{1}{2})^{2m-1} + b_{s+1-i}(c_{s+1-i} - \frac{1}{2})^{2m-1} = 0$$

Donc l'approximation numérique de $\int_0^1 (t - \frac{1}{2})^{2m-1} dt$ *est également nulle.*



3. Mise en oeuvre sur Matlab

3.1. **Notations.** Ici nous allons exécuter sur Matlab quelques méthodes de quadrature classiques pour approcher la valeur de l'intégrale

$$I_{exa} = \int_1^2 \frac{1}{t}\, dt = \log(2)$$

avec une subdivision de l'intervalle $[1,\ 2]$ correspondante à

$$\alpha_i = 1 + \frac{1}{N} i;\ 0 \leq i \leq N \text{ et } N = 4.$$

3.2. **Méthode des rectangles à gauche.** On note I_{rg} l'approximation de I_{exa} par la méthode des rectangles à gauche et E_{rg} l'erreur commise. Voici un programme Matlab qui calcule I_{rg} et E_{rg} :

Code Matlab 1.
```
%%%%%%%%%%%%%%%%%%%%%%%%%
%      Donnees
%%%%%%%%%%%%%%%%%%%%%%%%%
Iexa = log(2);
alpha = 1;
beta = 2;
N = 4;
h = (beta - alpha)/N;
x = [alpha:h:beta];
f = inline('1/x','x');
%%%%%%%%%%%%%%%%%%%%%%%%%
% Corps du programme
%%%%%%%%%%%%%%%%%%%%%%%%%
Irg = 0.0;
for i = 1:N
    Irg = Irg + h*f(x(i));
end
%%%%%%%%%%%%%%%%%%%%%%%%%
% Affichage des resultats
%%%%%%%%%%%%%%%%%%%%%%%%%
Irg
Erg = abs(Iexa - Irg)
%%%%%%%%%%%%%%%%%%%%%%%%%
```

Les résultats obtenus par ce programme sont :

```
Irg = 7.595238095e-01
Erg = 6.637662896e-02
```

3.3. **Méthode des trapèzes.** On note I_{tr} l'approximation de I_{exa} par la méthode des trapèzes et E_{tr} l'erreur commise. Voici un programme Matlab qui calcule I_{tr} et E_{tr} :

Code Matlab 2.
```
%%%%%%%%%%%%%%%%%%%%%%%%%
%      Donnees
%%%%%%%%%%%%%%%%%%%%%%%%%
Iexa = log(2);
alpha = 1;
beta = 2;
N = 4;
h = (beta - alpha)/N;
x = [alpha:h:beta];
f = inline('1/x','x');
%%%%%%%%%%%%%%%%%%%%%%%%%
% Corps du programme
%%%%%%%%%%%%%%%%%%%%%%%%%
Itr = 0.0;
for i = 1:N
    Itr = Itr + h*(0.5*f(x(i)) + 0.5*f(x(i+1)));
end
%%%%%%%%%%%%%%%%%%%%%%%%%
% Affichage des resultats
%%%%%%%%%%%%%%%%%%%%%%%%%
Itr
Etr = abs(Iexa - Itr)
```

Les résultats obtenus par ce programme sont :

```
Itr = 6.970238095e-01
Etr = 3.876628964e-03
```

3.4. **Méthode de Simpson.** On note I_{si} l'approximation de I_{exa} par la méthode de Simpson et E_{si} l'erreur commise. Voici un programme Matlab qui calcule I_{si} et E_{si} :

Code Matlab 3.
```
%%%%%%%%%%%%%%%%%%%%%%%%%
%      Donnees
%%%%%%%%%%%%%%%%%%%%%%%%%
Iexa = log(2);
alpha = 1;
beta = 2;
N = 4;
h = (beta - alpha)/N;
x = [alpha:h:beta];
f = inline('1/x','x');
%%%%%%%%%%%%%%%%%%%%%%%%%
% Corps du programme
%%%%%%%%%%%%%%%%%%%%%%%%%
Isi = 0.0;
for i = 1:N
Isi = Isi+h*(1/6*f(x(i))+2/3*f((x(i)+x(i+1))/2)+1/6*f(x(i+1)));
end
%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% Affichage des resultats
%%%%%%%%%%%%%%%%%%%%%%%%%
Isi
Es = abs(Iexa - Isi)
```

Les résultats obtenus par ce programme sont :

```
Isi = 6.931545307e-01
Esi = 7.350094585e-06
```

3.5. **Commentaires des résultats.** On voit bien que l'erreur absolue obtenue par la méthode de Simpson est beaucoup plus faible que celles obtenues par les deux autres. Ceci confirme la règle : **plus l'ordre de la méthode est grand, plus la précision est bonne**.

4. ETUDE DE L'ERREUR D'UNE MÉTHODE DE QUADRATURE

Pour étudier l'erreur commise en approchant une intégrale par l'une des formules de quadrature, nous commençons par une expérience *numérique* :

4.1. **Expérience numérique.**

4.1.1. *Nombre d'évaluation.* Prenons une fonction f définie sur $[a,\ b]$, subdivisons l'intervalle en plusieurs sous-intervalles équidistants ($h = \dfrac{b-a}{N}$) et appliquons l'une des formules de quadrature du paragraphe précédent. Ensuite, étudions l'erreur (en échelle logarithmique),

$$
\begin{aligned}
E(f) &= \int_a^b f(x)\,dx - \sum_{j=0}^{N-1} \int_{x_j}^{x_{j+1}} f(x)\,dx \\
&= \int_a^b f(x)\,dx - \sum_{j=0}^{N-1} h \int_0^1 f(x_j + th)\,dt \qquad (5) \\
&= \int_a^b f(x)\,dx - \sum_{j=0}^{N-1} h \sum_{i=1}^{s} b_i f(x_j + c_i h),
\end{aligned}
$$

en fonction du nombre d'évaluations f_e de la fonction f pour Newton-Cotes : f_e est défini par :

$$f_e = N(s-1)+1$$

Le nombre f_e repésente une mesure pour le **travail** effectué par l'ordinateur.

4.1.2. *Exemple.* Voici les résultats obtenus par les formules de Newton-Cotes (trapèzes, Simpson, Boole) pour l'intgrale

$$I_{exa} = \sin(2) = \int_0^2 \cos(x)\,dx$$

et $N = 1,\ 2,\ 4,\ 8,\ 16,\ \cdots$

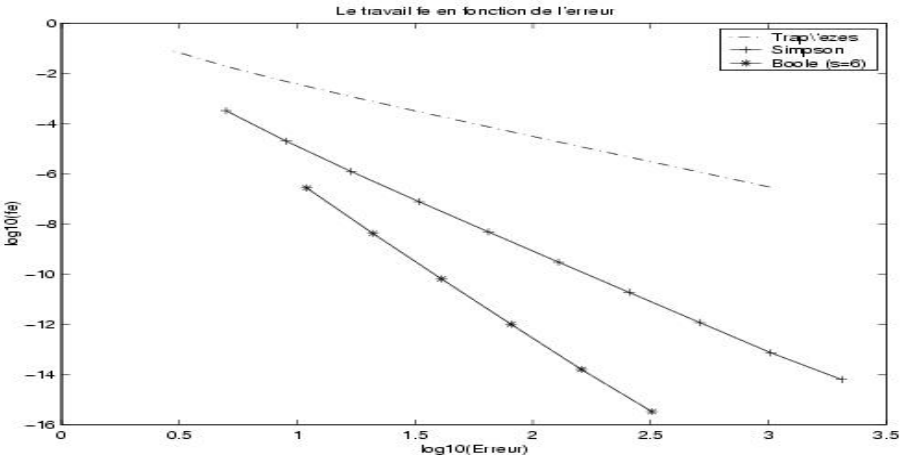
Code Matlab 4.

```
clear all;
Iexa = sin(2);
alpha = 0;
beta = 2;
f = inline('cos(x)','x');
%-------------------------
%-------------------------
% M\'ethode des trap\`ezes
%-------------------------
%-------------------------
s = 2;
```

```
for j = 1:1:10
%%%%%%%%%%%%%%%%%%%%%%%%
%      Donnees
%%%%%%%%%%%%%%%%%%%%%%%%
N = 2^j;
fetr(j) = log10(N*(s-1) +1);
h = (beta - alpha)/N;
x = [alpha:h:beta];
%%%%%%%%%%%%%%%%%%%%%%%%
% Corps du programme
%%%%%%%%%%%%%%%%%%%%%%%%
Itr=0.0;
for i = 1:N
Itr = Itr + h*(1/2*f(x(i)) + 1/2*f(x(i+1)));
end
%%%%%%%%%%%%%%%%%%%%%%%%
% Affichage des resultats
%%%%%%%%%%%%%%%%%%%%%%%%
Etr(j) = log10(abs(Iexa - Itr)) ;
end
%-------------------------
%-------------------------
% M\'ethode de Simpson
%-------------------------
%-------------------------
s = 3;
for j=1:1:10
%%%%%%%%%%%%%%%%%%%%%%%%
%      Donnees
%%%%%%%%%%%%%%%%%%%%%%%%
N = 2^j;
fesi(j) = log10(N*(s-1) +1);
h = (beta - alpha)/N;
x = [alpha:h:beta];
%%%%%%%%%%%%%%%%%%%%%%%%
% Corps du programme
%%%%%%%%%%%%%%%%%%%%%%%%
Isi = 0.0;
for i = 1:N
Isi = Isi+h*(1/6*f(x(i))+2/3*f((x(i)+x(i+1))/2)+1/6*f(x(i+1)));
end
%%%%%%%%%%%%%%%%%%%%%%%%
% Affichage des resultats
%%%%%%%%%%%%%%%%%%%%%%%%
Esi(j) = log10(abs(Iexa - Isi)) ;
end
%-------------------------
%-------------------------
% M\'ethode de Boole (s=6)
%-------------------------
%-------------------------
s = 6;
```

```
for j = 1:1:8
%%%%%%%%%%%%%%%%%%%%%%%%
%      Donnees
%%%%%%%%%%%%%%%%%%%%%%%%
N = 2^j;
febo(j) = log10(N*(s-1) +1);
h = (beta - alpha)/N;
x = [alpha:h:beta];
%%%%%%%%%%%%%%%%%%%%%%%%
% Corps du programme
%%%%%%%%%%%%%%%%%%%%%%%%
Ibo = 0.0;
for i = 1:N
Ibo = Ibo + h*(19/288*f(x(i)) + 75/288*f(x(i)+h/5) +
50/288*f(x(i)+(2*h/5)) + 50/288*f(x(i)+ (3*h/5)) +
75/288*f(x(i)+ (4*h/5)) + 19/288*f(x(i+1)));
end
%%%%%%%%%%%%%%%%%%%%%%%%
% Affichage des resultats
%%%%%%%%%%%%%%%%%%%%%%%%
Ebo(j) = log10(abs(Iexa - Ibo)) ;
plot(fetr, Etr, 'k-.', fesi, Esi, 'k-+', febo, Ebo, 'k-*')
legend('Trap\'ezes', 'Simpson', 'Boole (s=6)')
xlabel('log10(Erreur)');
ylabel('log10(fe)');
title('Le travail fe en fonction de l''erreur');
end
```

La figure ci-dessous montre donne les résultats pour $N = 1, 2, 4, 8, 16$.



4.1.3. *Interprétation des résultats.* Nous constatons que :

(1) $\log_{10}(f_e)$ dépend linéairement du nombre de chiffres exacts, donné par $-\log_{10}(E(\cos))$.

(2) La pente de chaque droite est $\frac{1}{p}$ où p est l'ordre de la méthode de quadrature.

(3) Pour un travail équivalent (même f_e), les formules avec un ordre élevé ont une meilleure précision.

4.1.4. *Justification des résultats.* Etudions d'abord l'erreur faite sur un sous-intervalle de longueur h.

$$\begin{aligned} E(f,\, x_0,\, h) &= \int_{x_0}^{x_0+h} f(x)\, dx - h\sum_{i=1}^{s} b_i f(x_0+c_i h) \\ &= h\left(\int_0^1 f(x_0+th)\, dt - \sum_{i=1}^{s} b_i f(x_0+c_i h)\right). \end{aligned} \tag{6}$$

On considère la formule de quadrature d'ordre p. En supposant que f est suffisament différentiable, on peut remplacer $f(x_0+th)$ et $f(x_0+c_ih)$ par les séries de Taylor au voisinage de x_0 :

$$\begin{aligned} E(f,\, x_0,\, h) &= \sum_{q\geq 0} \frac{h^{q+1}}{q!}\left(\int_0^1 t^q\, dt - \sum_{i=1}^{s} b_i c_i^q\right) f^{(q)}(x_0) \\ &= \frac{h^{p+1}}{p!}\left(\frac{1}{p+1} - \sum_{i=1}^{s} b_i c_i^p\right) f^{(p)}(x_0) + O(h^{p+2}) \end{aligned}$$

La constante définie par

$$C = \frac{1}{p!}\left(\frac{1}{p+1} - \sum_{i=1}^{s} b_i c_i^p\right) \tag{7}$$

s'appelle *constante de l'erreur.* Si on suppose que h est assez petit pour négliger $O(h^{p+2})$ devant Ch^{p+1}, on obtient :

$$\begin{aligned} E(f) &= \sum_{j=0}^{N-1} E(f,\, x_j,\, h) \approx Ch^p \sum_{j=0}^{N-1} h f^{(p)}(x_j) \approx Ch^p \int_a^b f^{(p)}(x)\, dx \\ &= Ch^p\left(f^{(p-1)}(b) - f^{(p-1)}(a)\right) \end{aligned}$$

Cette formule nous permet de mieux comprendre les résultats de la figure précédente. En effet, on peut écrire $E(f) \approx C_1 h^p$ et $fe \approx \dfrac{C_2}{h}$. Par conséquent,

$$-\log_{10}(E(f)) \approx -\log_{10}(C_1) - p\log_{10}(h) \approx Constante + p\log_{10}(fe).$$

Ceci montre la dépendance **linéaire** entre $\log_{10}(fe)$ et $\log_{10}(E(f))$ et le fait que la pente soit de $\dfrac{1}{p}$.

4.2. **Estimation rigoureuse de l'erreur.**

4.2.1. *Noyau de Peano.* Dans ce paragraphe on s'occupe de l'estimation exacte de l'erreur d'une formule de quadrature en vue de démontrer les théorèmes de convergence et assurer une certaine précision du résultat numérique.

Théorème 4 (et Définition)**.** *Soit une formule de quadrature d'ordre p et un entier k vérifiant $k \leq p$. Considérons une fonction $f : [x_0,\, x_0+h] \longrightarrow \mathbb{R}$ de classe $\mathcal{C}^k$, alors l'erreur $E(f,\, x_0,\, h)$ définie par la formule 6 vérifie :*

$$E(f,\, x_0,\, h) = h^{k+1}\int_0^1 N_k(s) f^{(k)}(x_0+s\,h)\, ds$$

où N_k est le noyau de Peano , défini par :

$$N_k(s) = \frac{(1-s)^k}{k!} - \sum_{i=1}^{s} b_i \frac{(c_i-s)_+^{k-1}}{(k-1)!} \quad où \;\; r_+^{k-1} = \begin{cases} r^{k-1} & si \quad r>0 \\ \\ 0 & si \quad r\leq 0 \end{cases}$$

Preuve 3. *La formule de Taylor avec reste intégral appliquée à f au point x_0 donne :*

$$f(x_0+t\,h)=\sum_{j=0}^{k-1}\frac{(t\,h)^j}{j!}f^{(j)}(x_0)+h^k\int_0^t\frac{(t-s)^{k-1}}{(k-1)!}f^{(k)}(x_0+s\,h)\,ds$$

En combinant cette dernière formule avec la formule 6 et en utilisant le fait que

$$\int_0^t(t-s)^{k-1}g(s)\,ds=\int_0^1(t-s)_+^{k-1}g(s)\,ds$$

et en remarquant que la partie polynomiale dans l'avant-dernière équation ne contribue pas à l'erreur (à cause que $p\geq k$), nous obtenons :

$$E(f,\,x_0,\,h)=h^{k+1}\int_0^1\left(\int_0^1\frac{(t-s)_+^{k-1}}{(k-1)!}\,dt-\sum_{i=1}^s b_i\frac{(c_i-s)_+^{k-1}}{(k-1)!}\right)f^{(k)}(x_0+s\,h)\,ds.$$

Une évaluation de l'intégrale intérieure donne le résultat.

Remarque 4. Pour une formule de quadrature d'ordre p et un entier k vérifiant $1\leq k\leq p$ on a :

$$\int_0^1 N_p(s)\,ds=\frac{1}{p!}\left(\frac{1}{p+1}-\sum_{i=1}^s b_i c_i^p\right)=C$$

où C est la constante de l'erreur définie par la formule 7.

Exemple 1. Les noyaux de Peano pour la méthode du point milieu sont donnés par :

$$N_1(s)=\begin{cases}-s & si \quad s<\frac{1}{2}\\ 1-s & si \quad s\geq\frac{1}{2}\end{cases}\qquad N_2(s)=\begin{cases}\dfrac{s^2}{2} & si \quad s\leq\frac{1}{2}\\ \dfrac{(1-s)^2}{2} & si \quad s\geq\frac{1}{2}\end{cases}$$

4.2.2. *Majoration de l'erreur.* Nous sommes maintenant en mesure d'estimer l'erreur commise pour l'intervalle $[a,\,b]$ tout entier et ceci pour une subdivision arbitraire $h_j=x_{j+1}-x_j$. Rappelons que, comme dans la formule 5, l'erreur est donnée par :

$$E(f)=\int_a^b f(x)\,dx-\sum_{j=0}^{N-1}h_j\sum_{i=1}^s b_i f(x_j+c_i h_j). \tag{8}$$

On a alors le théorème suivant :

Théorème 5. *Soit une formule de quadrature d'ordre p et un entier k vérifiant $k\leq p$. Considérons une fonction $f:\,[a,\,b]\longrightarrow\mathbb{R}$ de classe $\mathcal{C}^k$. Alors l'erreur $E(f)$ définie par la formule 8 vérifie l'estimation suivante :*

$$|E(f)|\leq h^k(b-a)\int_0^1|N_k(s)|\,ds\max_{x\in[a,\,b]}\left|f^{(k)}(x)\right| \tag{9}$$

où $h=\max_j(h_j)$.

Preuve 4. *La formule 5 donne :*

$$\begin{aligned}E(f,\,x_0,\,h) &\leq h^{k+1}\int_0^1|N_k(s)||f^{(k)}(x_0+s\,h)|\,ds\\ &\leq h^{k+1}\int_0^1|N_k(s)|\,ds\max_{x\in[x_0,\,x_0+h]}|f^{(k)}(x)|.\end{aligned} \tag{10}$$

Comme l'erreur 8 est la somme des erreurs sur les sous-intervalles de la subdivision, nous obtenons :

$$\begin{aligned} |E(f)| \leq \sum_{j=0}^{N-1} |E(f,\, x_j,\, h_j)| &\leq \sum_{j=0}^{N-1} h_j^{k+1} \int_0^1 |N_k(s)|\, ds \max_{x\in[x_j,\, x_{j+1}]} \left| f^{(k)}(x) \right| \\ &\leq \sum_{j=0}^{N-1} h^k\, h_j \int_0^1 |N_k(s)|\, ds \max_{x\in[a,\, b]} \left| f^{(k)}(x) \right| \end{aligned}$$

et puisque $\sum_{j=0}^{N-1} h_j = b-a$, *on obtient l'équation 9.*

Exemple 2. Pour la formule du point milieu, on a :

$$|E(f)| \leq h^2(b-a)\frac{1}{24} \max_{x\in[a,\, b]} \left| f''(x) \right|.$$

Pour la formule des trapèzes :

$$|E(f)| \leq h^2(b-a)\frac{1}{12} \max_{x\in[a,\, b]} \left| f''(x) \right|.$$

Pour la formule de Simpson :

$$|E(f)| \leq h^4(b-a)\frac{1}{2880} \max_{x\in[a,\, b]} \left| f^{(4)}(x) \right|.$$

Pour la formule de Newton-Cotes ($s=5$) :

$$|E(f)| \leq h^6(b-a)\frac{1}{1935360} \max_{x\in[a,\, b]} \left| f^{(6)}(x) \right|.$$

Remarque 5. Le calcul de $\int_0^1 |N_p(s)|\, ds$ pour ces formules n'est pas difficile. Considérons par exemple la formule de Newton-Cotes avec $s=5$. Nous constatons que $N_6(s)$ ne change pas de signe sur $[0,\, 1]$ et en utilisant la remarque 4, nous obtenons :

$$\begin{aligned} \int_0^1 |N_6(s)|\, ds &= \int_0^1 N_6(s)\, ds \\ &= \frac{1}{6!}\left| \frac{1}{7} - \left(\frac{32}{90}\left(\frac{1}{4}\right)^6 + \frac{12}{90}\left(\frac{1}{2}\right)^6 + \frac{32}{90}\left(\frac{3}{4}\right)^6 + \frac{7}{90}1^6 \right) \right| \\ &= \frac{1}{1935360}. \end{aligned}$$

Exercice 4. WIMS : Noyau de Peano

5. Exemples de calcul numérique de l'ordre

5.1. **Préliminaires.** Ici nous allons vérifier à l'aide de Matlab l'ordre de quelques méthodes de quadrature déjà étudiées précédemment pour approcher la valeur de l'intégrale

$$I_{exa} = \int_\alpha^\beta f(t)\, dt$$

avec

$$f(t) = \frac{1}{t},\ \alpha = 1,\ \beta = 2$$

et une subdivision de plus en plus fine de l'intervalle $[\alpha,\, \beta]$ correspondante à

$$\alpha_i = \alpha + \frac{\alpha-\beta}{N} i;\ 0 \leq i \leq N,\ \text{ et } N = 2^j;\ 1 \leq j \leq 20.$$

5.2. **Méthode des rectangles à gauche.** On note I_{rg} l'approximation de I_{exa} par la méthode des rectangles à gauche et E_{rg} l'erreur commise. On affiche les valeurs de j, I_{rg}, E_{rg}, E_{rg}/h, et E_{rg}/h^2.

Code Matlab 5.
```
clear all;
fid = 1;
fmt = '%% %10d  %20.9e     %20.9e   %20.9e     %20.9e \n';
for j = 1:2:17
%%%%%%%%%%%%%%%%%%%%%%%%
%      Donnees
%%%%%%%%%%%%%%%%%%%%%%%%
N = 2^j;
Iexa = log(2);
alpha = 1;
beta = 2;
h = (beta - alpha)/N;
x = [alpha:h:beta];
f = inline('1/x','x');
%%%%%%%%%%%%%%%%%%%%%%%%
% Corps du programme
%%%%%%%%%%%%%%%%%%%%%%%%
Irg = 0.0;
for i = 1:N
Irg = Irg + h*f(x(i));
end
%%%%%%%%%%%%%%%%%%%%%%%%
% Affichage des resultats
%%%%%%%%%%%%%%%%%%%%%%%%
%Irg
Erg = abs(Iexa - Irg) ;
Erg1 = abs(Iexa - Irg)/h ;
Erg2 = abs(Iexa - Irg)/h^2 ;
%%%%%%%%%%%%%%%%%%%%%%%%
fprintf(fid, fmt, j, Irg, Erg, Erg1, Erg2);
end
```

Les résultats obtenus par ce programme sont :

```
j         Irg              Erg               Erg/h           Erg/h^2

1    8.333333333e-01   1.401861528e-01   2.803723055e-01  5.607446111e-01
3    7.253718504e-01   3.222466981e-02   2.577973585e-01  2.062378868e+00
5    7.010207083e-01   7.873527709e-03   2.519528867e-01  8.062492374e+00
7    6.951041202e-01   1.956939668e-03   2.504882775e-01  3.206249952e+01
9    6.936357002e-01   4.885196685e-04   2.501220703e-01  1.280625000e+02
11   6.932692658e-01   1.220852137e-04   2.500305176e-01  5.120625000e+02
13   6.931776991e-01   3.051850945e-05   2.500076294e-01  2.048062500e+03
15   6.931548100e-01   7.629452736e-06   2.500019072e-01  8.192062496e+03
17   6.931490879e-01   1.907352286e-06   2.500004788e-01  3.276806276e+04
```

Commentaires : On constate que E_{rg}/h se stabilise autour de 2.5e-01 alors que E_{rg}/h^2 explose au fur et à mesure que j augmente (les subdivisions de plus en plus fines). Ceci confirme le fait que cette méthode est d'ordre 1.

5.3. **Méthode des trapèzes.** On note I_{tr} l'approximation de I_{exa} par la méthode des trapèzes et E_{tr} l'erreur commise. On affiche les valeurs de j, I_{tr}, E_{tr}, E_{tr}/h, E_{tr}/h^2, et E_{tr}/h^3.

Code Matlab 6.
```
clear all;
fid = 1;
fmt = '%% %10d  %12.9e     %12.9e   %12.9e   %12.9e     %12.9e \n';
for j = 1:2:17
%%%%%%%%%%%%%%%%%%%%%%%%
%      Donnees
%%%%%%%%%%%%%%%%%%%%%%%%
N = 2^j;
Iexa = log(2);
alpha = 1;
beta = 2;
h = (beta - alpha)/N;
x = [alpha:h:beta];
f = inline('1/x','x');
%%%%%%%%%%%%%%%%%%%%%%%%
% Corps du programme
%%%%%%%%%%%%%%%%%%%%%%%%
Itr = 0.0;
for i = 1:N
Itr = Itr + h*(0.5*f(x(i)) + 0.5*f(x(i+1)));
end
%%%%%%%%%%%%%%%%%%%%%%%%
% Affichage des resultats
%%%%%%%%%%%%%%%%%%%%%%%%
Etr = abs(Iexa - Itr) ;
Etr1 = abs(Iexa - Itr)/h ;
Etr2 = abs(Iexa - Itr)/h^2 ;
Etr3 = abs(Iexa - Itr)/h^3 ;
%%%%%%%%%%%%%%%%%%%%%%%%
fprintf(fid, fmt, j, Itr, Etr, Etr1 , Etr2, Etr3);
end
```

Les résultats obtenus par ce programme sont :

```
j       Itr            Etr            Etr/h         Etr/h^2      Etr/h^3

1    7.08333e-01    1.51862e-02    3.03723e-02    6.07446e-02   1.21489e-01
3    6.94122e-01    9.74670e-04    7.79736e-03    6.23789e-02   4.99031e-01
5    6.93208e-01    6.10277e-05    1.95289e-03    6.24924e-02   1.99976e+00
7    6.93151e-01    3.81467e-06    4.88278e-04    6.24995e-02   7.99994e+00
9    6.93147e-01    2.38418e-07    1.22070e-04    6.25000e-02   3.20000e+01
11   6.93147e-01    1.49012e-08    3.05176e-05    6.25000e-02   1.28000e+02
13   6.93147e-01    9.31321e-10    7.62938e-06    6.24999e-02   5.11999e+02
15   6.93147e-01    5.82108e-11    1.90745e-06    6.25033e-02   2.04811e+03
17   6.93147e-01    3.63654e-12    4.76648e-07    6.24752e-02   8.18875e+03
```

Commentaires : On constate que E_{tr}/h^2 se stabilise autour de 6.25e-02 alors que E_{rg}/h^3 explose au fur et à mesure que j augmente. Ceci confirme le fait que cette méthode est d'ordre 2.

5.4. **Méthode de Simpson.** On note I_{si} l'approximation de I_{exa} par la méthode de Simpson et E_{si} l'erreur commise. On affiche les valeurs de j, I_{si}, E_{si}, E_{si}/h^3, E_{si}/h^4, et E_{si}/h^5.

Code Matlab 7.
```
clear all;
fid = 1;
fmt = '%% %3d  %12.5e  %12.5e  %12.5e  %12.5e  %12.5e \n';
for j = 1:1:10
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%
%      Donnees
%%%%%%%%%%%%%%%%%%%%%%%%%
N = 2^j;
Iexa = log(2);
alpha = 1;
beta = 2;
h = (beta - alpha)/N;
x = [alpha:h:beta];
f = inline('1/x','x');
%%%%%%%%%%%%%%%%%%%%%%%%%
% Corps du programme
%%%%%%%%%%%%%%%%%%%%%%%%%
Isi = 0.0;
for i = 1:N
Isi = Isi + h*(1/6*f(x(i)) + 2/3*f((x(i) + x(i+1))/2) + 1/6*f(x(i+1)));
end
%%%%%%%%%%%%%%%%%%%%%%%%%
% Affichage des resultats
%%%%%%%%%%%%%%%%%%%%%%%%%
Esi = abs(Iexa - Isi) ;
Esi3 = abs(Iexa - Isi)/h^3 ;
Esi4 = abs(Iexa - Isi)/h^4 ;
Esi5 = abs(Iexa - Isi)/h^5 ;
%%%%%%%%%%%%%%%%%%%%%%%%%
fprintf(fid, fmt, j, Isi, Esi, Esi3 , Esi4, Esi5);
end
```

Les résultats obtenus par ce programme sont :

```
j       Isi           Esi          Esi/h^3       Esi/h^4      Esi/h^5

1    6.93254e-01   1.06788e-04   8.54302e-04   1.70860e-03   3.41721e-03
2    6.93155e-01   7.35009e-06   4.70406e-04   1.88162e-03   7.52650e-03
3    6.93148e-01   4.72259e-07   2.41797e-04   1.93437e-03   1.54750e-02
4    6.93147e-01   2.97299e-08   1.21774e-04   1.94838e-03   3.11740e-02
5    6.93147e-01   1.86151e-09   6.09979e-05   1.95193e-03   6.24619e-02
6    6.93147e-01   1.16398e-10   3.05130e-05   1.95283e-03   1.24981e-01
7    6.93147e-01   7.27574e-12   1.52583e-05   1.95307e-03   2.49992e-01
8    6.93147e-01   4.54081e-13   7.61822e-06   1.95026e-03   4.99268e-01
9    6.93147e-01   2.80886e-14   3.76999e-06   1.93024e-03   9.88281e-01
10   6.93147e-01   2.66454e-15   2.86102e-06   2.92969e-03   3.00000e+00
```

Commentaires : On constate que E_{si}/h^4 se stabilise autour de 1.95e-03 alors que E_{si}/h^5 explose au fur et à mesure que j augmente. Ceci confirme le fait que cette méthode est d'ordre 4.

6. Bibliographie

(1) Philipe G. Ciarlet. *Introduction à l'analyse numérique et à l'optimisation*. Dunod 1990.

(2) Jean-Pierre Demailly. *Analyse numérique et équations différentielles*. Presses Universitaires de Grenoble, 1996.

(3) Ernst Hairer. *Introduction à l'analyse numérique*. Université de Genève, section mathématiques, case postale 240. Octobre 2001.

7. EXERCICES

Exercice 5. Soit $f : [-1,1] \longrightarrow \mathbb{R}$ une fonction de classe $\mathcal{C}^2$. On considère la méthode d'intégration numérique approchée donnée par

$$\int_{-1}^{1} f(x)\,dx \simeq f(-w) + f(w), \text{ avec } w \in [0,\ 1]$$

(1) Calculer l'ordre de cette méthode en fonction de w.

(2) On se place dans le cas où cette méthode est d'ordre 1.

(a) Calculer le noyau de Peano $G_1(t)$ et tracer le graphe de G_1 pour $w = \dfrac{5}{8}$. Pour quelles valeurs de w le noyau G_1 est-il de signe constant ?

(b) Montrer que l'erreur $E(f) = \displaystyle\int_{-1}^{1} f(x)\,dx - f(-w) - f(w)$ vérifie la majoration

$$|E(f)| \leq C(w) \sup_{\xi\in[-1,1]} (|f''(\xi)|)$$

où $C(w)$ est une constante dont on déterminera la valeur optimale

(i) lorsque G_1 est de signe constant.

(ii) lorsque $w = \dfrac{5}{8}$.

Exercice 6. On rappelle que par construction, les méthodes de Newton-Cotes sont les formules de quadratures élémentaires de type

$$\int_0^1 P(x)\,dx = \sum_{i=0}^{n} \lambda_i P(x_i)$$

telles que les noeuds $x_0 < x_1 < \cdots < x_{n-1} < x_n$ soient équidistants et centrés dans l'intervalle $[0,\ 1]$, les λ_i étant choisis de telle façon que ces formules soient exactes pour tout polynôme P de degré inférieur ou égale à n. Montrer que si n est pair ces formules sont aussi exactes pour les polynômes de degré $n+1$.
(Indication : on pourra remarquer que $\lambda_i = \lambda_{n-i}$ et en tirer les conséquences pour les polynômes impairs.)

Exercice 7. Construire les formules d'intégration numérique suivantes :

$$\int_{-1}^{1} \varphi(s)ds \simeq \varphi(-1/3) + \varphi(1/3), \text{ exacte si } \varphi \in \mathcal{P}_1;$$

$$\int_{-1}^{1} \varphi(s)\,dx \simeq \frac{2}{3}(2\varphi(-1/2) - \varphi(0) + 2\varphi(1/2)), \text{ exacte si } \varphi \in \mathcal{P}_3;$$

$$\int_{-1}^{1} \varphi(s)ds \simeq \frac{1}{12}(11\varphi(-3/5) + \varphi(-1/5) + \varphi(1/5) + 11\varphi(3/5)), \text{ exacte si } \varphi \in \mathcal{P}_3$$

Déterminer leur noyau de Peano et en déduire l'erreur commise.

Exercice 8. (1) Montrer que

$$\frac{1}{e^x - 1} = \frac{b_0}{x} + b_1 + \frac{b_2}{2!}x + \cdots + \frac{b_{2n}}{(2n)!}x^{2n-1} + o(x^{2n})$$

(Indication : appliquer la formule d'Euler-MacLaurin à e^{-x} entre 0 et 1.)

(2) Montrer que si $f \in \mathcal{C}^\infty(\mathbb{R})$ est une fonction périodique de période $b-a$, alors

$$\left| T_h(f) - \int_a^b f(x)\,dx \right| \leq C_n(f,\ a,\ b)h^n$$

$\forall\, n \in \mathbb{N}$ et où $T_h(f)$ représente l'évaluation de la formule des trapèzes de pas h pour f sur $[a,\ b]$.

Exercice 9. Soient x_1 et x_2 deux points de $[-1,\ 1]$ et λ_1 et $\lambda_2 \in \mathbb{R}$. On considére la formule d'itégration suivante :

$$\int_{-1}^{1} f(x)\,dx \approx \lambda_1 f(x_1) + \lambda_2 f(x_2)$$

Quelles conditions doivent vérifier x_1, x_2, λ_1 et λ_2 pour que cette formule soit exacte pour

(1) les fonctions constantes ?
(2) les fonctions affines ?
(3) les polynômes de degré inférieur ou égale à 2 ?

Exercice 10. On considère la formule d'intégration suivante :

$$\int_{-1}^{1} f(x)\,dx \approx k f(\alpha) + f(\beta) \quad (1)$$

(1) Déterminer les valeurs de k, α et β pour que (1) soit exacte sur $\mathcal{P}_2$.
(2) En déduire les valeurs de k, α et β pour que (1) soit d'ordre le plus élevé possible.
(3) (a) Calculer le noyau de Peano dans le cas où (1) est d'ordre 3 et vérifier que ce noyau est une fonction paire.
(b) En déduire qu'il existe $\xi \in]-1,\ 1[$ tel que $E(f) = \dfrac{1}{135} f^{(4)}(\xi)$ où $E(f)$ est l'erreur d'intégration.
(c) Donner la formule d'intégration relative à (1) sur $[a,\ b]$.
(d) Estimer l'erreur d'intégration obtenue par la méthode composée associée à (1) sur $[a,\ b]$ avec un pas constant $h = \dfrac{b-a}{n}$.

Exercice 11. On considére la formule d'intégration suivante :

$$\int_0^{\pi} f(x)\sin(2x)\,dx \approx a_1 f(x_1) + a_2 f(x_2)$$

(1) Déterminer $a_1,\ a_2,\ x_1$ et x_2 de sorte que cette formule soit exacte sur $\mathcal{P}_3$.
(2) Calculer alors l'ordre de cette méthode.

Exercice 12. (1) Soit $f(x) = \dfrac{1}{1+x^2}$. Montrer qu'il existe un polynôme P unique de degré 2 vérifiant :

$$P(0) = f(0),\ P(1) = f(1) \text{ et } P\left(\frac{1}{2}\right) = f\left(\frac{1}{2}\right).$$

Déterminer $\displaystyle\int_0^1 P(t)dt$ et la comparer à $\displaystyle\int_0^1 f(t)dt$.

(2) On considére $\displaystyle\int_0^1 \sin\frac{\pi t^2}{2}dt$. On veut calculer cette intégrale avec une erreur inférieur à 10^{-3}.
(a) Déterminer le pas h nécessaire pour la méthode des trapèzes.
(b) Déterminer le pas h nécessaire pour la méthode de Simpson.

Exercice 13. (1) Trouver l'ordre des formule de : rectangle, trapèze et Simpson.

(2) Pour $f \in \mathcal{C}^4([-1,1])$. On pose

$$E(f) = \int_{-1}^{1} f(x)\,dx - \frac{2}{6}[f(-1) + 4f(0) + f(1)] \qquad \text{et} \qquad K_3(t) = E(x \longrightarrow (x-t)_+^3)$$

Montrer que $K_3(t) = \displaystyle\int_t^1 (x-t)^3\,dx - 2\left(\frac{2}{3}t_-^3 + \frac{1}{6}(1-t)^3\right)$.

(3) En déduire $K_3(t) = -\dfrac{1}{12}(1-|t|)^3(1+|t|)$. Calculer par deux méthodes différentes $\displaystyle\int_{-1}^1 K_3(t)dt$.

(4) Enoncer le thérème de Peano et montrer que si $f \in \mathcal{C}^4([a,b])$ alors on a :

$$\left|\int_a^b f(t)dt - \frac{h}{6}\left(f(a) + 2\sum_{k=1}^{n-1} f(a_k) + 4\sum_{k=0}^{n-1} f\left(a_k + \frac{b-a}{2n}\right) + f(b)\right)\right| \leq$$

$$\frac{(b-a)^5}{2880n^4} \sup_{[a,b]} |f^4(t)|$$