Contents

Chapter 1

CM1

1.1 Modèles Discrets

Les modèles discrets considèrent que les changements de population se produisent à intervalles de temps distincts.

1.1.1 Equation Générale des Modèles Discrets

Considérons $N(t)$ comme la population d'individus à l'instant t. L'équation générale d'un modèle discret est donnée par la variation de la population entre deux instants discrets t et $t + \Delta t$:

$N(t+\Delta t) - N(t) =$ nombre de naissances − nombre de décès + nombre d'immigrations − nombre d'émigrations

En termes de taux, nous pouvons écrire:

$$N(t + \Delta t) - N(t) = n - m + i - e$$

où:

- n représente le nombre de naissances pendant l'intervalle Δt.
- m représente le nombre de décès pendant l'intervalle Δt.
- i représente le nombre d'immigrations pendant l'intervalle Δt.
- e représente le nombre d'émigrations pendant l'intervalle Δt.

1.1.2 Modèle de Croissance Géométrique

Le modèle de croissance géométrique est un modèle discret simple qui décrit la croissance d'une population dans des conditions idéales, où les ressources sont illimitées.

Hypothèses

- **Solde migratoire nul**: On suppose que le nombre d'immigrations est égal au nombre d'émigrations, donc $i - e = 0$.
- **Croissance proportionnelle à la taille de la population**: Le nombre de naissances est proportionnel à la taille de la population, avec un taux de natalité λ, et le nombre de décès est proportionnel à la taille de la population, avec un taux de mortalité μ. Ainsi, pendant l'intervalle Δt:
  - Nombre de naissances: $n = \lambda \Delta t N(t)$
  - Nombre de décès: $m = \mu \Delta t N(t)$

Équation et Solution

En posant $N_n = N(t_n)$ où $t_n = n\Delta t$, l'équation du modèle devient:

$$N_{n+1} - N_n = \lambda \Delta t N_n - \mu \Delta t N_n$$

Soit en posant $z = \lambda - \mu$, le taux de croissance:

$$N_{n+1} - N_n = z\Delta t N_n$$

$$N_{n+1} = N_n + z\Delta t N_n = (1 + z\Delta t)N_n$$

En définissant le taux de croissance par unité de temps $c = z\Delta t$, on a:

$$N_{n+1} = (1 + c)N_n$$

La solution de cette équation de récurrence, avec condition initiale N_0 (taille initiale de la population), est:

$$N_n = (1 + c)^n N_0 = (1 + z\Delta t)^n N_0$$

Visualisation

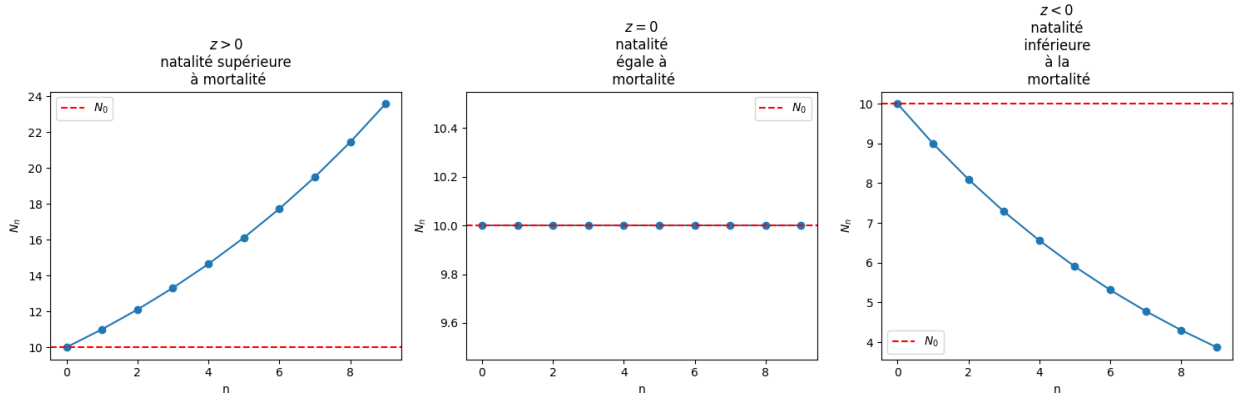

Figure 1.1: Visualisation du modèle de croissance géométrique pour différents taux de croissance z.

Propriétés du Modèle Géométrique

- **Tendance à l'infini pour** $z > 0$: Lorsque $z = \lambda - \mu > 0$, la population croît exponentiellement et tend vers l'infini lorsque $n \to \infty$. La solution $N(t) = N_0 e^{zt}$ est une approximation continue pour de petits Δt.
- **Croissance indéfinie pour** $z > 0$: Si $z > 0$, la population croît indéfiniment.
- **Extinction pour** $z < 0$: Si $z < 0$, la population décroît exponentiellement et tend vers l'extinction.
- **Population constante pour** $z = 0$: Si $z = 0$, la population reste constante au fil du temps, $N_n = N_0$ pour tout n.

Inconvénients du Modèle Géométrique

- **Croissance infinie irréaliste**: Une croissance infinie n'est pas réaliste dans le monde réel car les ressources sont limitées.
- **Approximation de partie entière**: Pour être rigoureux, on devrait écrire $E(\lambda \Delta t N_n)$ et $E(\mu \Delta t N_n)$ pour tenir compte du fait que le nombre d'individus doit être un entier, où $E(x)$ désigne la partie entière de x.

1.1.3 Modèle de croissance logistique discret

Le modèle de croissance logistique discret sera traité dans un exercice ultérieur.

1.2 Modèles Continus

Les modèles continus considèrent que les changements de population se produisent de manière continue dans le temps.

1.2.1 Motivation pour les Modèles Continus

Remark 1.2.1. L'utilisation de modèles continus est motivée par le fait que l'observation des populations sur des intervalles de temps très courts (Δt proche de 0) fournit beaucoup plus d'informations sur la dynamique de la population.

1.2.2 Modèle de Malthus

Le modèle de Malthus est le modèle continu le plus simple de croissance de population. Il est obtenu en passant à la limite du modèle géométrique lorsque $\Delta t \to 0$.

Hypothèses

- **Solde migratoire nul**: Comme pour le modèle géométrique, on suppose un solde migratoire nul.
- **Vitesses de natalité et de mortalité proportionnelles à la population**: On suppose que la vitesse de natalité et la vitesse de mortalité sont proportionnelles à la taille de la population à l'instant t.
  - Vitesse de natalité: $n(t) = \lambda N(t)$
  - Vitesse de mortalité: $m(t) = \mu N(t)$

Équation et Solution

Proposition 1.2.2. En reprenant l'équation de variation et en considérant les vitesses instantanées, l'équation différentielle du modèle de Malthus est:

$$N'(t) = \lim_{\Delta t \to 0} \frac{N(t+\Delta t) - N(t)}{\Delta t} = n(t) - m(t) = \lambda N(t) - \mu N(t)$$

Soit:

$$N'(t) = (\lambda - \mu) N(t)$$

En posant $z = \lambda - \mu$, on obtient:

$$N'(t) = zN(t)$$

Avec la condition initiale $N(0) = N_0$, la solution de cette équation différentielle est:

$$N(t) = N_0 e^{zt} = N_0 e^{(\lambda - \mu)t}$$

Propriétés du Modèle de Malthus

- Similaire au modèle géométrique en termes de comportement qualitatif, mais décrit la croissance de manière continue.
- **Croissance exponentielle pour** $z > 0$: Si $z = \lambda - \mu > 0$, la population croît exponentiellement.
- **Population constante pour** $z = 0$: Si $z = \lambda - \mu = 0$, la population reste constante.

- **Décroissance exponentielle pour** $z < 0$: Si $z = \lambda - \mu < 0$, la population décroît exponentiellement vers zéro.

Inconvénients du Modèle de Malthus

- **Croissance exponentielle irréaliste**: Comme le modèle géométrique, le modèle de Malthus prédit une croissance exponentielle infinie, ce qui n'est pas réaliste à long terme en raison de la limitation des ressources.
- **Ne prend pas en compte la limitation des ressources et l'interaction avec l'environnement**.

1.2.3 Modèle de Verhulst (ou Logistique)

Le modèle de Verhulst est une amélioration du modèle de Malthus qui prend en compte la limitation des ressources.

Idée Centrale

Definition 1.2.3. Limiter la croissance de la population à un seuil maximal K, appelé *capacité biotique* ou *capacité de charge* du milieu.

Hypothèses

- **Solde migratoire nul**.
- **Taux de natalité fonction affine décroissante de la population**: Le taux de natalité diminue à mesure que la population approche de la capacité biotique K. On le modélise par une fonction affine décroissante: $\lambda = \lambda_0(1 - \frac{N(t)}{K})$, où λ_0 est le taux de natalité maximal (lorsque $N(t)$ est très petit).
- **Taux de mortalité fonction affine croissante de la population**: Le taux de mortalité augmente à mesure que la population approche de K. On le modélise par une fonction affine croissante: $\mu = \mu_0(1 + \frac{N(t)}{K})$, où μ_0 est le taux de mortalité minimal (lorsque $N(t)$ est très petit). Pour simplifier, on prend souvent μ constant. Dans les notes, il est considéré comme une fonction affine croissante $\mu = -\mu_1(1-\frac{N(t)}{K})$, ce qui implique que μ diminue quand $N(t)$ augmente, ce qui n'est pas biologiquement réaliste. On corrigera par $\mu = \mu_0 + \mu_1\frac{N(t)}{K} = \mu_0(1 + \frac{N(t)}{K})$ ou simplement μ constant.

En utilisant la version simplifiée avec μ constant et en posant $\lambda = \lambda_0(1 - \frac{N(t)}{K})$, et $z_0 = \lambda_0 - \mu$, le taux de croissance intrinsèque maximal, on obtient:

$$z = \lambda - \mu = \lambda_0(1 - \frac{N(t)}{K}) - \mu = (\lambda_0 - \mu) - \frac{\lambda_0}{K}N(t) = z_0 - \frac{\lambda_0}{K}N(t)$$

On approche souvent $\lambda_0 \approx z_0$, et on pose simplement $z \approx z_0(1 - \frac{N(t)}{K})$.

Équation et Solution

Proposition 1.2.4. L'équation différentielle du modèle de Verhulst est alors:

$$N'(t) = zN(t) = z_0\left(1 - \frac{N(t)}{K}\right)N(t)$$

Avec condition initiale $N(0) = N_0$. La solution de cette équation différentielle est donnée par:

$$N(t) = \frac{K}{1 + \left(\frac{K}{N_0} - 1\right) e^{-z_0 t}}$$

Visualisation

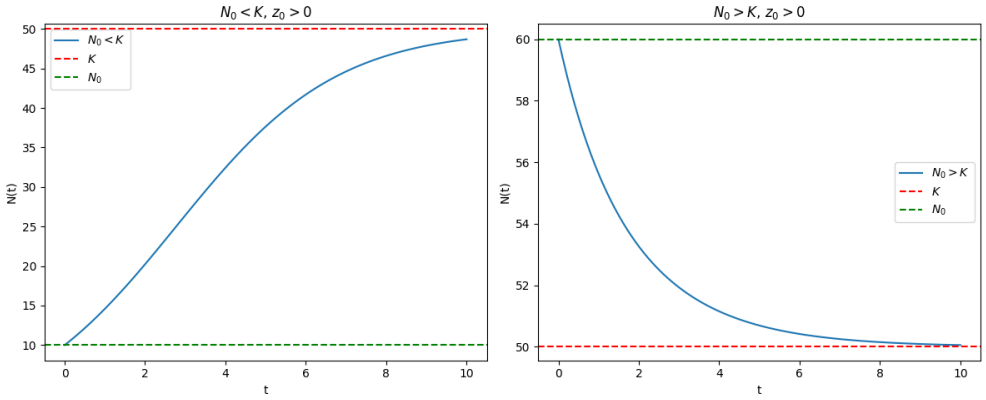

Figure 1.2: Visualisation du modèle de Verhulst pour différentes conditions initiales N_0 par rapport à la capacité biotique K.

1.3 Conclusion

Nous avons exploré les modèles discrets (géométrique) et continus (Malthus et Verhulst) pour la modélisation de populations. Le modèle géométrique et le modèle de Malthus, bien que simples, présentent des limitations importantes, notamment la prédiction d'une croissance infinie. Le modèle de Verhulst améliore ces modèles en introduisant la notion de capacité biotique, offrant une description plus réaliste de la dynamique des populations en tenant compte de la limitation des ressources.

Chapter 2

CM2

2.1 Introduction: Notion de Champ de Vecteurs et EDO

2.1.1 Généralités et Définitions

Nous allons étudier la notion de champ de vecteurs associé à une équation différentielle ordinaire (EDO).

Les modèles continus de la dynamique des populations sont des exemples de problèmes de Cauchy pour les EDOs.

Considérons une EDO du type :

$$y'(t) = f(t, y(t)), \quad t \in]t_0, T[, \tag{2.1}$$

avec la condition initiale :

$$y(t_0) = y_0, \tag{2.2}$$

où $y :]t_0, T[\to \mathbb{R}$ et $f :]t_0, T[\times \mathbb{R} \to \mathbb{R}$ est une fonction donnée, et $(t, x) \mapsto f(t, x)$.

2.2 Analyse Qualitative des Solutions d'EDO

Si l'on veut résoudre analytiquement une EDO, c'est-à-dire donner l'expression de $t \mapsto y(t)$, alors c'est terminé. Dans de nombreux cas, il suffit d'étudier la fonction $t \mapsto y(t)$.

Si l'on ne sait pas déterminer la solution analytique, on peut suivre une approche en deux étapes pour comprendre les solutions :

1. S'assurer de l'existence et de l'unicité de la solution, et de sa stabilité vis-à-vis des données du problème.
2. Puis analyser les propriétés qualitatives de cette solution par une simple analyse de $f(t, x)$. C'est ici qu'interviennent les champs de vecteurs.

2.3 Champ de Vecteurs: Définitions et Propriétés

2.3.1 Vecteur Tangent à une Courbe Paramétrée

Considérons une courbe paramétrée $t \mapsto \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} t \\ g(t) \end{pmatrix}$. Le vecteur tangent $\vec{v}$ à cette courbe est donné par :

$$\begin{aligned} \vec{v} = \begin{pmatrix} x'(t) \\ y'(t) \end{pmatrix} = \begin{pmatrix} \frac{dx}{dt} \\ \frac{dy}{dt} \end{pmatrix} = \begin{pmatrix} \frac{dx}{dt} \\ \frac{dy}{dx}\frac{dx}{dt} \end{pmatrix} = \frac{dx}{dt}\begin{pmatrix} 1 \\ \frac{dy}{dx} \end{pmatrix} \\ = \frac{1}{\frac{dt}{dx}}\begin{pmatrix} 1 \\ g'(x) \end{pmatrix} = \frac{1}{\dot{x}(t)}\begin{pmatrix} 1 \\ \dot{y}(t) \end{pmatrix} = \begin{pmatrix} 1 \\ \frac{\dot{y}(t)}{\dot{x}(t)} \end{pmatrix} = \begin{pmatrix} 1 \\ \dot{y}(t) \end{pmatrix} \end{aligned}$$

Si $x(t) = t$, alors $\dot{x}(t) = 1$, et le vecteur tangent devient $\vec{v} = \begin{pmatrix} 1 \\ \dot{y}(t) \end{pmatrix}$.

2.3.2 Lien entre Solution d'EDO et Vecteurs Vitesse

Proposition 2.3.1. y est solution de l'EDO $y'(t) = f(t, y(t))$ si et seulement si les vecteurs vitesses de la courbe paramétrée $t \mapsto \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} t \\ y(t) \end{pmatrix}$ au point t sont donnés par $u(t) = \begin{pmatrix} 1 \\ f(t, y(t)) \end{pmatrix}$.

2.3.3 Définition d'un Champ de Vecteurs

Definition 2.3.2. Soit $V : \mathbb{R}^2 \to \mathbb{R}^2$ un champ de vecteurs, défini par $(t, y) \mapsto V(t, y)$. Si ce champ de vecteurs est associé à l'EDO $y'(t) = f(t, y(t))$, alors $V(t, y) = \begin{pmatrix} 1 \\ f(t, y) \end{pmatrix}$.

2.4 Visualisation des Champs de Vecteurs (Implémentation Python)

2.4.1 Principe

Pour dessiner un champ de vecteurs, à chaque point $P = (P_x, P_y)$, on trace le vecteur $V \in V(P) = (V_x, V_y)$. On choisit une constante positive pour représenter les vecteurs trop longs.

Figure 2.1: Schéma de principe pour le dessin d'un champ de vecteurs.

2.4.2 Utilisation de `quiver` de Python

Pour implémenter le dessin de champs de vecteurs en Python, on utilise la fonction `quiver` de `matplotlib.pyplot`. Les arguments principaux sont : `plt.quiver(Px, Py, Vx, Vy, angles='xy', scale)`.

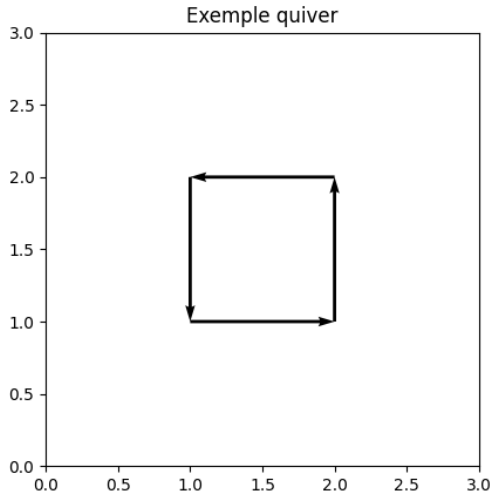

Figure 2.2: Exemple d'utilisation de quiver

2.4.3 Contrôle de la Longueur des Vecteurs

On peut ajouter un paramètre pour contrôler la longueur des vecteurs. Il est souvent nécessaire de normaliser les vecteurs pour une visualisation claire du champ de vecteurs.

Pour normaliser les vecteurs (V_x, V_y), on calcule d'abord la norme :

$$\text{norm} = \sqrt{V_x^2 + V_y^2} \tag{2.3}$$

Puis on normalise chaque composante :

$$V_x = V_x/\text{norm}$$
$$V_y = V_y/\text{norm}$$

2.5 Application: Recherche Approchée de Solutions (Python)

2.5.1 Objectif

On cherche une solution approchée de l'EDO $y'(t) = f(t, y(t))$, pour $t \in [t_0, t_0 + T]$ avec $y(t_0) = y_0$, en utilisant Python. Pour cela, il suffit de dessiner en quelques points où aboutit cette solution.

2.5.2 Méthode Python

Définissons un exemple de champ de vecteur avec $f(t, y) = r \cdot y \cdot (1 - y/k)$.

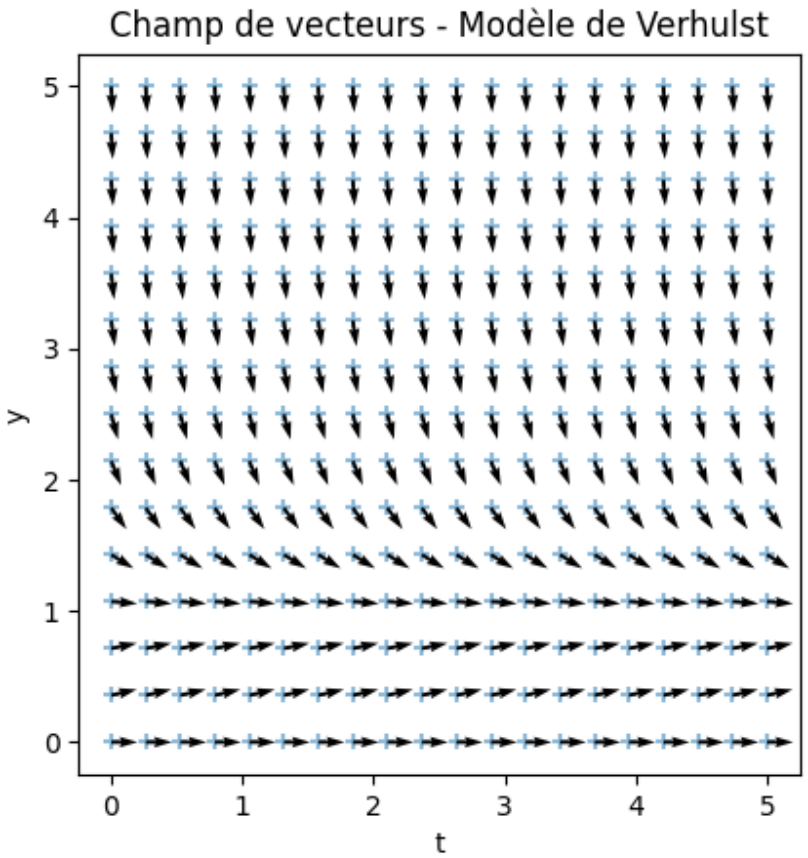

Figure 2.3: Champ de vecteurs pour le modèle de Verhulst.

2.6 Exploitation des Champs de Vecteurs pour Comprendre l'Allure des Solutions

Si l'on connaît les valeurs minimales et maximales de la solution, on peut avoir une idée de l'allure de la solution en observant le champ de vecteurs. Par exemple, si les vecteurs pointent vers le haut dans une certaine région, et vers le bas dans une autre, on peut déduire le comportement qualitatif des solutions dans ces régions.

Chapter 3

CM3

3.1 Analyse de la convergence

On va essayer de construire des polynômes qui passent par un ensemble (nuage) de points donnés.

Si ces points sont les valeurs d'une fonction, on aimerait :

- avoir un polynôme construit et d'autant plus proche de la fonction que le nombre de points est grand.

C'est-à-dire, est-ce que la suite des "meilleurs" polynômes tend vers la fonction lorsque le nombre de points tend vers l'infini?

Question : Comment quantifier cette convergence? C'est-à-dire à quelle vitesse (ordre) cette convergence a lieu.

3.1.1 Approches

- **Approche 1 :** Approximation linéaire
  - Moindre carré de degré 1
- **Approche 2 :** Polynôme d'ordre 1
  - Interpolation polynomiale (Lagrange)
- **Approche 3 :** Autres approches
  - Splines, ondelettes, etc.

3.1.2 Valeur approchée par itération

Définition de convergence

Definition 3.1.1. Soit $(x_n)_{n\in\mathbb{N}} \subset \mathbb{R}^d$ une suite qui converge vers $x^* \in \mathbb{R}^d$, pour une norme $\|\cdot\|$ de $\mathbb{R}^d$.

Vitesse (ordre) de convergence

- **Convergence linéaire (ordre 1)**: Si $K_1 = \lim_{n\to+\infty} \dfrac{\|x_{n+1} - x^*\|}{\|x_n - x^*\|}$ existe et $K_1 \in [0, 1[$, on dit que la suite converge **linéairement** vers x^*, ou que la convergence est d'ordre 1.
- **Convergence quadratique (ordre 2)**: Si $K_1 = 0$, $K_2 = \lim_{n\to+\infty} \dfrac{\|x_{n+1} - x^*\|}{\|x_n - x^*\|^2}$ existe et non nul, on dit que la suite converge **quadratiquement** vers x^*, ou que la convergence est d'ordre 2.

- **Convergence d'ordre q**: Si $K_q = \lim_{n\to+\infty} \dfrac{\|x_{n+1} - x^*\|}{\|x_n - x^*\|^q}$ existe et $\neq 0$, la convergence est d'ordre q.

Remark 3.1.2. La constante K_1 est appelée constante asymptotique d'erreur pour la convergence linéaire.

Exemples

Example 3.1.3. Soit $x_n = (0.2)^n$. On a $\lim_{n\to+\infty} x_n = 0$. La convergence est vers $x^* = 0$.

$$\lim_{n\to+\infty} \frac{\|x_{n+1} - x^*\|}{\|x_n - x^*\|} = \lim_{n\to+\infty} \frac{(0.2)^{n+1}}{(0.2)^n} = 0.2 \in [0, 1[$$

D'où, x_n converge à l'ordre 1. Sa constante asymptotique est $K_1 = 0.2$.

Example 3.1.4. Soit $y_n = (0.2)^{2^n}$.

$$y_{n+1} = (0.2)^{2^{n+1}} = (0.2)^{2^n \cdot 2} = ((0.2)^{2^n})^2 = (y_n)^2$$

$$\lim_{n\to+\infty} \frac{\|y_{n+1} - x^*\|}{\|y_n - x^*\|^2} = \lim_{n\to+\infty} \frac{y_{n+1}}{(y_n)^2} = \lim_{n\to+\infty} \frac{(y_n)^2}{(y_n)^2} = 1$$

D'où, convergence d'ordre 2, de constante $K_2 = 1$.

Définition formelle de la convergence d'ordre q

Definition 3.1.5. On dit que x_n converge vers x^* à l'ordre q s'il existe un entier $N \in \mathbb{N}$ et des constantes $A, B \in \mathbb{R}$ telles que pour tout $n > N$,

$$0 < A \leq \frac{\|x_{n+1} - x^*\|}{\|x_n - x^*\|^q} \leq B < +\infty$$

Remark 3.1.6. La convergence est au moins d'ordre q si seulement on a

$$\limsup_{n\to+\infty} \frac{\|x_{n+1} - x^*\|}{\|x_n - x^*\|^q} < +\infty$$

3.1.3 Interprétation pratique de la vitesse de convergence

Nombre de chiffres significatifs

Remark 3.1.7. Si $|x_n - x^*| = 4 \cdot 10^{-8} = 0.\underbrace{00000004}_{8 \text{ digits}}$, on dit que x_n et x^* ont 8 chiffres exacts après la virgule.

$$\log_{10} |x_n - x^*| = \log_{10}(4 \cdot 10^{-8}) = \log_{10} 4 - 8 \approx -8$$

On pose $d_n = -\log_{10} |x_n - x^*|$. d_n mesure approximativement le nombre de chiffres décimaux exacts entre x_n et x^*.

$$\lim_{n\to+\infty} \frac{\|x_{n+1}-x^*\|}{\|x_n-x^*\|^q} = K_q \Rightarrow \|x_{n+1}-x^*\| \approx K_q \|x_n-x^*\|^q$$

$$\begin{aligned}\log_{10}\|x_{n+1}-x^*\| &\approx \log_{10}(K_q\|x_n-x^*\|^q)\\ &= \log_{10} K_q + q\log_{10}\|x_n-x^*\|\end{aligned}$$

$$\begin{aligned}-d_{n+1} &\approx \log_{10} K_q + q(-d_n)\\ d_{n+1} &\approx qd_n - \log_{10} K_q\end{aligned}$$

Si $q=1$, $d_{n+1} \approx d_n - \log_{10} K_1$. À chaque itération, on gagne environ $-\log_{10} K_1$ chiffres significatifs.

Si $q>1$, $d_{n+1} \approx qd_n$. Le nombre de chiffres significatifs est approximativement multiplié par q à chaque itération.

3.1.4 Nombre d'itérations pour gagner un chiffre en convergence linéaire

Proposition 3.1.8. Si x_n converge à l'ordre 1 vers x^* avec une constante asymptotique K_1. Alors, le nombre d'itérations nécessaires pour gagner un chiffre significatif est approximativement $-\dfrac{1}{\log_{10}(K_1)}$.

Preuve. Soit m le nombre d'itérations pour gagner 1 chiffre significatif. Pour une convergence linéaire, $d_{n+1} \approx d_n - \log_{10} K_1$. En partant de d_n, après m itérations, on aura:

$$d_{n+m} \approx d_n - m\log_{10} K_1$$

On souhaite gagner 1 chiffre significatif, donc $d_{n+m} \approx d_n + 1$.

$$d_n + 1 = d_n - m\log_{10} K_1 \Leftrightarrow 1 = -m\log_{10} K_1 \Leftrightarrow m = -\frac{1}{\log_{10} K_1}$$

□

3.1.5 Linéarisation pour estimation graphique de q

$$\begin{aligned}\frac{\|x_{n+1}-x^*\|}{\|x_n-x^*\|^q} &\approx K_q\\ \log_{10}\|x_{n+1}-x^*\| &\approx \log_{10}(K_q\|x_n-x^*\|^q)\\ \log_{10}\|x_{n+1}-x^*\| &\approx \underbrace{q\log_{10}\|x_n-x^*\|}_{x} + \underbrace{\log_{10} K_q}_{b}\end{aligned}$$

C'est de la forme $y = qx + b$, où $y = \log_{10}\|x_{n+1}-x^*\|$, $x = \log_{10}\|x_n-x^*\|$, q est la pente et $b = \log_{10} K_q$ est l'ordonnée à l'origine.

3.1.6 Procédure pour déterminer q graphiquement

1. Calculer les erreurs $\|x_n-x^*\|$ et $\|x_{n+1}-x^*\|$ pour les itérations disponibles.
2. Calculer les logarithmes (base 10) de ces erreurs: $\log_{10}\|x_n-x^*\|$ et $\log_{10}\|x_{n+1}-x^*\|$.
3. Tracer le nuage de points $(\log_{10}\|x_n-x^*\|, \log_{10}\|x_{n+1}-x^*\|)$.
4. Estimer graphiquement ou par régression linéaire la pente q de la droite qui approxime au mieux ce nuage de points. Cette pente q est une estimation de l'ordre de convergence.

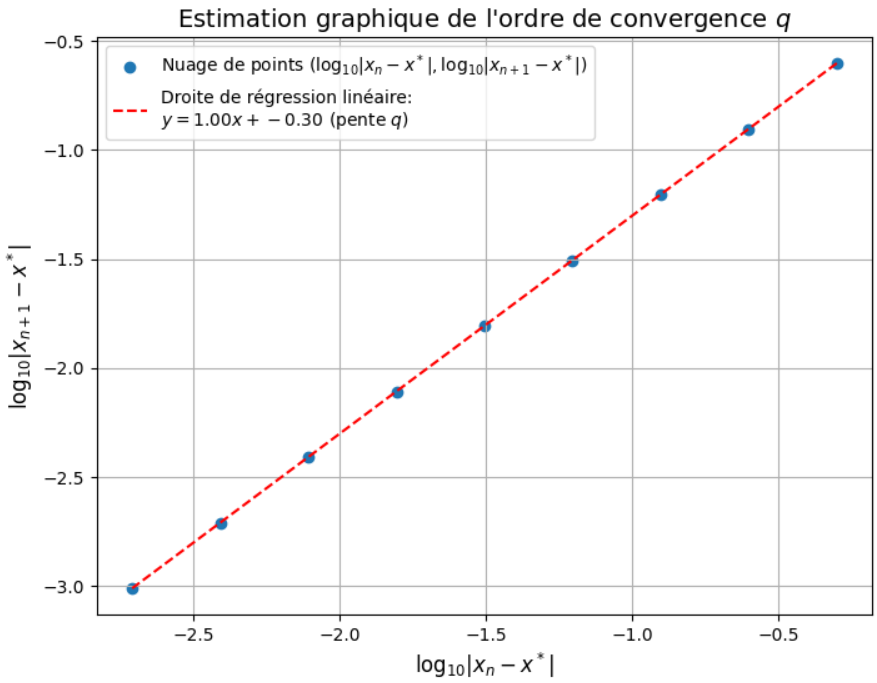

Figure 3.1: Estimation graphique de l'ordre de convergence q

3.2 Python code pour l'estimation de la convergence (exemple)

Listing 3.1: Code Python pour l'estimation de la convergence

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

# Exemple de suite xn (remplacez avec votre suite)
xn = np.array([0.7**(i) for i in range(1, 20)])
x_star = 0 # Valeur limite de la suite

errors_n = np.abs(xn - x_star) # Erreur absolue |xn - x*|
log_errors_n = np.log10(errors_n) # Logarithme base 10 des erreurs

ex = log_errors_n[:-1] # $\log_{10} \|x_n - x^*\|$
ey = log_errors_n[1:]  # $\log_{10} \|x_{n+1} - x^*\|$

plt.figure(figsize=(8, 6))
plt.scatter(ex, ey, label="Nuage de points")

ab = np.polyfit(ex, ey, 1) # Regression lineaire (y = ax + b)
y_fit = ab[0]*ex + ab[1]
plt.plot(ex, y_fit, color='red', linestyle='--', label=f"Droite de ré
    ↪ gression: $y = {ab[0]:.2f}x + {ab[1]:.2f}$")
```

```
plt.xlabel("$\log_{10}␣\|x_n␣-␣x^*\|$", fontsize=12)
plt.ylabel("$\log_{10}␣\|x_{n+1}␣-␣x^*\|$", fontsize=12)
plt.title("Estimation␣graphique␣de␣l'ordre␣de␣convergence␣$q$", fontsize
    ↪ =14)
plt.legend()
plt.grid(True)
plt.show() # Affiche le graphique (pour execution hors LaTeX)
```

Chapter 4

CM4

4.1 Introduction à l'interpolation

4.1.1 Définition

Definition 4.1.1. Soit un nuage de points (exemple: un ensemble discret de points du graphe d'une fonction). Interpréter ce nuage de points correspond à chercher un polynôme de degré $N-1$ qui passe par chacun de ces points.

- Comment le construire ?
- $P_{N-1} \in \mathbb{P}_{N-1}[x]$
- $P_{N-1}(x_i) = y_i$

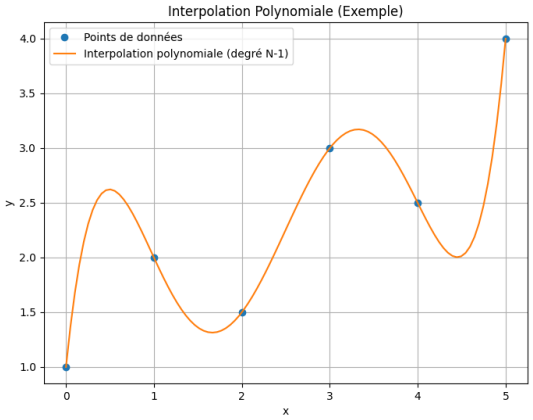

Figure 4.1: Illustration de l'interpolation polynomiale.

4.1.2 Motivations

- La solution d'un problème est fournie par une formule représentative : noyau de la chaleur (ex: convolution) et on cherche la solution en un nombre de points. $\Longrightarrow$ on approche alors la fonction par un polynôme i.e. chercher le polynôme de degré "bas" proche de la fonction.
- La solution d'un problème n'est connue qu'à travers ses valeurs en un nombre fini de points et on souhaite l'évaluer partout. $\Longrightarrow$ Interpolation.

- On peut utiliser l'interpolation dans :
  - la résolution numérique
  - la résolution numérique des Équations Différentielles Ordinaires (EDO)
  - la visualisation scientifique

Definition 4.1.2. Un tel polynôme est appelé **polynôme interpolateur de Lagrange** de degré $N-1$ de ces points.

4.1.3 Exemples d'interpolation

Théorème: Polynôme interpolateur de degré 1

Theorem 4.1.3. Soient (x_1, y_1) et (x_2, y_2) deux points distincts de $\mathbb{R}^2$. Il existe une unique droite D passant par ces deux points.

$$(x, y) \in D \iff (x - x_1)(y_2 - y_1) - (y - y_1)(x_2 - x_1) = 0$$

Si de plus, $x_1 \neq x_2$, il existe un unique polynôme de degré 1 (i.e., $P \in \mathbb{P}_1[x]$) tel que $y = P(x)$.
avec

$$P_1(x) = \frac{(x - x_2)}{(x_1 - x_2)} y_1 + \frac{(x - x_1)}{(x_2 - x_1)} y_2$$

pour des abscisses x_1, x_2 distinctes.

Example 4.1.4. Montrons que $M(x, y)$ est sur la droite (M_1M_2) si et seulement si les vecteurs $\overrightarrow{M_1M}$ et $\overrightarrow{M_1M_2}$ sont colinéaires. Soient $M_1(x_1, y_1)$, $M_2(x_2, y_2)$ et $M(x, y)$.

$$M \in (M_1M_2) \Leftrightarrow \overrightarrow{M_1M} // \overrightarrow{M_1M_2}$$

$$\Leftrightarrow \det(\overrightarrow{M_1M}, \overrightarrow{M_1M_2}) = 0$$

$$\Leftrightarrow \begin{vmatrix} x - x_1 & x_2 - x_1 \\ y - y_1 & y_2 - y_1 \end{vmatrix} = 0$$

$$\Leftrightarrow (x - x_1)(y_2 - y_1) - (x_2 - x_1)(y - y_1) = 0$$

Si $x_1 \neq x_2$, alors on peut réécrire l'équation de la droite sous la forme $y = ax + b$:

$$\implies y - y_1 = \frac{(y_2 - y_1)}{(x_2 - x_1)}(x - x_1)$$

$$\Leftrightarrow y = P_1(x)$$

Remark 4.1.5. On a l'écriture équivalente de P_1 :

$$P_1(x) = \frac{x - x_2}{x_1 - x_2} y_1 + \frac{x - x_1}{x_2 - x_1} y_2 = \frac{y_1 - y_2}{x_1 - x_2} x + \frac{x_2 y_1 - x_1 y_2}{x_2 - x_1} = a_1 x + a_0$$

c'est l'écriture dans la base $(1, x)$ de $\mathbb{P}_1[x]$ (base canonique).

- $P_1(x) = \frac{x-x_2}{x_1-x_2} y_1 + \frac{x-x_1}{x_2-x_1} y_2$

$$= \underbrace{\frac{x - x_2}{x_1 - x_2}}_{\ell_1(x)} y_1 + \underbrace{\frac{x - x_1}{x_2 - x_1}}_{\ell_2(x)} y_2$$

C'est l'écriture dans la base (ℓ_1, ℓ_2) de $\mathbb{P}_1[x]$ (base de Lagrange).

Remark 4.1.6. $\ell_1(x_1) = 1, \ell_1(x_2) = 0$
$\ell_2(x_1) = 0, \ell_2(x_2) = 1$

- $P_1(x) = y_1 + \frac{y_2 - y_1}{x_2 - x_1}(x - x_1)$ C'est l'écriture dans la base $(1, x - x_1)$ de $\mathbb{P}_1[x]$ (base de Newton).

Exemple: méthode de calcul employée

Chercher le polynôme interpolateur de Lagrange aux points $(x_1, y_1), (x_2, y_2), (x_3, y_3)$.

Méthode 1. $(x_1 \neq x_2, x_2 \neq x_3, x_1 \neq x_3)$
P_2 sera un polynôme de degré 2 :

$$P_2(x) = a_0 + a_1 x + a_2 x^2$$

Comme $P_2(x_i) = y_i$, pour $i = 1, 2, 3$, on a le système d'équations linéaires:

$$\begin{cases} P_2(x_1) = y_1 \\ P_2(x_2) = y_2 \\ P_2(x_3) = y_3 \end{cases} \implies \begin{cases} a_0 + a_1 x_1 + a_2 x_1^2 = y_1 \\ a_0 + a_1 x_2 + a_2 x_2^2 = y_2 \\ a_0 + a_1 x_3 + a_2 x_3^2 = y_3 \end{cases}$$

Matriciellement :

$$\begin{pmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} \implies \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \end{pmatrix}^{-1}}_{H^{-1}} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$$

Matrice de Vandermonde mal-conditionnée mais facile à construire.

Remark 4.1.7. Pour 2 points :

$$H = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \end{pmatrix} \implies H^{-1} = \frac{1}{x_2 - x_1} \begin{pmatrix} x_2 & -x_1 \\ -1 & 1 \end{pmatrix}$$

si $x_1 \neq x_2$.

Méthode 2. Base de Newton

$$P_2(x) = a_0 + a_1(x - x_1) + a_2(x - x_1)(x - x_2)$$

$$\begin{cases} P_2(x_1) = y_1 \implies a_0 = y_1 \\ P_2(x_2) = y_2 \implies a_0 + a_1(x_2 - x_1) = y_2 \\ P_2(x_3) = y_3 \implies a_0 + a_1(x_3 - x_1) + a_2(x_3 - x_1)(x_3 - x_2) = y_3 \end{cases}$$

$$\implies \begin{cases} a_0 = y_1 \\ a_1 = \frac{y_2 - y_1}{x_2 - x_1} \\ a_2 = \frac{y_3 - y_1 - \frac{y_2 - y_1}{x_2 - x_1}(x_3 - x_1)}{(x_3 - x_1)(x_3 - x_2)} = \frac{\frac{y_3 - y_1}{x_3 - x_1} - \frac{y_2 - y_1}{x_2 - x_1}}{(x_3 - x_2)} = \frac{\frac{y_3 - y_1}{x_3 - x_1} - \frac{y_2 - y_1}{x_2 - x_1}}{x_3 - x_2} \end{cases}$$

Cette construction est différentielle et facile à mettre à jour quand on rajoute un point supplémentaire (on rajoute uniquement une ligne).

On a donc :

$$a_0 = y_1, \quad a_1 = \frac{y_2 - y_1}{x_2 - x_1}, \quad a_2 = \frac{\frac{y_3 - y_1}{x_3 - x_1} - \frac{y_2 - y_1}{x_2 - x_1}}{x_3 - x_2}$$

Le polynôme P_2 s'écrit donc :

$$P_2(x) = y_1 + \frac{y_2 - y_1}{x_2 - x_1}(x - x_1) + \frac{\frac{y_3-y_1}{x_3-x_1} - \frac{y_2-y_1}{x_2-x_1}}{x_3 - x_2}(x - x_1)(x - x_2)$$

	a_0	a_1	a_2
x_1	y_1		
x_2	y_1	$\frac{y_2-y_1}{x_2-x_1}$	
x_3	y_1	$\frac{y_2-y_1}{x_2-x_1}$	$\frac{\frac{y_3-y_1}{x_3-x_1} - \frac{y_2-y_1}{x_2-x_1}}{x_3-x_2}$

Table 4.1: Tableau des coefficients pour la base de Newton.

Construction facile et différentielle par différences divisées : ajout d'un terme.

Méthode 3. Base de Lagrange

$$P_2(x) = \frac{(x - x_2)(x - x_3)}{(x_1 - x_2)(x_1 - x_3)}y_1 + \frac{(x - x_1)(x - x_3)}{(x_2 - x_1)(x_2 - x_3)}y_2 + \frac{(x - x_1)(x - x_2)}{(x_3 - x_1)(x_3 - x_2)}y_3$$

$$P_2(x) = \sum_{i=1}^{3} \left(\prod_{\substack{j=1 \\ j \neq i}}^{3} \frac{x - x_j}{x_i - x_j} \right) y_i$$

$$= \ell_1(x)y_1 + \ell_2(x)y_2 + \ell_3(x)y_3$$

Remark 4.1.8. Pour deux points (x_1, y_1) et (x_2, y_2), le polynôme interpolateur de Lagrange de degré 1 est :

$$P_1(x) = \frac{x - x_2}{x_1 - x_2}y_1 + \frac{x - x_1}{x_2 - x_1}y_2$$

4.2 Polynôme interpolateur de Lagrange

4.2.1 Définitions et propriétés

Théorème: Existence et unicité

Theorem 4.2.1 (Existence et unicité)**.** Soient $x_1, \ldots, x_n$ des réels deux à deux distincts et $y_1, \ldots, y_n$ des réels quelconques. Il existe un unique polynôme $P \in \mathbb{P}_{n-1}[x]$ (i.e. de degré au plus $n - 1$) tel que $P(x_i) = y_i, \forall i = 1, \ldots, n$.

On dit que P est le **polynôme interpolateur de Lagrange** aux points $(x_1, y_1), \ldots, (x_n, y_n)$.

Preuve: Soit l'application linéaire $\Phi : \mathbb{P}_{n-1}[x] \to \mathbb{R}^n$ définie par

$$P \mapsto \begin{pmatrix} P(x_1) \\ \vdots \\ P(x_n) \end{pmatrix}$$

Montrons que Φ est injective. Si $\Phi(P) = 0$, alors $P(x_i) = 0$ pour tout $i = 1, \ldots, n$. Donc P a n racines distinctes $x_1, \ldots, x_n$. Comme P est un polynôme de degré au plus $n-1$ avec n racines, il s'ensuit que $P \equiv 0$. Donc Φ est injective.

Comme $\mathbb{P}_{n-1}[x]$ et $\mathbb{R}^n$ sont deux espaces vectoriels de même dimension n, une application linéaire injective est aussi bijective, donc un isomorphisme d'espaces vectoriels. La bijectivité de Φ assure l'existence et l'unicité du polynôme interpolateur.

Definition 4.2.2. Si f est une fonction continue sur $[a,b] \to \mathbb{R}$, et $x_1, \ldots, x_n \in [a,b]$ sont n points deux à deux distincts, alors l'unique polynôme $P \in \mathbb{P}_{n-1}[x]$ tel que $P(x_i) = f(x_i)$, pour $i = 1, \ldots, n$ est appelé **polynôme d'interpolation de Lagrange** de f aux points $x_1, \ldots, x_n$.

4.2.2 Estimation de l'erreur d'interpolation

Théorème: Erreur d'interpolation

Theorem 4.2.3 (Erreur d'interpolation)**.** Soient $a < b$, $f : [a,b] \to \mathbb{R}$ une fonction continue, et $x_1, \ldots, x_n$ n points deux à deux distincts dans $[a,b]$. Soit P_n le polynôme d'interpolation de Lagrange de f aux points x_i. Si f est de classe $\mathcal{C}^n$ sur $[a,b]$, alors pour tout $x \in [a,b]$, il existe $\xi \in [a,b]$ tel que :

$$f(x) - P_n(x) = \frac{f^{(n)}(\xi)}{n!} \underbrace{\omega_n(x)}_{=\prod_{i=1}^{n}(x-x_i)}$$

où $\omega_n(x) = (x - x_1) \cdots (x - x_n)$.

Corollaire: Si $|f^{(n)}(x)|$ est bornée par M sur $[a,b]$ pour tout $x \in [a,b]$, alors $\forall x \in [a,b]$,

$$|f(x) - P_n(x)| \leq \frac{M}{n!}|\omega_n(x)| \leq \frac{M}{n!}(b-a)^n$$

Preuve: (à faire)

4.2.3 Implémentation avec Python

```
from scipy.interpolate import lagrange

x = np.array([1, 2, 3]) #à remplacer par les valeurs de x_1, x_2, x_3
y = np.array([2, 3, 1]) #à remplacer par les valeurs de y_1, y_2, y_3
p = lagrange(x,y)
print(p) # affiche le polynôme
print(p(2.5)) # évalue le polynôme en x=2.5
```

4.3 Construction des polynômes d'interpolation de Lagrange

4.3.1 Interpolation dans la base canonique (Vandermonde)

Construction

Soit $P(x) = \sum_{i=0}^{n-1} a_i x^i \in \mathbb{P}_{n-1}[x]$. On cherche les coefficients $a_0, \ldots, a_{n-1}$ tels que $P(x_k) = y_k$ pour $k = 1, \ldots, n$.

$$\sum_{i=0}^{n-1} a_i x_k^i = y_k, \quad k = 1, \ldots, n$$

Ce qui conduit au système linéaire matriciel suivant :

$$\begin{pmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^{n-1} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

$$V(x_1, \ldots, x_n)\mathbf{a} = \mathbf{y}$$

où $V(x_1, \ldots, x_n)$ est la matrice de Vandermonde.

C'est une matrice pleine, souvent mal conditionnée, mais facile à construire.

```
def VDM_Mat(x):
    n = len(x)
    V = np.zeros((n, n))
    for i in range(n):
        for j in range(n):
            V[i, j] = x[i]**j
    return V

def VDM_Poly(x,y):
    M = VDM_Mat(x)
    a = np.linalg.solve(M, y)
    return a
```

4.3.2 Evaluation efficace : Algorithme de Horner

Proposition: Algorithme de Horner

Proposition 4.3.1. Soit $P(x) = a_0x^n + a_1x^{n-1} + \cdots + a_{n-1}x + a_n$ un polynôme. On définit la suite $(q_k)_{k=0}^n$ par :

$$\begin{cases} q_0 = a_0 \\ q_k = q_{k-1}x + a_k, \quad k = 1, \ldots, n \end{cases}$$

Alors $q_n = P(x)$.

Exemple: $P(x) = x^2 + 2x + 1 = (x+1)^2$

Pour évaluer $P(2)$: $q_0 = a_0 = 1$ $q_1 = q_0 \times 2 + a_1 = 1 \times 2 + 2 = 4$ $q_2 = q_1 \times 2 + a_2 = 4 \times 2 + 1 = 9 = P(2) = 2^2 + 2 \times 2 + 1 = 9$

```
def Horner(P, xx):
    y = 0
    for a in P:
        y = y*xx + a
    return y

def IntVal_VDM (x, y, xx):
    a = VDM_Poly(x,y)
    YY = Horner(a[::-1], xx) # reverse a pour correspondre à l'ordre des
        ↪ coefficients dans Horner
    return YY
```

4.3.3 Interpolation dans la base duale: Formule de Lagrange et points barycentriques

Construction

L'idée est de prendre pour base de $\mathbb{P}_{n-1}[x]$ l'image réciproque de la base canonique de $\mathbb{R}^n$ par l'application Φ définie dans le théorème d'existence et unicité. On cherche donc une base $\{\mathcal{L}_j\}_{j=1}^n$ de $\mathbb{P}_{n-1}[x]$ telle que

$$\mathcal{L}_j(x_i) = \begin{cases} 1 & \text{si } i = j \\ 0 & \text{si } i \neq j \end{cases}$$

On construit les polynômes de Lagrange $\mathcal{L}_j(x)$ comme suit :

$$\mathcal{L}_j(x) = \prod_{\substack{i=1 \\ i \neq j}}^{n} \frac{x - x_i}{x_j - x_i} = \frac{\prod_{i \neq j}(x - x_i)}{\prod_{i \neq j}(x_j - x_i)}$$

Le polynôme interpolateur de Lagrange s'écrit alors :

$$P(x) = \sum_{j=1}^{n} y_j \mathcal{L}_j(x)$$

Chapter 5

CM5

5.1 Introduction à l'interpolation polynomiale

L'interpolation polynomiale est une technique fondamentale en analyse numérique qui consiste à trouver un polynôme qui passe par un ensemble donné de points. Plus précisément, étant donnés $n+1$ points $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$ où les x_i sont distincts, l'interpolation polynomiale cherche à construire un polynôme $P(x)$ de degré au plus n tel que $P(x_i) = y_i$ pour $i = 0, 1, \ldots, n$. Ce polynôme $P(x)$ est appelé le polynôme d'interpolation.

L'interpolation polynomiale a de nombreuses applications dans divers domaines tels que l'approximation de fonctions, l'intégration numérique, la résolution d'équations différentielles et le traitement de données expérimentales. Différentes méthodes existent pour construire ce polynôme d'interpolation, chacune ayant ses avantages et ses inconvénients en termes de complexité, de stabilité et de facilité d'implémentation. Nous allons explorer ici les méthodes de Lagrange et de Newton.

5.2 Interpolation de Lagrange

5.2.1 Formule de Lagrange

La formule d'interpolation de Lagrange est une manière explicite d'écrire le polynôme d'interpolation. Elle repose sur l'utilisation des polynômes de base de Lagrange.

Definition 5.2.1 (Polynômes de base de Lagrange)**.** Soient $x_0, x_1, \ldots, x_n$ des points distincts. Le polynôme nodal $\omega_{n+1}(x)$ est défini par :

$$\omega_{n+1}(x) = \prod_{i=0}^{n} (x - x_i)$$

et les polynômes de base de Lagrange $l_i(x)$ associés aux points $x_0, x_1, \ldots, x_n$ sont définis par :

$$l_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j} = \frac{\omega_{n+1}(x)}{(x - x_i)\omega'_{n+1}(x_i)}$$

pour $i = 0, 1, \ldots, n$.

On remarque que les polynômes de base de Lagrange vérifient la propriété suivante :

$$l_i(x_j) = \delta_{ij} = \begin{cases} 1 & \text{si } i = j \\ 0 & \text{si } i \neq j \end{cases}$$

Proposition 5.2.2 (Formule d'interpolation de Lagrange)**.** Le polynôme d'interpolation de Lagrange $P(x)$ de degré au plus n qui interpole les points $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$ est donné par :

$$P(x) = \sum_{i=0}^{n} y_i l_i(x)$$

Preuve. Pour vérifier que $P(x)$ est bien le polynôme d'interpolation, il suffit de montrer que $P(x_j) = y_j$ pour tout $j = 0, 1, \ldots, n$.

$$P(x_j) = \sum_{i=0}^{n} y_i l_i(x_j) = \sum_{i=0}^{n} y_i \delta_{ij} = y_j$$

De plus, chaque $l_i(x)$ est un polynôme de degré n, donc $P(x)$ est un polynôme de degré au plus n. □

5.2.2 Exemple

[Insérer un exemple si disponible dans les notes manuscrites, sinon, un exemple simple peut être construit ici.]

5.3 Erreur d'interpolation

5.3.1 Formule de l'erreur d'interpolation

L'erreur d'interpolation mesure la différence entre la fonction $f(x)$ que l'on cherche à interpoler et le polynôme d'interpolation $P(x)$.

Proposition 5.3.1 (Formule de l'erreur d'interpolation)**.** Soit $f \in C^{n+1}([a, b])$ et $P(x)$ le polynôme d'interpolation de Lagrange de degré au plus n interpolant f aux points $x_0, x_1, \ldots, x_n \in [a, b]$. Alors, pour tout $x \in [a, b]$, il existe un point $\xi_x \in [a, b]$ tel que :

$$f(x) - P(x) = \frac{f^{(n+1)}(\xi_x)}{(n+1)!} \omega_{n+1}(x) = \frac{f^{(n+1)}(\xi_x)}{(n+1)!} \prod_{i=0}^{n} (x - x_i)$$

5.3.2 Analyse de l'erreur

La formule de l'erreur montre que l'erreur d'interpolation dépend de deux facteurs principaux :

- La dérivée $(n+1)$-ième de la fonction f, $f^{(n+1)}(\xi_x)$. Si la dérivée $(n+1)$-ième de f est petite sur $[a, b]$, alors l'erreur d'interpolation sera petite.
- Le polynôme nodal $\omega_{n+1}(x) = \prod_{i=0}^{n} (x - x_i)$. La distribution des points d'interpolation $x_0, x_1, \ldots, x_n$ influence la magnitude de $\omega_{n+1}(x)$. Choisir des points d'interpolation de manière à minimiser $|\omega_{n+1}(x)|$ sur $[a, b]$ peut réduire l'erreur d'interpolation. Par exemple, les points de Chebyshev sont connus pour minimiser la norme infinie de $\omega_{n+1}(x)$ sur $[-1, 1]$.

5.3.3 Convergence

Pour assurer la convergence de l'interpolation polynomiale, c'est-à-dire que $P_n(x) \to f(x)$ lorsque $n \to \infty$, il ne suffit pas d'augmenter le degré du polynôme d'interpolation en utilisant des points équidistants. Le phénomène de Runge montre que pour certaines fonctions, l'interpolation polynomiale avec des points équidistants peut diverger entre les nœuds, même si la fonction est analytique. Cependant, si on choisit judicieusement les points d'interpolation, comme les points de Chebyshev, et si la fonction f est suffisamment régulière, on peut garantir la convergence de l'interpolation polynomiale.

5.4 Interpolation de Newton

5.4.1 Différences divisées

La méthode de Newton utilise les différences divisées pour construire le polynôme d'interpolation.

Definition 5.4.1 (Différences divisées)**.** Les différences divisées d'ordre zéro sont définies par $f[x_i] = f(x_i)$. Les différences divisées d'ordre supérieur sont définies par la formule de récurrence :

$$f[x_i, x_{i+1}, \ldots, x_{i+k}] = \frac{f[x_{i+1}, \ldots, x_{i+k}] - f[x_i, \ldots, x_{i+k-1}]}{x_{i+k} - x_i}$$

Proposition 5.4.2 (Formule d'interpolation de Newton)**.** Le polynôme d'interpolation de Newton de degré au plus n s'écrit :

$$P(x) = f[x_0] + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1) + \cdots + f[x_0, x_1, \ldots, x_n] \prod_{i=0}^{n-1} (x - x_i)$$

que l'on peut écrire sous la forme :

$$P(x) = \sum_{k=0}^{n} f[x_0, x_1, \ldots, x_k] \prod_{i=0}^{k-1} (x - x_i)$$

avec $\prod_{i=0}^{-1}(x - x_i) = 1$.

5.4.2 Algorithme de calcul des différences divisées

Les différences divisées peuvent être organisées dans un tableau triangulaire.

Listing 5.1: Calcul des différences divisées

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

def DifferencesDivisees(x,y):
    n = len(x)
    d = np.zeros([n, n])
    for i in range(n):
        d[i, 0] = y[i]
    for j in range(1, n):
        for i in range(n - j):
            d[i, j] = (d[i+1, j-1] - d[i, j-1]) / (x[i+j] - x[i])
    return d
```

Remark 5.4.3. La fonction `DifferencesDivisees(x,y)` prend en entrée les abscisses `x` et les ordonnées `y` des points d'interpolation et retourne une matrice `d` contenant les différences divisées. La diagonale supérieure de cette matrice contient les coefficients $f[x_0], f[x_0, x_1], \ldots, f[x_0, x_1, \ldots, x_n]$ nécessaires pour la formule d'interpolation de Newton.

5.4.3 Évaluation du polynôme de Newton : Formule de Horner-Newton

Pour évaluer efficacement le polynôme de Newton, on utilise la formule de Horner-Newton.

Listing 5.2: Évaluation du polynôme de Newton (Horner-Newton)

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

def HornerNewton(a,x,xx):
    n = len(a)
    yy = a[n-1]
    for i in range(n-2, -1, -1):
        yy = a[i] + (xx - x[i]) * yy
    return yy
```

Remark 5.4.4. La fonction `HornerNewton(a,x,xx)` prend en entrée le vecteur des coefficients des différences divisées `a` (diagonale de la matrice retournée par `DifferencesDivisees`), le vecteur des abscisses `x` des points d'interpolation, et un point `xx` où l'on souhaite évaluer le polynôme. Elle retourne la valeur du polynôme de Newton évalué en `xx`. Cette méthode est plus efficace pour évaluer le polynôme que l'évaluation directe de la formule de Newton.

5.5 Comparaison des méthodes et complexité

5.5.1 Complexité

- **Interpolation de Lagrange**: Le calcul des polynômes de base de Lagrange $l_i(x)$ et l'évaluation du polynôme d'interpolation de Lagrange nécessitent $\mathcal{O}(n^2)$ opérations pour un point donné x. Si l'on souhaite obtenir la forme développée du polynôme, la complexité est plus élevée.
- **Interpolation de Newton**: Le calcul des différences divisées nécessite $\mathcal{O}(n^2)$ opérations. L'évaluation du polynôme de Newton en utilisant la formule de Horner-Newton nécessite $\mathcal{O}(n)$ opérations par point. C'est une méthode efficace pour évaluer le polynôme une fois les différences divisées calculées.

5.5.2 Optimisation et ajout de points

- **Formule de Horner pour Newton**: La formule de Horner-Newton est cruciale pour l'efficacité de l'évaluation du polynôme de Newton. Elle réduit la complexité de l'évaluation à $\mathcal{O}(n)$ une fois les coefficients (différences divisées) sont connus.
- **Ajout d'un nouveau point**:
  - *Lagrange*: L'ajout d'un nouveau point d'interpolation nécessite de recalculer tous les polynômes de base de Lagrange et de refaire la somme. Cela peut être coûteux.
  - *Newton*: Si on ajoute un nouveau point (x_{n+1}, y_{n+1}), on peut facilement étendre le polynôme d'interpolation de Newton en calculant une différence divisée supplémentaire $f[x_0, x_1, \ldots, x_{n+1}]$ et en ajoutant un terme à la formule existante. Les différences divisées déjà calculées restent valides. C'est un avantage majeur de la méthode de Newton.

5.6 Conclusion

L'interpolation polynomiale est un outil puissant pour approximer des fonctions et traiter des données. Les méthodes de Lagrange et Newton offrent différentes approches pour construire et évaluer le polynôme d'interpolation. La méthode de Lagrange donne une formule explicite mais est moins pratique pour les calculs et l'ajout de nouveaux points. La méthode de Newton, basée sur les différences divisées, est efficace pour l'évaluation grâce à la formule de Horner-Newton et permet d'ajouter facilement de nouveaux points d'interpolation. Le choix de la méthode dépend du contexte et des besoins spécifiques de l'application.

Chapter 6

CM6

6.1 Polynômes de Tchebychev

6.1.1 Définition

On définit les polynômes de Tchebychev par récurrence :

- $T_0(x) = 1$
- $T_1(x) = x$
- $T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x), \quad n \geq 1$

Les premiers polynômes de Tchebychev sont donc :

- $T_0(x) = 1$
- $T_1(x) = x$
- $T_2(x) = 2xT_1(x) - T_0(x) = 2x^2 - 1$
- $T_3(x) = 2xT_2(x) - T_1(x) = 2x(2x^2 - 1) - x = 4x^3 - 3x$

6.1.2 Expression trigonométrique

On a aussi l'expression trigonométrique suivante pour les polynômes de Tchebychev :

$$T_n(x) = \cos(n \arccos(x)), \quad x \in [-1, 1]$$

On vérifie pour $n = 0, 1, 2$:

- $T_0(x) = \cos(0 \arccos(x)) = \cos(0) = 1$
- $T_1(x) = \cos(1 \arccos(x)) = \cos(\arccos(x)) = x$
- $T_2(x) = \cos(2 \arccos(x)) = 2\cos^2(\arccos(x)) - 1 = 2x^2 - 1$

Pour vérifier la relation de récurrence, posons $\theta = \arccos(x)$, donc $x = \cos(\theta)$. Alors

$$\begin{aligned} 2xT_n(x) - T_{n-1}(x) &= 2\cos(\theta)\cos(n\theta) - \cos((n-1)\theta) \\ &= \cos((n+1)\theta) + \cos((n-1)\theta) - \cos((n-1)\theta) \\ &= \cos((n+1)\theta) \\ &= T_{n+1}(x) \end{aligned}$$

On a utilisé la formule trigonométrique : $\cos(a)\cos(b) = \frac{1}{2}[\cos(a+b) + \cos(a-b)]$.

6.1.3 Propriétés

1. **Racines de $T_n(x)$**: $T_n(x) = 0 \Leftrightarrow \cos(n \arccos(x)) = 0$. Posons $x = \cos(\theta)$. Alors $\cos(n\theta) = 0 \Leftrightarrow n\theta = \frac{\pi}{2} + k\pi, k \in \mathbb{Z}$. Donc $\theta = \frac{\pi}{2n} + k\frac{\pi}{n}$. Pour avoir n racines distinctes dans $[-1, 1]$, on prend $k = 0, 1, \ldots, n-1$.
$$x_k = \cos\left(\frac{\pi}{2n} + k\frac{\pi}{n}\right), \quad k = 0, 1, \ldots, n-1$$
sont les n racines de $T_n(x)$ dans $[-1, 1]$.

2. $|T_n(x)| \leq 1$ pour $x \in [-1, 1]$. En effet, pour $x \in [-1, 1]$, $T_n(x) = \cos(n \arccos(x))$, et $|\cos(\cdot)| \leq 1$. De plus, $T_n(\cos(\frac{k\pi}{n})) = \cos(k\pi) = (-1)^k$. Donc $\max_{x\in[-1,1]} |T_n(x)| = 1$ atteint en $x'_k = \cos(\frac{k\pi}{n})$, $k = 0, \ldots, n$.

3. **Orthogonalité**: Les polynômes de Tchebychev sont orthogonaux pour le produit scalaire :
$$\langle f, g\rangle = \int_{-1}^{1} \frac{f(x)g(x)}{\sqrt{1-x^2}} dx$$
En effet, posons $x = \cos(\theta)$, $dx = -\sin(\theta)d\theta$, $\sqrt{1-x^2} = \sin(\theta)$.
$$\int_{-1}^{1} \frac{T_n(x)T_m(x)}{\sqrt{1-x^2}} dx = \int_{\pi}^{0} \frac{\cos(n\theta)\cos(m\theta)}{\sin(\theta)}(-\sin(\theta))d\theta = \int_{0}^{\pi} \cos(n\theta)\cos(m\theta)d\theta$$
On sait que $\int_0^\pi \cos(n\theta)\cos(m\theta)d\theta = 0$ si $n \neq m$, et $\int_0^\pi \cos^2(n\theta)d\theta = \frac{\pi}{2}$ si $n \neq 0$, et $\int_0^\pi \cos^2(0)d\theta = \pi$. Donc
$$\int_{-1}^{1} \frac{T_n(x)T_m(x)}{\sqrt{1-x^2}} dx = \begin{cases} 0 & \text{si } n \neq m \\ \pi & \text{si } n = m = 0 \\ \frac{\pi}{2} & \text{si } n = m \neq 0 \end{cases}$$

6.1.4 Application : Polynôme de meilleure approximation uniforme

Proposition 6.1.1. Soit P un polynôme de degré n avec coefficient dominant égal à 1, alors
$$\max_{x\in[-1,1]} |P(x)| \geq \max_{x\in[-1,1]} \left|\frac{1}{2^{n-1}} T_n(x)\right| = \frac{1}{2^{n-1}}$$
De plus, il y a égalité si et seulement si $P(x) = \frac{1}{2^{n-1}} T_n(x)$. On dit que $\frac{1}{2^{n-1}} T_n(x)$ est le polynôme de Tchebychev normalisé.

Soient $x_0, \ldots, x_n$ des points 2 à 2 distincts de $[-1, 1]$. On a :
$$\max_{x\in[-1,1]} \prod_{i=0}^{n} |x - x_i| \geq \max_{x\in[-1,1]} \prod_{i=1}^{n} |x - x_i^*|$$
avec x_i^* les racines de $T_{n+1}(x)$ translatées et dilatées sur $[-1, 1]$ (racines de Tchebychev).
$$x_k^* = \cos\left(\frac{\pi}{2(n+1)} + \frac{k\pi}{n+1}\right), \quad k = 0, \ldots, n$$
sont les racines de T_{n+1}.

6.1.5 Application à l'interpolation polynomiale

Soient $x_0, \ldots, x_n$ $n+1$ points 2 à 2 distincts, f une fonction $n+1$ fois continûment dérivable. Soit $P_n(x)$ le polynôme d'interpolation de Lagrange de f aux points x_i. Alors l'erreur d'interpolation est donnée par :
$$f(x) - P_n(x) = \frac{f^{(n+1)}(\xi_x)}{(n+1)!} \prod_{i=0}^{n} (x - x_i), \quad \xi_x \in [\min(x, x_i), \max(x, x_i)]$$

Donc

$$|f(x) - P_n(x)| \leq \frac{\max_{\xi \in [a,b]} |f^{(n+1)}(\xi)|}{(n+1)!} \max_{x \in [a,b]} \prod_{i=0}^{n} |x - x_i|$$

Pour minimiser l'erreur d'interpolation, il faut minimiser $\max_{x \in [a,b]} \prod_{i=0}^{n} |x - x_i|$. D'après le corollaire précédent, les points de Tchebychev minimisent ce terme (à translation et dilatation près pour adapter l'intervalle $[-1, 1]$ à $[a, b]$).

6.2 Intégration numérique

6.2.1 Motivation et concept général

On cherche à approcher numériquement l'intégrale d'une fonction f sur un intervalle $[a, b]$:

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

On approche $I(f)$ par une somme pondérée de valeurs de f en certains points $x_i \in [a, b]$:

$$Q_n(f) = \sum_{i=0}^{n} \omega_i f(x_i)$$

où x_i sont les **nœuds** de quadrature et ω_i sont les **poids** de quadrature. On cherche à construire des formules de quadrature $Q_n(f)$ qui soient exactes pour les polynômes de degré le plus élevé possible.

Definition 6.2.1. On dit qu'une formule de quadrature $Q_n(f) = \sum_{i=0}^{n} \omega_i f(x_i)$ est de degré de précision r si elle est exacte pour tous les polynômes de degré $\leq r$, et n'est pas exacte pour au moins un polynôme de degré $r + 1$. C'est-à-dire :

- $\forall P \in \mathbb{P}_r, \quad Q_n(P) = I(P) = \int_a^b P(x)dx$
- $\exists P \in \mathbb{P}_{r+1}, \quad Q_n(P) \neq I(P) = \int_a^b P(x)dx$

6.2.2 Construction des formules de quadrature

Idée : utiliser l'interpolation polynomiale. Soient $x_0, \ldots, x_n$ $n + 1$ points distincts dans $[a, b]$. Soit $P_n(x)$ le polynôme d'interpolation de Lagrange de f aux points x_i. On approche $I(f)$ par $I(P_n) = \int_a^b P_n(x)dx$. On sait que $P_n(x) = \sum_{i=0}^{n} f(x_i)L_i(x)$ où $L_i(x)$ sont les polynômes de Lagrange :

$$L_i(x) = \prod_{j=0, j \neq i}^{n} \frac{x - x_j}{x_i - x_j}$$

Donc

$$Q_n(f) = I(P_n) = \int_a^b P_n(x)dx = \int_a^b \sum_{i=0}^{n} f(x_i)L_i(x)dx = \sum_{i=0}^{n} f(x_i) \int_a^b L_i(x)dx$$

On pose $\omega_i = \int_a^b L_i(x)dx$. Alors $Q_n(f) = \sum_{i=0}^{n} \omega_i f(x_i)$ est une formule de quadrature.

Proposition 6.2.2. La formule de quadrature $Q_n(f)$ construite à partir de l'interpolation de Lagrange aux points $x_0, \ldots, x_n$ est de degré de précision au moins n.

Preuve. Si $P \in \mathbb{P}_n$, alors $P_n(x) = P(x)$ (le polynôme d'interpolation d'un polynôme de degré $\leq n$ est lui-même). Donc $Q_n(P) = \int_a^b P_n(x)dx = \int_a^b P(x)dx = I(P)$. Donc Q_n est exacte pour les polynômes de degré $\leq n$. □

6.2.3 Exemples

Example 6.2.3 (Formule du point milieu). $n = 0$, un seul point $x_0 = \frac{a+b}{2}$ (milieu de l'intervalle). $L_0(x) = 1$. $\omega_0 = \int_a^b L_0(x)dx = \int_a^b 1dx = b - a$. $Q_0(f) = (b-a)f\left(\frac{a+b}{2}\right)$. Degré de précision : 1. Exacte pour les polynômes de degré ≤ 1. Exemple : $\int_0^1 xdx = \frac{1}{2}$. $Q_0(x) = (1-0) \times \frac{0+1}{2} = \frac{1}{2}$. Exact. $\int_0^1 x^2dx = \frac{1}{3}$. $Q_0(x^2) = (1-0) \times \left(\frac{0+1}{2}\right)^2 = \frac{1}{4} \neq \frac{1}{3}$. Non exacte pour degré 2.

Example 6.2.4 (Formule des trapèzes). $n = 1$, deux points $x_0 = a$, $x_1 = b$. $L_0(x) = \frac{x-x_1}{x_0-x_1} = \frac{x-b}{a-b}$, $L_1(x) = \frac{x-x_0}{x_1-x_0} = \frac{x-a}{b-a}$. $\omega_0 = \int_a^b \frac{x-b}{a-b}dx = \frac{1}{a-b}\left[\frac{x^2}{2} - bx\right]_a^b = \frac{1}{a-b}\left[\left(\frac{b^2}{2} - b^2\right) - \left(\frac{a^2}{2} - ba\right)\right] = \frac{1}{a-b}\left[-\frac{b^2}{2} - \frac{a^2}{2} + ba\right] = \frac{b-a}{2}$. $\omega_1 = \int_a^b \frac{x-a}{b-a}dx = \frac{1}{b-a}\left[\frac{x^2}{2} - ax\right]_a^b = \frac{1}{b-a}\left[\left(\frac{b^2}{2} - ab\right) - \left(\frac{a^2}{2} - a^2\right)\right] = \frac{1}{b-a}\left[\frac{b^2}{2} - ab + \frac{a^2}{2}\right] = \frac{b-a}{2}$. $Q_1(f) = \frac{b-a}{2}[f(a) + f(b)]$. Degré de précision : 1. Exacte pour les polynômes de degré ≤ 1. Exemple : $\int_0^1 x^2dx = \frac{1}{3}$. $Q_1(x^2) = \frac{1-0}{2}[0^2 + 1^2] = \frac{1}{2} \neq \frac{1}{3}$. Non exacte pour degré 2.

6.2.4 Estimation de l'erreur

Soit $Q_n(f) = \sum_{i=0}^n \omega_i f(x_i)$ la formule de quadrature construite par interpolation de Lagrange aux points $x_0, \ldots, x_n$. On sait que l'erreur d'interpolation est :

$$f(x) - P_n(x) = \frac{f^{(n+1)}(\xi_x)}{(n+1)!} \prod_{i=0}^n (x - x_i)$$

Donc l'erreur de quadrature est :

$$E_n(f) = I(f) - Q_n(f) = \int_a^b f(x)dx - \int_a^b P_n(x)dx = \int_a^b [f(x) - P_n(x)]dx$$
$$= \int_a^b \frac{f^{(n+1)}(\xi_x)}{(n+1)!} \prod_{i=0}^n (x - x_i)dx = \frac{f^{(n+1)}(\xi)}{(n+1)!} \int_a^b \prod_{i=0}^n (x - x_i)dx$$

Si on suppose que $f^{(n+1)}$ est continue, on peut utiliser la formule de la moyenne pour l'intégrale :

$$E_n(f) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \int_a^b \prod_{i=0}^n (x - x_i)dx, \quad \xi \in [a, b]$$

où ξ est une valeur intermédiaire dans $[a, b]$. En pratique, on borne l'erreur :

$$|E_n(f)| \leq \frac{\max_{\xi \in [a,b]} |f^{(n+1)}(\xi)|}{(n+1)!} \int_a^b \prod_{i=0}^n |x - x_i|dx$$

Remark 6.2.5. Pour la formule du point milieu sur $[-1, 1]$, $x_0 = 0$, $n = 0$. $Q_0(f) = 2f(0)$. $\int_{-1}^1 (x - x_0)dx = \int_{-1}^1 xdx = 0$. Donc la formule d'erreur simple ne s'applique pas directement car $\int_a^b \prod_{i=0}^n (x - x_i)dx = 0$ dans certains cas (comme ici). Il faut une formule d'erreur plus précise.

Chapter 7

CM7

7.1 Introduction

L'intégration numérique est essentielle pour approximer les intégrales définies, en particulier lorsque le calcul analytique est impossible ou complexe. Les formules de quadrature offrent une approche pour estimer ces intégrales en utilisant une somme pondérée de valeurs de la fonction en des points spécifiques. Ce chapitre introduit les formules de quadrature composites, qui consistent à appliquer des formules de quadrature élémentaires sur des subdivisions de l'intervalle d'intégration pour améliorer la précision de l'approximation. Nous explorerons les concepts de formules de quadrature élémentaires et composites, et nous détaillerons les méthodes classiques telles que les méthodes des rectangles, des trapèzes, du point milieu et de Simpson.

7.2 Formules de Quadrature Élémentaires

Definition 7.2.1 (Formule de Quadrature Élémentaire)**.** On appelle formule de quadrature élémentaire sur l'intervalle $I_e = [-1, 1]$, associée aux points $x_i \in [-1, 1]$ et aux poids w_i, la formule :

$$I_e(f) = \sum_{i=1}^{N} w_i f(x_i)$$

qui approche l'intégrale $\int_{-1}^{1} f(t)dt$ pour une fonction $f \in C^0([-1, 1])$.

Plus généralement, pour un intervalle $[a, b]$, on peut définir une formule de quadrature élémentaire.

Definition 7.2.2. Étant donné un intervalle $[a, b]$, une formule de quadrature élémentaire $I_e(f)$ est une approximation de l'intégrale $\int_a^b f(t)dt$ de la forme :

$$I_e(f) = \sum_{i=1}^{N} \omega_i f(\xi_i)$$

où $\xi_i \in [a, b]$ sont les points de quadrature et ω_i sont les poids associés.

7.3 Formules de Quadrature Composites

7.3.1 Principe de construction

Pour améliorer la précision de l'approximation, on utilise des formules de quadrature composites. L'idée est de subdiviser l'intervalle d'intégration $[a, b]$ en n sous-intervalles $[x_i, x_{i+1}]$ où $a = x_0 < x_1 < \cdots < x_n = b$.

Sur chaque sous-intervalle $[x_i, x_{i+1}]$, on applique une formule de quadrature élémentaire. La formule de quadrature composite $I_c(f)$ sur $[a, b]$ est la somme des approximations sur chaque sous-intervalle :

$$I_c(f) = \sum_{i=0}^{n-1} I_e(f|_{[x_i, x_{i+1}]})$$

où $I_e(f|_{[x_i, x_{i+1}]})$ est la formule de quadrature élémentaire appliquée à la fonction f sur l'intervalle $[x_i, x_{i+1}]$.

Dans le cas d'une subdivision uniforme de l'intervalle $[a, b]$ en n sous-intervalles, on pose $h = \frac{b-a}{n}$ et $x_i = a + ih$ pour $i = 0, 1, \ldots, n$.

7.3.2 Visualisation

La visualisation des formules de quadrature composites permet de comprendre géométriquement l'approximation de l'intégrale par des sommes d'aires de rectangles ou de trapèzes.

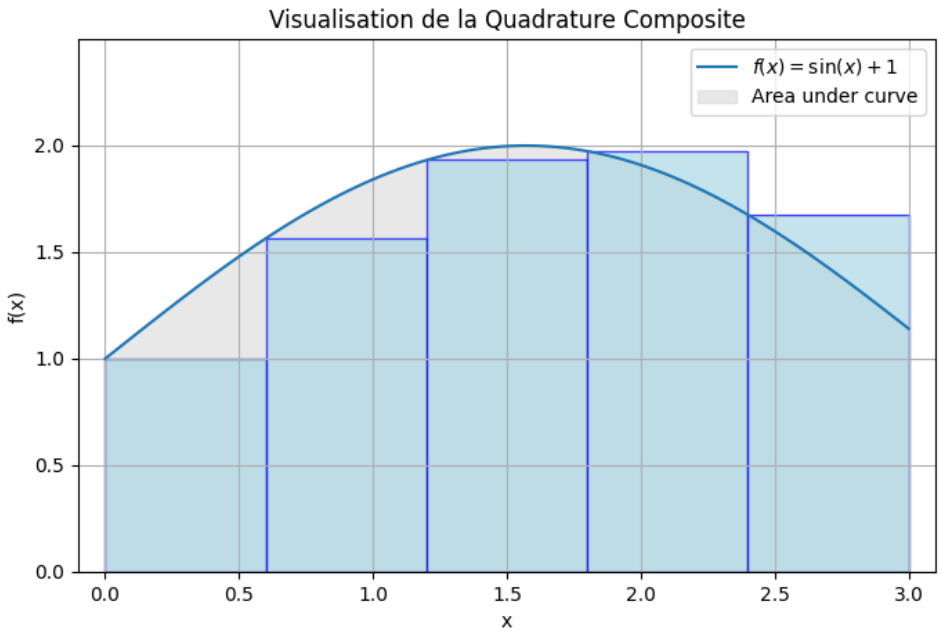

Figure 7.1: Visualisation d'une formule de quadrature composite (méthode des rectangles gauches).

Considérons une subdivision de l'intervalle $[a, b]$ en n intervalles $[x_j, x_{j+1}]$ de longueur $h = \frac{b-a}{n}$. La formule de quadrature composite s'écrit alors comme la somme des contributions sur chaque intervalle.

7.4 Méthodes de Quadrature Composites Classiques

7.4.1 Méthode des Rectangles

Définition

La méthode des rectangles approche l'intégrale sur chaque sous-intervalle $[x_j, x_{j+1}]$ par l'aire du rectangle de hauteur $f(x_j)$ (rectangle gauche) ou $f(x_{j+1})$ (rectangle droit), ou $f(\frac{x_j + x_{j+1}}{2})$ (point milieu). Pour la méthode des rectangles gauches élémentaire sur un intervalle $[\alpha, \beta]$, on a:

$$\int_\alpha^\beta f(t)dt \approx (\beta - \alpha) f(\alpha)$$

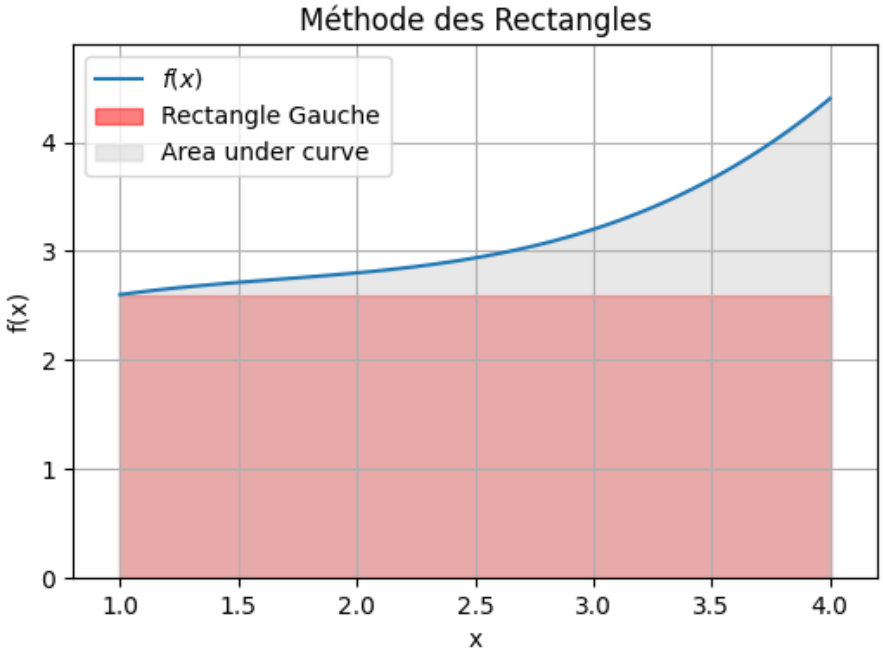

Figure 7.2: Méthode des Rectangles (gauche).

La formule de quadrature composite des rectangles gauches sur $[a, b]$ avec n intervalles est donnée par:

$$I_c(f) = h \sum_{j=0}^{n-1} f(x_j) = h \sum_{j=0}^{n-1} f(a + jh)$$

où $h = \frac{b-a}{n}$ et $x_j = a + jh$.

Erreur et Degré d'exactitude

L'erreur de la méthode des rectangles est d'ordre $O(h)$. Pour une fonction $f \in C^1([a, b])$, l'erreur $E_c(f) = \int_a^b f(t)dt - I_c(f)$ de la méthode des rectangles composite est majorée par :

$$|E_c(f)| \leq h\|f'\|_{\infty,[a,b]}(b - a) = \frac{(b - a)^2}{n}\|f'\|_{\infty,[a,b]}$$

Le degré d'exactitude de la méthode des rectangles élémentaire est 0, car elle est exacte pour les polynômes constants (de degré 0).

7.4.2 Méthode des Trapèzes

Définition

La méthode des trapèzes approche l'intégrale sur chaque sous-intervalle $[x_j, x_{j+1}]$ par l'aire du trapèze formé par les points $(x_j, 0), (x_{j+1}, 0), (x_{j+1}, f(x_{j+1})), (x_j, f(x_j))$. Pour la méthode des trapèzes élémentaire sur un intervalle $[\alpha, \beta]$, on a:

$$\int_\alpha^\beta f(t)dt \approx \frac{(\beta - \alpha)}{2}[f(\alpha) + f(\beta)]$$

La formule de quadrature composite des trapèzes sur $[a, b]$ avec n intervalles est donnée par:

$$I_c(f) = \frac{h}{2}[f(x_0) + 2\sum_{j=1}^{n-1} f(x_j) + f(x_n)] = \frac{h}{2}[f(a) + 2\sum_{j=1}^{n-1} f(a + jh) + f(b)]$$

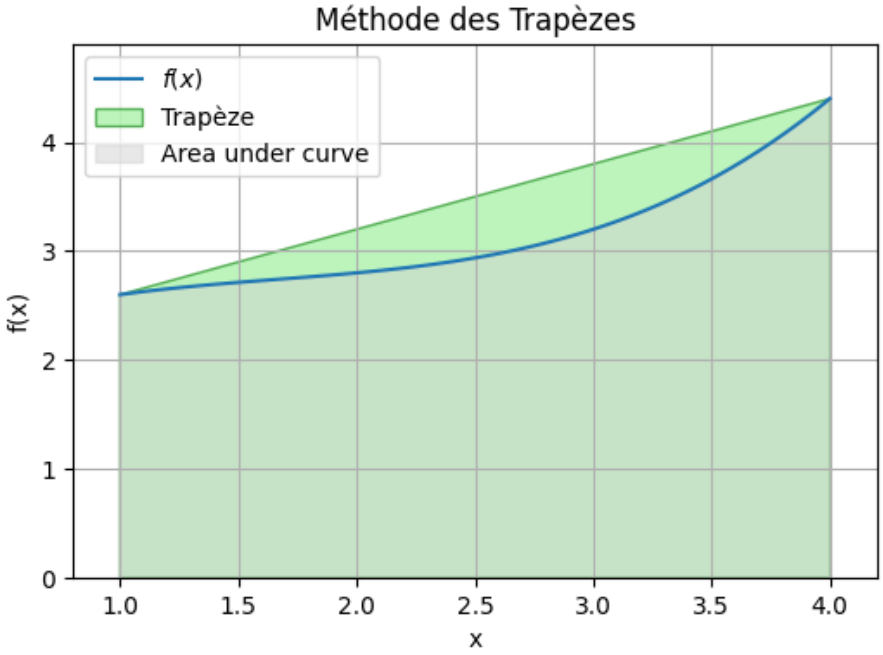

Figure 7.3: Méthode des Trapèzes.

où $h = \frac{b-a}{n}$ et $x_j = a + jh$.

Erreur et Degré d'exactitude

L'erreur de la méthode des trapèzes est d'ordre $O(h^2)$. Pour une fonction $f \in C^2([a,b])$, l'erreur $E_c(f) = \int_a^b f(t)dt - I_c(f)$ de la méthode des trapèzes composite est majorée par :

$$|E_c(f)| \leq \frac{h^2}{12}\|f''\|_{\infty,[a,b]}(b-a) = \frac{(b-a)^3}{12n^2}\|f''\|_{\infty,[a,b]}$$

Le degré d'exactitude de la méthode des trapèzes élémentaire est 1, car elle est exacte pour les polynômes de degré au plus 1.

7.4.3 Méthode du Point Milieu

Définition

La méthode du point milieu approche l'intégrale sur chaque sous-intervalle $[x_j, x_{j+1}]$ par l'aire du rectangle de hauteur $f(\frac{x_j+x_{j+1}}{2})$ au milieu de l'intervalle. Pour la méthode du point milieu élémentaire sur un intervalle $[\alpha, \beta]$, on a:

$$\int_\alpha^\beta f(t)dt \approx (\beta-\alpha)f\left(\frac{\alpha+\beta}{2}\right)$$

La formule de quadrature composite du point milieu sur $[a,b]$ avec n intervalles est donnée par:

$$I_c(f) = h\sum_{j=0}^{n-1} f\left(\frac{x_j + x_{j+1}}{2}\right) = h\sum_{j=0}^{n-1} f\left(a + (j+\frac{1}{2})h\right)$$

où $h = \frac{b-a}{n}$ et $x_j = a + jh$.

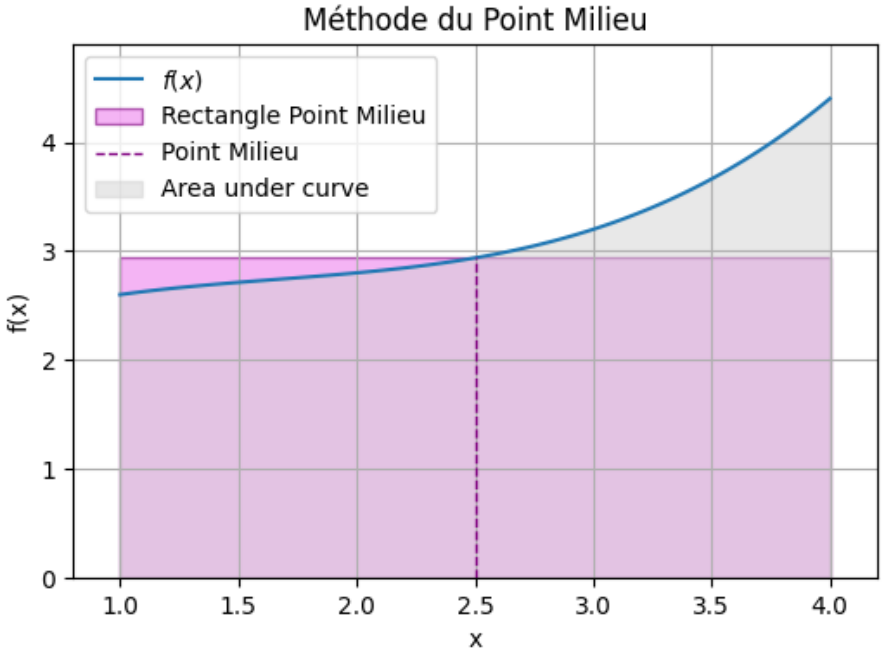

Figure 7.4: Méthode du Point Milieu.

Exercice : Degré d'exactitude

Déterminons le degré d'exactitude de la formule du point milieu élémentaire sur $[-1, 1]$. On teste pour les polynômes $f(x) = 1, x, x^2, \ldots$.

- Pour $f(x) = 1$:

$$\int_{-1}^{1} 1dx = [x]_{-1}^{1} = 2$$

$$I_e(f) = (1 - (-1))f\left(\frac{-1+1}{2}\right) = 2 \cdot f(0) = 2 \cdot 1 = 2$$

  Donc, $I_e(f) = \int_{-1}^{1} f(x)dx$.

- Pour $f(x) = x$:

$$\int_{-1}^{1} xdx = \left[\frac{x^2}{2}\right]_{-1}^{1} = 0$$

$$I_e(f) = 2 \cdot f(0) = 2 \cdot 0 = 0$$

  Donc, $I_e(f) = \int_{-1}^{1} f(x)dx$.

- Pour $f(x) = x^2$:

$$\int_{-1}^{1} x^2dx = \left[\frac{x^3}{3}\right]_{-1}^{1} = \frac{2}{3}$$

$$I_e(f) = 2 \cdot f(0) = 2 \cdot 0^2 = 0$$

  Donc, $I_e(f) \neq \int_{-1}^{1} f(x)dx$.

- Pour $f(x) = x^3$:

$$\int_{-1}^{1} x^3 dx = \left[\frac{x^4}{4}\right]_{-1}^{1} = 0$$

$$I_e(f) = 2 \cdot f(0) = 2 \cdot 0^3 = 0$$

Donc, $I_e(f) = \int_{-1}^{1} f(x)dx$.

- Pour $f(x) = x^4$:

$$\int_{-1}^{1} x^4 dx = \left[\frac{x^5}{5}\right]_{-1}^{1} = \frac{2}{5}$$

$$I_e(f) = 2 \cdot f(0) = 2 \cdot 0^4 = 0$$

Donc, $I_e(f) \neq \int_{-1}^{1} f(x)dx$.

La formule élémentaire du point milieu est exacte pour les polynômes de degré au moins 1. En fait, elle est exacte pour les polynômes de degré au plus 1. Testons pour un polynôme de degré 2 pour être sûr. On a vu que pour $f(x) = x^2$, la formule n'est pas exacte. Testons pour $f(x) = s^2$:

$$\int_{-1}^{1} s^2 ds = \frac{2}{3}$$

$$2 \cdot f(0) = 2 \cdot 0^2 = 0 \neq \frac{2}{3}$$

Donc la formule élémentaire n'est pas exacte pour les polynômes de degré 2.

Conclusion : La formule du point milieu élémentaire est exacte pour les polynômes de degré 1. En fait, par un calcul plus précis et en considérant l'erreur, on montre que la formule du point milieu élémentaire est exacte pour les polynômes de degré au plus 1. On dit que le degré d'exactitude est 1. En réalité, le degré d'exactitude de la méthode du point milieu élémentaire est en fait 1, car elle est exacte pour les polynômes de degré au plus 1. La formule est d'ordre 2, donc l'erreur est en $O(h^2)$ pour les fonctions suffisamment régulières.

En réalité, la formule du point milieu élémentaire est exacte pour les polynômes de degré au plus 1. La formule du point milieu élémentaire est en fait exacte pour les polynômes de degré au plus 1. Donc le degré d'exactitude est 1. On dit que la formule du point milieu est d'ordre 2 car l'erreur est en $O(h^2)$.

7.4.4 Méthode de Simpson

Définition

La méthode de Simpson approche l'intégrale sur chaque intervalle double $[x_j, x_{j+2}]$ en utilisant un polynôme de degré 2 qui interpole les points $(x_j, f(x_j)), (x_{j+1}, f(x_{j+1})), (x_{j+2}, f(x_{j+2}))$. Sur un intervalle $[\alpha, \beta]$, en posant $\gamma = \frac{\alpha+\beta}{2}$, la formule de Simpson élémentaire est:

$$\int_{\alpha}^{\beta} f(x)dx \approx \frac{(\beta - \alpha)}{6}\left[f(\alpha) + 4f\left(\frac{\alpha+\beta}{2}\right) + f(\beta)\right]$$

La formule de quadrature composite de Simpson sur $[a, b]$ avec n intervalles (n pair) est donnée par:

$$I_c(f) = \frac{h}{3}\left[f(x_0) + 4\sum_{j=1}^{n/2} f(x_{2j-1}) + 2\sum_{j=1}^{n/2-1} f(x_{2j}) + f(x_n)\right]$$

où $h = \frac{b-a}{n}$ et $x_j = a + jh$.

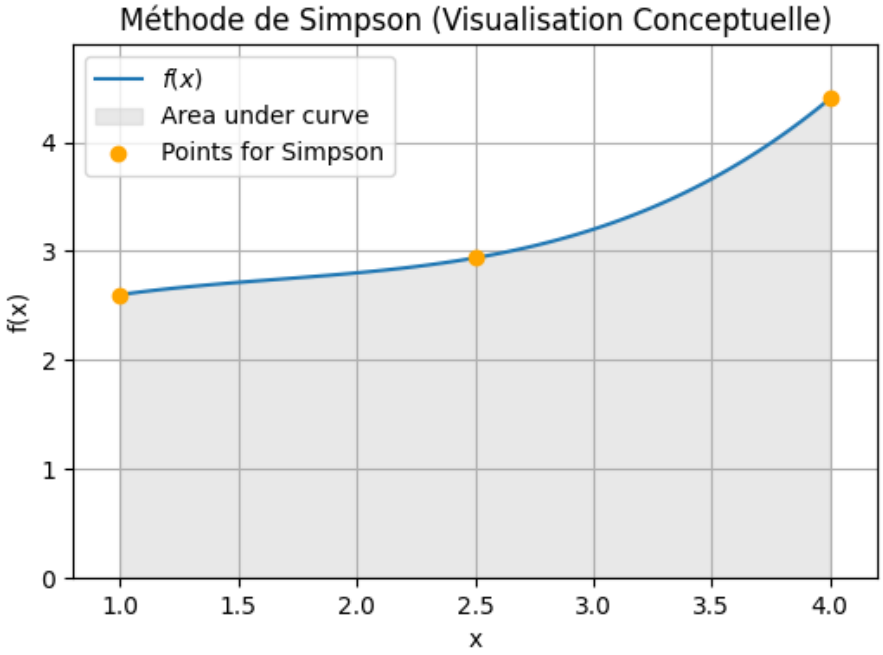

Figure 7.5: Méthode de Simpson (visualisation conceptuelle montrant l'aire sous la courbe).

Erreur et Degré d'exactitude

L'erreur de la méthode de Simpson est d'ordre $O(h^4)$. Pour une fonction $f \in C^4([a,b])$, l'erreur $E_c(f) = \int_a^b f(t)dt - I_c(f)$ de la méthode de Simpson composite est majorée par :

$$|E_c(f)| \leq \frac{h^4}{2880}\|f^{(4)}\|_{\infty,[a,b]}(b-a) = \frac{(b-a)^5}{2880n^4}\|f^{(4)}\|_{\infty,[a,b]}$$

Le degré d'exactitude de la méthode de Simpson élémentaire est 3, car elle est exacte pour les polynômes de degré au plus 3.

7.5 Comparaison des Méthodes

Proposition 7.5.1. Si $f \in C^2([a,b])$, l'erreur de quadrature de la méthode composite associée à une subdivision uniforme de pas h est majorée par :

$$|E_c(f)| = |\int_a^b f(t)dt - I_c(f)| \leq h^2\frac{(b-a)}{12}\|f''\|_{\infty,[a,b]}$$

pour la méthode des trapèzes.

Proposition 7.5.2. Si $f \in C^4([a,b])$, l'erreur de quadrature de la méthode composite de Simpson associée à une subdivision uniforme de pas h est majorée par :

$$|E_c(f)| = |\int_a^b f(t)dt - I_c(f)| \leq h^4\frac{(b-a)}{2880}\|f^{(4)}\|_{\infty,[a,b]}$$

Méthode	Formule élémentaire (sur $[\alpha, \beta]$)	Ordre de convergence	Degré d'exactitude
Rectangles (gauche)	$(\beta - \alpha) f(\alpha)$	$O(h)$	0
Trapèzes	$\frac{(\beta-\alpha)}{2}[f(\alpha) + f(\beta)]$	$O(h^2)$	1
Point Milieu	$(\beta - \alpha) f\left(\frac{\alpha+\beta}{2}\right)$	$O(h^2)$	1
Simpson	$\frac{(\beta-\alpha)}{6}\left[f(\alpha) + 4f\left(\frac{\alpha+\beta}{2}\right) + f(\beta)\right]$	$O(h^4)$	3

Table 7.1: Comparaison des méthodes de quadrature composites classiques.

7.6 Conclusion

Les formules de quadrature composites offrent des outils efficaces pour l'approximation numérique d'intégrales définies. Le choix de la méthode dépend de la régularité de la fonction à intégrer et de la précision souhaitée. La méthode de Simpson, avec son ordre de convergence élevé, est généralement préférée pour les fonctions régulières lorsque la précision est primordiale. Cependant, les méthodes des rectangles et des trapèzes peuvent être suffisantes dans des contextes où une précision moindre est acceptable ou pour des fonctions moins régulières.

7.7 Exercices

1. Calculer l'intégrale $\int_0^1 x^2 dx$ en utilisant les méthodes des rectangles gauches, des trapèzes et du point milieu avec $n = 4$ sous-intervalles. Comparer les résultats avec la valeur exacte de l'intégrale.

2. Estimer l'erreur pour chaque méthode utilisée dans l'exercice précédent en utilisant les bornes d'erreur théoriques.

3. Déterminer le nombre de sous-intervalles nécessaires pour que la méthode des trapèzes approche l'intégrale $\int_0^2 e^x dx$ avec une précision de 10^{-3}.

4. Calculer le degré d'exactitude de la formule de quadrature élémentaire des trapèzes sur $[-1, 1]$.

Chapter 8

CM8

8.1 Méthodes de Newton-Cotes

C'est une généralisation des méthodes élémentaires d'intégration numérique.

Definition 8.1.1 (Méthode de Newton-Cotes)**.** On appelle méthode de Newton-Cotes d'ordre K, la méthode élémentaire consistant en l'utilisation du polynôme d'interpolation associé aux K+1 points équidistants $x_i = \alpha + i\frac{\beta-\alpha}{K}$ pour $i = 0, ..., K$. La formule de quadrature est donnée par :

$$\int_\alpha^\beta f(x)dx \approx \sum_{i=0}^{K} w_i f(x_i) \tag{8.1}$$

où les poids w_i sont calculés comme suit :

$$w_i = \int_\alpha^\beta L_i(x)dx = \int_\alpha^\beta \prod_{\substack{j=0 \\ j\neq i}}^{K} \frac{x - x_j}{x_i - x_j} dx$$

Pour illustrer la méthode de Newton-Cotes, considérons l'exemple de la méthode des trapèzes (Newton-Cotes d'ordre 1). Le code Python suivant génère une visualisation de cette méthode.

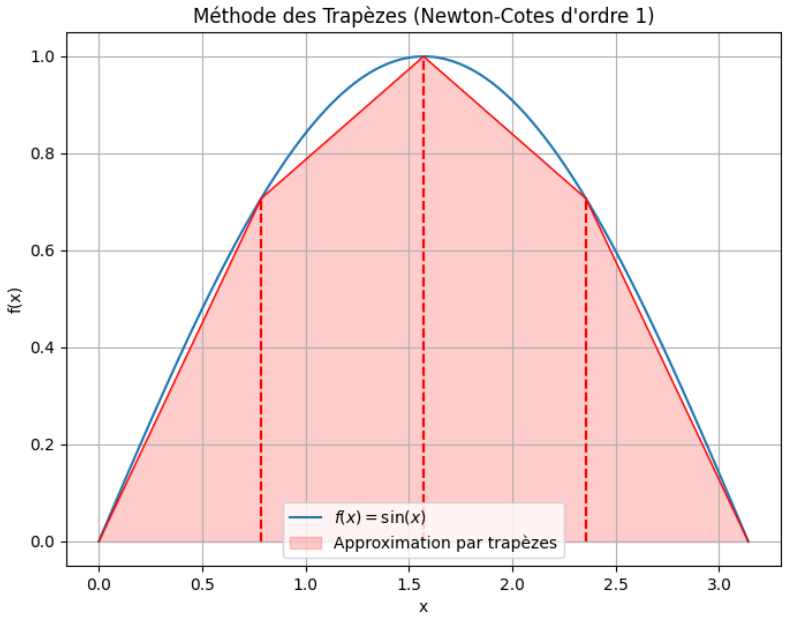

Figure 8.1: Illustration de la méthode des trapèzes pour l'intégration numérique de $f(x) = \sin(x)$ sur $[0, \pi]$.

La formule générale de Newton-Cotes est donnée par l'équation (8.1). Pour référence, nous présentons également la formule dans un environnement `listings`:

Listing 8.1: Formule de Newton-Cotes d'ordre K

```
$\int_{\alpha}^{\beta} f(x) dx \approx \sum_{i=0}^{K} w_i f(x_i)$
```

Remark 8.1.2.

- Cette formule est d'ordre K si K est impair.
- Elle est d'ordre K+1 si K est pair.
- On n'utilise ces méthodes que pour K pair, sauf le cas K=1.

Example 8.1.3 (Cas particuliers).

- Si K=1, on a la formule des trapèzes.
- Si K=2, on a la formule de Simpson.
- Si K=4, on a la formule de Boole-Villarceau. Pour un intervalle $[a, b]$ et $h = (b - a)/4$:

$$\int_a^b f(x)dx \approx \frac{2h}{45}[7f(a) + 32f(a + h) + 12f(a + 2h) + 32f(a + 3h) + 7f(b)]$$

Note : la transcription dans les notes manuscrites donne une formule légèrement différente, potentiellement normalisée ou spécifique :

$$\int_a^b f(x)dx \approx \frac{7}{40}f(a) + \frac{16}{45}f(a + h) + \frac{2}{15}f(a + 2h) + \frac{16}{45}f(a + 3h) + \frac{7}{40}f(b)$$

(Il est recommandé de vérifier cette formule car la somme des coefficients n'est pas égale à $(b-a)$.)

- Pour K=6, on a la formule de Hardy.

Remark 8.1.4. Pour $K \geq 8$, on a des poids w_i négatifs, ce qui rend les formules sensibles aux erreurs d'arrondi.

Theorem 8.1.5 (Erreur de la méthode de Newton-Cotes)**.** Soient $I(f) = \int_\alpha^\beta f(x)dx$ l'intégrale exacte et $I_K(f) = \sum_{i=0}^K w_i f(x_i)$ l'approximation par la formule de Newton-Cotes d'ordre K. L'erreur est $E(f) = I(f) - I_K(f)$. Supposons que la méthode d'intégration soit d'ordre p ($\geq K$). Posons la fonction $g(x) = (x-t)_+^p/p!$ où $(x-t)_+^p = (x-t)^p$ si $x > t$ et 0 sinon. Le noyau de Peano est défini par $K(t) = E(g) = E\left(x \mapsto \frac{(x-t)_+^p}{p!}\right)$.

$$K(t) = \int_\alpha^\beta \frac{(x-t)_+^p}{p!}dx - \sum_{i=0}^K w_i \frac{(x_i-t)_+^p}{p!}$$

Alors, pour toute fonction $f \in C^{p+1}([\alpha,\beta])$, on a :

$$\boxed{E(f) = \int_\alpha^\beta \frac{K(t)}{p!} f^{(p+1)}(t)dt}$$

Si $K(t)/p!$ est de signe constant sur $[\alpha,\beta]$, alors il existe $\xi \in [\alpha,\beta]$ tel que :

$$\boxed{E(f) = \frac{f^{(p+1)}(\xi)}{p!} \int_\alpha^\beta K(t)dt}$$

On appelle noyau de Peano associé à la méthode, la fonction $t \mapsto K(t)/p!$.

Remark 8.1.6.
- Lorsque K(t) est de signe constant, on peut écrire $E(f) = Cf^{(p+1)}(\xi)$ avec $C = \frac{1}{p!}\int_\alpha^\beta K(t)dt$.
- On a aussi la relation $E(f) = \frac{E(x\mapsto x^{p+1})}{(p+1)!} f^{(p+1)}(\xi)$.
- Dans les méthodes de Newton-Cotes, le noyau de Peano $K(t)$ a un signe constant.

8.2 Construction de Formule de Quadrature (à points inconnus)

8.2.1 Formule de Gauss-Legendre

On cherche s'il existe un meilleur choix des points (noeuds) $x_1, \ldots, x_n$ dans $[\alpha,\beta]$ pour que la formule de quadrature associée $\int_\alpha^\beta f(x)dx \approx \sum_{i=1}^n w_i f(x_i)$ soit exacte sur $\mathbb{R}_N[X]$ avec N le plus grand possible. Pour n points, on a $2n$ inconnues (n poids w_i et n noeuds x_i). On peut donc espérer rendre la formule exacte pour les polynômes de degré jusqu'à $2n-1$.

Pour simplifier, on travaille sur l'intervalle $[-1,1]$.

Example 8.2.1 (Formule à 2 points (n=2))**.** On cherche x_1, x_2, w_1, w_2 tels que $\int_{-1}^1 f(x)dx \approx w_1 f(x_1) +$

$w_2 f(x_2)$ soit exacte pour $f(x) = 1, x, x^2, x^3$.

$$\int_{-1}^{1} 1dx = 2 = w_1 + w_2$$
$$\int_{-1}^{1} xdx = 0 = w_1 x_1 + w_2 x_2$$
$$\int_{-1}^{1} x^2 dx = \frac{2}{3} = w_1 x_1^2 + w_2 x_2^2$$
$$\int_{-1}^{1} x^3 dx = 0 = w_1 x_1^3 + w_2 x_2^3$$

La solution de ce système est :

- $x_1 = -\frac{1}{\sqrt{3}}$, $x_2 = \frac{1}{\sqrt{3}}$ (racines du polynôme de Legendre $L_2(x) = \frac{1}{2}(3x^2 - 1)$)
- $w_1 = 1$, $w_2 = 1$

La formule est donc :

$$\int_{-1}^{1} f(x)dx \approx f\left(-\frac{1}{\sqrt{3}}\right) + f\left(\frac{1}{\sqrt{3}}\right)$$

Cette formule est exacte sur $\mathbb{R}_3[X]$ (degré $2n - 1 = 2(2) - 1 = 3$).

Example 8.2.2 (Formule à 3 points (n=3))**.** On cherche $x_1, x_2, x_3, w_1, w_2, w_3$ pour que $\int_{-1}^{1} f(x)dx \approx w_1 f(x_1) + w_2 f(x_2) + w_3 f(x_3)$ soit exacte pour $f(x) = 1, x, x^2, x^3, x^4, x^5$ (degré $2n - 1 = 5$). On obtient que les x_i sont les racines du polynôme de Legendre de degré 3, $L_3(x) = \frac{1}{2}(5x^3 - 3x)$. Les racines sont $x_1 = -\sqrt{\frac{3}{5}}$, $x_2 = 0$, $x_3 = \sqrt{\frac{3}{5}}$. Les poids associés sont $w_1 = \frac{5}{9}$, $w_2 = \frac{8}{9}$, $w_3 = \frac{5}{9}$. La formule est :

$$\int_{-1}^{1} f(x)dx \approx \frac{5}{9} f\left(-\sqrt{\frac{3}{5}}\right) + \frac{8}{9} f(0) + \frac{5}{9} f\left(\sqrt{\frac{3}{5}}\right)$$

Proposition 8.2.3 (Quadrature de Gauss-Legendre)**.** Considérons la formule de quadrature à n points sur $[-1, 1]$:

$$\int_{-1}^{1} P(x)dx \approx \sum_{i=1}^{n} w_i P(x_i)$$

Pour que cette formule soit exacte pour les polynômes de degré $\leq 2n - 1$, il faut et il suffit que :

- Les abscisses $x_1, \ldots, x_n$ soient les n racines du polynôme de Legendre de degré n, $L_n(x)$. Ces polynômes peuvent être définis par la relation de récurrence :

$$L_0(x) = 1$$
$$L_1(x) = x$$
$$(n+1)L_{n+1}(x) = (2n+1)xL_n(x) - nL_{n-1}(x) \quad \text{pour } n \geq 1$$

- Les poids w_i soient donnés par l'intégration des polynômes de Lagrange associés aux points x_i:

$$w_i = \int_{-1}^{1} \prod_{\substack{j=1 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j} dx, \quad i = 1, \ldots, n$$

La formule de quadrature ainsi obtenue est appelée Formule de Gauss-Legendre.

Résolution alternative pour l'exemple à 2 points

Reprenons le cas $n = 2$: $\int_{-1}^{1} f(x)dx \approx w_1 f(x_1) + w_2 f(x_2)$. La formule doit être exacte pour les polynômes de degré ≤ 3. Considérons le polynôme $P(x) = (x - x_1)(x - x_2) = x^2 - (x_1 + x_2)x + x_1x_2$. Il est de degré 2, donc la formule doit être exacte pour $P(x)$.

$$\int_{-1}^{1} (x - x_1)(x - x_2)dx = w_1(x_1 - x_1)(x_1 - x_2) + w_2(x_2 - x_1)(x_2 - x_2) = 0 + 0 = 0$$

Calculons l'intégrale :

$$\int_{-1}^{1} (x^2 - (x_1 + x_2)x + x_1x_2)dx = \left[\frac{x^3}{3} - (x_1 + x_2)\frac{x^2}{2} + x_1x_2x\right]_{-1}^{1}$$

$$= \left(\frac{1}{3} - \frac{x_1 + x_2}{2} + x_1x_2\right) - \left(-\frac{1}{3} - \frac{x_1 + x_2}{2} - x_1x_2\right) = \frac{2}{3} + 2x_1x_2$$

Donc, $\frac{2}{3} + 2x_1x_2 = 0 \implies x_1x_2 = -\frac{1}{3}$. (Produit des racines)

Considérons le polynôme $Q(x) = x(x - x_1)(x - x_2) = x^3 - (x_1 + x_2)x^2 + x_1x_2x$. Il est de degré 3, donc la formule doit être exacte pour $Q(x)$.

$$\int_{-1}^{1} x(x - x_1)(x - x_2)dx = w_1x_1(x_1 - x_1)(x_1 - x_2) + w_2x_2(x_2 - x_1)(x_2 - x_2) = 0 + 0 = 0$$

Calculons l'intégrale :

$$\int_{-1}^{1} (x^3 - (x_1 + x_2)x^2 + x_1x_2x)dx = \left[\frac{x^4}{4} - (x_1 + x_2)\frac{x^3}{3} + x_1x_2\frac{x^2}{2}\right]_{-1}^{1}$$

$$= \left(\frac{1}{4} - \frac{x_1 + x_2}{3} + \frac{x_1x_2}{2}\right) - \left(\frac{1}{4} + \frac{x_1 + x_2}{3} + \frac{x_1x_2}{2}\right) = -\frac{2(x_1 + x_2)}{3}$$

Donc, $-\frac{2(x_1+x_2)}{3} = 0 \implies x_1 + x_2 = 0$. (Somme des racines)

On a le système :

$$\begin{aligned} x_1 + x_2 &= 0 \\ x_1x_2 &= -\frac{1}{3} \end{aligned}$$

De la première équation, $x_2 = -x_1$. En substituant dans la seconde : $x_1(-x_1) = -\frac{1}{3} \implies -x_1^2 = -\frac{1}{3} \implies x_1^2 = \frac{1}{3}$. Donc $x_1 = \pm\frac{1}{\sqrt{3}}$. Les abscisses sont $x_1 = -\frac{1}{\sqrt{3}}$ et $x_2 = \frac{1}{\sqrt{3}}$.

Pour trouver les poids w_1, w_2, on utilise l'exactitude pour $f(x) = 1$ et $f(x) = x$:

- $f(x) = 1$: $\int_{-1}^{1} 1dx = 2 = w_1 + w_2$
- $f(x) = x$: $\int_{-1}^{1} xdx = 0 = w_1x_1 + w_2x_2 = w_1(-\frac{1}{\sqrt{3}}) + w_2(\frac{1}{\sqrt{3}}) = \frac{1}{\sqrt{3}}(w_2 - w_1)$

De $w_2 - w_1 = 0$, on tire $w_1 = w_2$. En substituant dans $w_1 + w_2 = 2$, on obtient $2w_1 = 2 \implies w_1 = 1$. Donc $w_1 = w_2 = 1$.

On retrouve bien la formule : $\int_{-1}^{1} f(x)dx \approx f(-\frac{1}{\sqrt{3}}) + f(\frac{1}{\sqrt{3}})$.

Chapter 9

CM9

9.1 Solution approchée d'EDO

9.1.1 Motivations

Définitions

Soit $f : [a,b] \times \mathbb{R}^d \to \mathbb{R}^d$, $(t,x) \mapsto f(t,x)$, avec $a, b \in \mathbb{R}$, $a < b$. La fonction f est donnée par ses composantes $f_i : [a,b] \times \mathbb{R}^d \to \mathbb{R}$, $f(t,x) = (f_1(t,x), \dots, f_d(t,x))^T$. On note $g^{(n)}$ la dérivée d'ordre n d'une fonction $g : \mathbb{R} \to \mathbb{R}$, et g' la dérivée d'ordre 1.

Si $g : [a,b] \to \mathbb{R}^d$ est continue ainsi que toutes ses dérivées jusqu'à l'ordre p, on notera $g \in C^p([a,b], \mathbb{R}^d)$ ou simplement $g \in C^p([a,b])$ s'il n'y a pas d'ambiguïté. On a : $(g_i \in C^p([a,b], \mathbb{R}), \forall i = 1, \dots, d) \iff (g \in C^p([a,b], \mathbb{R}^d))$.

Definition 9.1.1.

- On appelle équation différentielle d'ordre 1 une équation de la forme :

$$y'(t) = f(t, y(t)), \quad \forall t \in [t_0, t_0 + T]$$

- On appelle EDO d'ordre p une équation de la forme $y^{(p)}(t) = f(t, y(t), y'(t), \dots, y^{(p-1)}(t))$, où $f : [a,b] \times (\mathbb{R}^d)^p \to \mathbb{R}^d$ est continue.
- Une fonction y de classe C^p vérifiant une EDO est dite solution de l'EDO.
- Résoudre une EDO, c'est déterminer toutes les solutions de cette EDO.
- Lorsque $d \neq 1$, on parle de système d'EDOs.

Remark 9.1.2. Toute EDO d'ordre $p \geq 1$ peut se ramener à un système d'EDOs d'ordre 1.

Definition 9.1.3 (Problème de Cauchy)**.** On appelle *problème de Cauchy* de l'EDO, la donnée de la valeur de la solution en un point $t_0 \in [a,b]$. Le couple (t_0, y_0) est appelé *condition initiale* et le problème de Cauchy consiste à trouver $y(t)$ tel que:

$$\begin{cases} y'(t) = f(t, y(t)), & t \in [t_0, t_0 + T] \\ y(t_0) = y_0 \in \mathbb{R}^d \end{cases}$$

La recherche d'une fonction de classe C^1 vérifiant le système ci-dessus. Pour une EDO d'ordre p, $y^{(p)}(t) = f(t, y, y', \dots, y^{(p-1)})$, on donne p conditions initiales : $y(t_0)$, $y'(t_0)$, ..., $y^{(p-1)}(t_0)$.

Exemples

Example 9.1.4 (Pendule). L'équation du pendule simple est :

$$y''(t) + \frac{g}{L}\sin(y(t)) = 0$$

C'est une EDO d'ordre 2. Posons $x_1(t) = y(t)$ et $x_2(t) = y'(t)$. Alors on a :

$$\begin{aligned} x_1'(t) &= y'(t) = x_2(t) \\ x_2'(t) &= y''(t) = -\frac{g}{L}\sin(y(t)) = -\frac{g}{L}\sin(x_1(t)) \end{aligned}$$

On pose $X(t) = \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} \in \mathbb{R}^2$. Le système s'écrit alors $X'(t) = f(t, X(t))$ avec

$$f(t, X) = f(t, x_1, x_2) = \begin{pmatrix} x_2 \\ -\frac{g}{L}\sin(x_1) \end{pmatrix}$$

Ici, $f : \mathbb{R} \times \mathbb{R}^2 \to \mathbb{R}^2$. La fonction f ne dépend pas explicitement de t. La condition initiale serait $y(t_0) = y_0$, $y'(t_0) = y_0'$, ce qui correspond à $X(t_0) = \begin{pmatrix} y_0 \\ y_0' \end{pmatrix}$.

Example 9.1.5 (Chute libre avec frottement). L'équation est $z'' = -g + k(z')z'$, où z est l'altitude et $k(z')$ modélise le coefficient de frottement qui dépend de la vitesse z'. C'est une EDO d'ordre 2. Posons $v_1 = z$ et $v_2 = z'$. Le système devient :

$$\begin{aligned} v_1' &= z' = v_2 \\ v_2' &= z'' = -g + k(v_2)v_2 \end{aligned}$$

On pose $V(t) = \begin{pmatrix} v_1(t) \\ v_2(t) \end{pmatrix}$. Le système s'écrit $V'(t) = F(t, V(t))$ avec

$$F(t, V) = F(t, v_1, v_2) = \begin{pmatrix} v_2 \\ -g + k(v_2)v_2 \end{pmatrix}$$

C'est un système différentiel d'ordre 1.

Example 9.1.6 (Épidémiologie - Modèle SI). Considérons un modèle simple Susceptible-Infecté (SI). Soit $X(t)$ la proportion de susceptibles et $Y(t)$ la proportion d'infectés. On suppose $X(t) + Y(t) = 1$. Le taux de nouvelles infections est proportionnel aux rencontres entre susceptibles et infectés.

$$\begin{aligned} Y'(t) &= aX(t)Y(t) \\ X'(t) &= -aX(t)Y(t) \end{aligned}$$

où a est le taux de transmission. En utilisant $X(t) = 1 - Y(t)$, on peut réduire à une seule équation pour $Y(t)$:

$$Y'(t) = a(1 - Y(t))Y(t)$$

C'est une EDO d'ordre 1 pour $Y(t)$. Le diagramme représente un modèle avec une population N_{pop} divisée en individus sains X_{sains} et malades $X_{malades}$, avec un taux d'infection a.

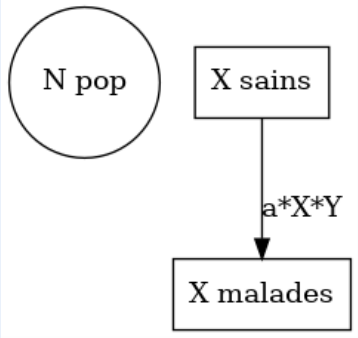

Figure 9.1: Diagramme simplifié d'un modèle épidémiologique.

Le bloc de code suivant semble définir une fonction pour une API, sans lien direct avec le modèle SI décrit ci-dessus.

```
import numpy as np
def F(t, y):
  v1, v2 = y[0], y[1]
  return np.array([v2, -g*L*np.exp(-ay)]) * np.exp(-ay)
```

Illustration (Graphique)

Considérons le problème de Cauchy $x'(t) = f(t, x(t))$, $x(t_0) = x_0$. On cherche à approcher la solution $x(t)$ sur l'intervalle $[t_0, T]$. On discrétise l'intervalle de temps : $t_0 < t_1 < \cdots < t_N = T$. Souvent, on utilise un pas constant $\Delta t = (T - t_0)/N$, et $t_n = t_0 + n\Delta t$. On calcule une suite de points $(x_n)_{n=0}^{N}$ où x_n est une approximation de $x(t_n)$. x_0 est donné par la condition initiale. On place les points (t_n, x_n) sur un graphique. En reliant ces points, on obtient une approximation du graphe de la solution $x(t)$.

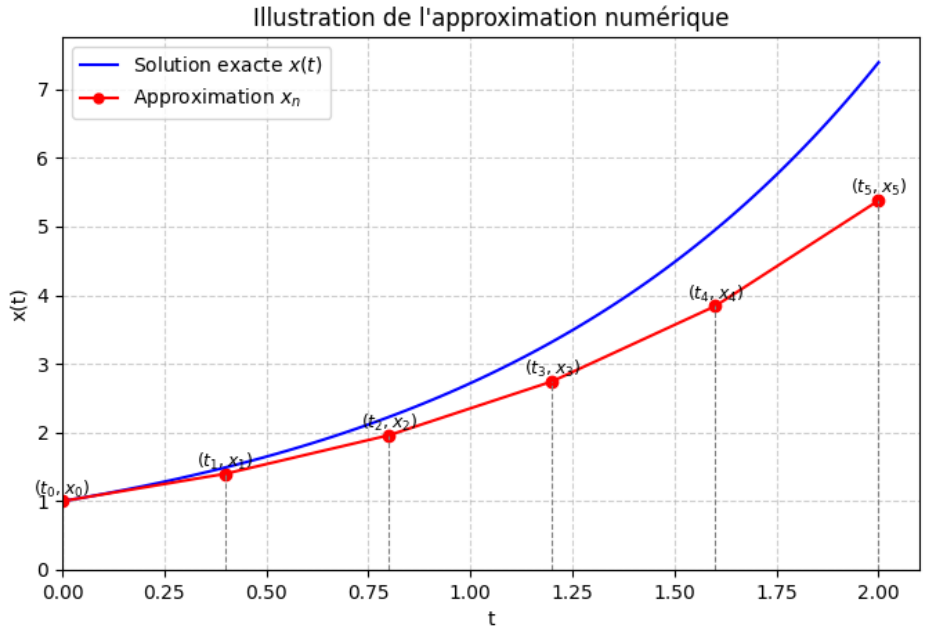

Figure 9.2: Illustration graphique de l'approximation d'une solution d'EDO par des points (t_n, x_n).

Nécessité de la solution approchée

On considère le point suivant :

1. Il existe des solutions $x(t)$ pour le problème de Cauchy $x'(t) = f(t, x(t))$, $x(t_0) = x_0$ sur $[t_0, t_0 + T]$.
2. On ne sait pas résoudre (analytiquement) la plupart des EDOs. Par exemple, l'équation $y' = \sin(t^2)$ n'a pas de primitive exprimable avec des fonctions usuelles. Des cas particuliers comme les EDOs linéaires ou à variables séparées peuvent être résolus.
3. L'objectif est de trouver $x(t)$ tel que $x'(t) = f(t, x(t))$.

9.1.2 Problème de population des lapins

Considérons un modèle prédateur-proie (Lotka-Volterra modifié). Soit $L(t)$ la population de lapins (proies) et $R(t)$ la population de renards (prédateurs). On a le problème de Cauchy suivant :

$$\begin{cases} L'(t) = R(t)L(t)(1 - pL(t)) \\ R'(t) = R(t)(L(t)^2 - pR(t)^2) \\ L(0) = L_0 \\ R(0) = R_0 \end{cases}$$

où p, L_0, R_0 sont des paramètres (notation L_2, pL, pR_2 from notes seems unclear, interpreting based on context as L_0, R_0, p). Ce système n'est pas résoluble analytiquement en général.

On peut cependant résoudre numériquement ce problème et obtenir une approximation de la solution. La condition de résolution numérique dépend de plusieurs facteurs :

- Stabilité : Dépendance continue de la solution vis-à-vis des données du problème.
- Régularité de la solution.
- Existence et unicité de la solution.
- Le problème est bien posé.

9.1.3 Théorème de Cauchy-Lipschitz

Ce théorème garantit l'existence et l'unicité de la solution pour une classe importante de problèmes de Cauchy.

Definition 9.1.7 (Fonction Lipschitzienne)**.** On dit que $f : [a, b] \times \mathbb{R}^d \to \mathbb{R}^d$ est Lipschitzienne par rapport à sa seconde variable s'il existe une constante positive K, appelée constante de Lipschitz, telle que pour tout $t \in [a, b]$ et pour tous $y_1, y_2 \in \mathbb{R}^d$:

$$\|f(t, y_1) - f(t, y_2)\| \leq K\|y_1 - y_2\|$$

où $\|\cdot\|$ est une norme sur $\mathbb{R}^d$.

Theorem 9.1.8 (Cauchy-Lipschitz)**.** Soit $f : [t_0, t_0 + T] \times \mathbb{R}^d \to \mathbb{R}^d$ une fonction continue. Si f est Lipschitzienne par rapport à sa seconde variable sur $[t_0, t_0 + T] \times \mathbb{R}^d$, alors pour tout $y_0 \in \mathbb{R}^d$, le problème de Cauchy

$$\begin{cases} y'(t) = f(t, y(t)), \quad t \in [t_0, t_0 + T] \\ y(t_0) = y_0 \end{cases}$$

admet une unique solution $y \in C^1([t_0, t_0 + T], \mathbb{R}^d)$.

9.1.4 Exemple de schémas numériques

Formulation Intégrale

Proposition 9.1.9. Si $x \in C^1([t_0, t_0+T])$ est solution de $x'(t) = f(t, x(t))$, alors $x(t)$ vérifie l'équation intégrale :

$$x(t) = x(t_0) + \int_{t_0}^{t} f(s, x(s))ds, \quad \forall t \in [t_0, t_0+T]$$

Preuve. On intègre l'équation $x'(s) = f(s, x(s))$ entre t_0 et t:

$$\int_{t_0}^{t} x'(s)ds = \int_{t_0}^{t} f(s, x(s))ds$$

Par le théorème fondamental de l'analyse, $\int_{t_0}^{t} x'(s)ds = x(t) - x(t_0)$. Donc, $x(t) - x(t_0) = \int_{t_0}^{t} f(s, x(s))ds$, ce qui donne le résultat. □

Construction du schéma d'Euler explicite

On cherche à approcher $x(t)$ aux points de la discrétisation $t_n = t_0 + n\Delta t$. On part de la formulation intégrale entre t_n et t_{n+1}:

$$x(t_{n+1}) = x(t_n) + \int_{t_n}^{t_{n+1}} f(s, x(s))ds$$

L'idée des méthodes numériques est d'approcher l'intégrale. Pour le schéma d'Euler explicite, on approxime l'intégrale par la méthode des rectangles à gauche :

$$\int_{t_n}^{t_{n+1}} f(s, x(s))ds \approx (t_{n+1} - t_n) f(t_n, x(t_n)) = \Delta t f(t_n, x(t_n))$$

En remplaçant $x(t_n)$ par son approximation x_n et $x(t_{n+1})$ par x_{n+1}, on obtient le schéma :

$$x_{n+1} = x_n + \Delta t f(t_n, x_n)$$

Avec la condition initiale $x_0 = x(t_0)$. Ceci permet de calculer $x_1, x_2, \ldots, x_N$ de proche en proche.

Étapes de la construction

1. **Étape 1: Maillage du domaine**
   On choisit $N \in \mathbb{N}^*$ et on définit le pas de temps $\Delta t = (T - t_0)/N$. Les points de discrétisation sont $t_n = t_0 + n\Delta t$ pour $n = 0, 1, \ldots, N$.

2. **Étape 2: Formulation Intégrale**
   Sur chaque sous-intervalle $[t_n, t_{n+1}]$, la solution exacte vérifie :

   $$x(t_{n+1}) = x(t_n) + \int_{t_n}^{t_{n+1}} f(s, x(s))ds$$

3. **Étape 3: Approximation des Intégrales (Formules de Quadrature)**
   On remplace l'intégrale par une formule de quadrature et $x(t_k)$ par x_k.

Approximation des Intégrales (Formules de Quadrature)

Schéma d'Euler Explicite (Rectangles à gauche) On approxime $f(s, x(s))$ par sa valeur en t_n, soit $f(t_n, x_n)$.

$$\int_{t_n}^{t_{n+1}} f(s, x(s))ds \approx \Delta t f(t_n, x(t_n))$$

Le schéma est :

$$x_{n+1} = x_n + \Delta t f(t_n, x_n), \quad n = 0, \ldots, N-1$$

L'erreur locale est en $O(\Delta t^2)$.

Schéma du Point Milieu On approxime l'intégrale par la valeur au point milieu $t_n + \Delta t/2$.

$$\int_{t_n}^{t_{n+1}} f(s, x(s))ds \approx \Delta t f(t_n + \Delta t/2, x(t_n + \Delta t/2))$$

On a besoin d'approcher $x(t_n + \Delta t/2)$. On peut utiliser une étape d'Euler : $x(t_n + \Delta t/2) \approx x(t_n) + (\Delta t/2) f(t_n, x(t_n)) \approx x_n + (\Delta t/2) f(t_n, x_n)$. Le schéma devient :

$$x_{n+1} = x_n + \Delta t f\left(t_n + \frac{\Delta t}{2}, x_n + \frac{\Delta t}{2} f(t_n, x_n)\right), \quad n = 0, \ldots, N-1$$

L'erreur locale est en $O(\Delta t^3)$.

Schéma des Trapèzes (Implicite) On approxime l'intégrale par la moyenne des valeurs aux extrémités :

$$\int_{t_n}^{t_{n+1}} f(s, x(s))ds \approx \frac{\Delta t}{2}[f(t_n, x(t_n)) + f(t_{n+1}, x(t_{n+1}))]$$

Le schéma est :

$$x_{n+1} = x_n + \frac{\Delta t}{2}[f(t_n, x_n) + f(t_{n+1}, x_{n+1})], \quad n = 0, \ldots, N-1$$

Ce schéma est implicite car x_{n+1} apparaît des deux côtés de l'équation (dans $f(t_{n+1}, x_{n+1})$). Il faut résoudre une équation (souvent non linéaire) à chaque étape pour trouver x_{n+1}. L'erreur locale est en $O(\Delta t^3)$.

Autres schémas et forme générale des schémas explicites

Schéma d'Euler Implicite (Rectangle à droite) On approxime l'intégrale par la valeur à l'extrémité droite t_{n+1}.

$$\int_{t_n}^{t_{n+1}} f(s, x(s))ds \approx \Delta t f(t_{n+1}, x(t_{n+1}))$$

Le schéma est :

$$x_{n+1} = x_n + \Delta t f(t_{n+1}, x_{n+1}), \quad n = 0, \ldots, N-1$$

C'est un schéma implicite.

Schéma de Crank-Nicolson C'est un autre nom pour le schéma des trapèzes vu précédemment.

$$x_{n+1} = x_n + \frac{\Delta t}{2}[f(t_n, x_n) + f(t_{n+1}, x_{n+1})]$$

Il est implicite.

Schéma de Heun (Euler amélioré / Runge-Kutta d'ordre 2) C'est un schéma explicite qui combine une étape de prédiction (type Euler explicite) et une étape de correction (type trapèze utilisant la prédiction).

- Prédiction : $\tilde{x}_{n+1} = x_n + \Delta t f(t_n, x_n)$
- Correction : $x_{n+1} = x_n + \frac{\Delta t}{2}[f(t_n, x_n) + f(t_{n+1}, \tilde{x}_{n+1})]$

Le schéma s'écrit en une seule ligne :

$$x_{n+1} = x_n + \frac{\Delta t}{2}[f(t_n, x_n) + f(t_{n+1}, x_n + \Delta t f(t_n, x_n))], \quad n = 0, \ldots, N-1$$

Ce schéma est explicite. L'erreur locale est en $O(\Delta t^3)$.

Généralisation des schémas à un pas (explicites) Un schéma numérique à un pas explicite peut s'écrire sous la forme générale :

$$x_{n+1} = x_n + \Delta t \Phi(t_n, x_n, \Delta t), \quad n = 0, \ldots, N-1$$

avec x_0 donné, et où $\Phi : \mathbb{R} \times \mathbb{R}^d \times \mathbb{R}^+ \to \mathbb{R}^d$ est appelée fonction d'incrément.

- **Euler explicite :** $\Phi(t, y, \Delta t) = f(t, y)$
- **Point Milieu :** $\Phi(t, y, \Delta t) = f(t + \frac{\Delta t}{2}, y + \frac{\Delta t}{2} f(t, y))$
- **Heun :** $\Phi(t, y, \Delta t) = \frac{1}{2}[f(t, y) + f(t + \Delta t, y + \Delta t f(t, y))]$

Chapter 10

CM10

10.1 Introduction à la résolution numérique des EDO

Nous nous intéressons à la résolution numérique du problème de Cauchy pour une équation différentielle ordinaire (EDO) du premier ordre : Trouver une fonction $t \mapsto x(t)$ définie sur un intervalle $[t_0, T]$ telle que

$$\begin{cases} x'(t) = f(t, x(t)), \quad t \in [t_0, T] \\ x(t_0) = x_0 \end{cases} \tag{10.1}$$

où $f : [t_0, T] \times \mathbb{R}^d \to \mathbb{R}^d$ est une fonction donnée et $x_0 \in \mathbb{R}^d$ est la condition initiale.

Lorsque la solution exacte $x(t)$ ne peut pas être déterminée analytiquement, on a recours à des méthodes numériques pour calculer des approximations x_n de $x(t_n)$ en des points de discrétisation $t_n = t_0 + nh$, $n = 0, 1, \ldots, N$, où $h = (T - t_0)/N$ est le pas de temps.

10.2 Schémas numériques à un pas

Definition 10.2.1 (Schéma numérique à un pas)**.** Un schéma numérique à un pas pour l'approximation de (10.1) est une méthode de la forme :

$$\begin{cases} x_{n+1} = x_n + h\Phi(t_n, x_n, h), \quad n = 0, 1, \ldots, N-1 \\ x_0 \text{ donné (généralement } x_0 = x(t_0)) \end{cases} \tag{P}$$

où $\Phi : [t_0, T] \times \mathbb{R}^d \times [0, h_0] \to \mathbb{R}^d$ est la fonction d'incrément qui caractérise le schéma, et h est le pas de temps. Le schéma génère une suite de points $(x_n)_{n=0}^N$ qui approxime la solution $(x(t_n))_{n=0}^N$.

La question centrale est de savoir si la solution numérique (x_n) converge vers la solution exacte $x(t)$ lorsque le pas h tend vers 0. Pour cela, on analyse généralement deux propriétés fondamentales du schéma : la consistance et la stabilité.

10.2.1 Exemples de schémas à un pas

Example 10.2.2 (Schéma d'Euler explicite)**.** Le schéma d'Euler explicite est obtenu en choisissant $\Phi(t, x, h) = f(t, x)$. La formule itérative est :

$$x_{n+1} = x_n + hf(t_n, x_n)$$

Considérons l'EDO $x'(t) = x(t)^2$ avec $x(0) = x_0$. Le schéma d'Euler explicite s'écrit :

$$x_{n+1} = x_n + hx_n^2$$

avec $t_n = nh$.

Example 10.2.3 (Schéma du Point Milieu (Runge-Kutta d'ordre 2))**.** Le schéma du Point Milieu est défini par la fonction d'incrément :

$$\Phi(t, x, h) = f\left(t + \frac{h}{2}, x + \frac{h}{2}f(t, x)\right)$$

La formule itérative est :

$$x_{n+1} = x_n + hf\left(t_n + \frac{h}{2}, x_n + \frac{h}{2}f(t_n, x_n)\right)$$

10.3 Analyse des schémas numériques

Pour choisir un schéma pour une EDO, on procède en 2 étapes :

1. On cherche un schéma qui soit *consistant* (l'erreur locale est petite).
2. Parmi les schémas consistants, on regarde ceux qui sont *stables* (l'erreur ne s'amplifie pas démesurément).

Un théorème fondamental (souvent attribué à Lax pour les EDP, mais valable pour les EDO sous certaines conditions) stipule que pour une classe de problèmes bien posés et de schémas, Consistance + Stabilité $\iff$ Convergence.

10.3.1 Consistance

La consistance mesure à quel point le schéma numérique approxime l'équation différentielle originale lorsque le pas h tend vers 0.

Definition 10.3.1 (Consistance)**.** Un schéma à un pas (P) défini par la fonction Φ est dit **consistant** avec l'EDO $x' = f(t, x)$ si

$$\Phi(t, x, 0) = f(t, x)$$

pour tout (t, x) pertinent.

Definition 10.3.2 (Erreur de consistance locale)**.** Soit $x(t)$ la solution exacte de (10.1). L'erreur de consistance locale (ou erreur de troncature locale) du schéma (P) au point t_n est définie par :

$$\epsilon(t_n, h) = \frac{x(t_{n+1}) - x(t_n)}{h} - \Phi(t_n, x(t_n), h)$$

Alternativement, pour un t générique :

$$\epsilon(t, h) = \frac{x(t + h) - x(t)}{h} - \Phi(t, x(t), h)$$

L'erreur de consistance locale mesure de combien la solution exacte échoue à satisfaire la relation du schéma numérique.

Definition 10.3.3 (Ordre de consistance)**.** Un schéma (P) est dit consistant d'ordre p si, pour toute solution $x(t)$ suffisamment régulière de l'EDO, il existe une constante $C > 0$ indépendante de h (mais pouvant dépendre de x et de l'intervalle de temps) telle que :

$$\|\epsilon(t,h)\| \leq Ch^p$$

pour tout $t \in [t_0, T]$ et pour h suffisamment petit. Si $p \geq 1$, le schéma est consistant.

Example 10.3.4 (Consistance du schéma d'Euler explicite)**.** Pour le schéma d'Euler explicite, $\Phi(t,x,h) = f(t,x)$.

1. Consistance : $\Phi(t,x,0) = f(t,x)$. Le schéma est donc consistant.
2. Ordre : Supposons que la solution exacte $x(t)$ est de classe C^2. Par développement de Taylor de $x(t+h)$ autour de t :
$$x(t+h) = x(t) + hx'(t) + \frac{h^2}{2}x''(t^*)$$
pour un certain $t^* \in (t, t+h)$. Comme $x'(t) = f(t, x(t))$, on a :
$$\frac{x(t+h) - x(t)}{h} = f(t, x(t)) + \frac{h}{2}x''(t^*)$$
L'erreur de consistance locale est :
$$\begin{aligned}\epsilon(t,h) &= \frac{x(t+h) - x(t)}{h} - \Phi(t, x(t), h) \\ &= \left(f(t, x(t)) + \frac{h}{2}x''(t^*)\right) - f(t, x(t)) \\ &= \frac{h}{2}x''(t^*)\end{aligned}$$
Si $x \in C^2([t_0, T])$, alors x'' est bornée sur $[t_0, T]$, disons par M_2. Donc,
$$\|\epsilon(t,h)\| \leq \frac{M_2}{2}h$$
Le schéma d'Euler explicite est donc d'ordre $p = 1$.

Example 10.3.5 (Consistance du schéma du Point Milieu)**.** Pour le schéma du Point Milieu, $\Phi(t,x,h) = f\left(t + \frac{h}{2}, x + \frac{h}{2}f(t,x)\right)$.

1. Consistance : $\Phi(t,x,0) = f(t+0, x+0f(t,x)) = f(t,x)$. Le schéma est consistant.
2. Ordre : Supposons $x(t)$ de classe C^3 et f suffisamment régulière. On a $x(t+h) = x(t) + hx'(t) + \frac{h^2}{2}x''(t) + \frac{h^3}{6}x'''(t) + O(h^4)$. Et par développement de Taylor de $\Phi(t, x(t), h)$ par rapport à h autour de $h = 0$:
$$\begin{aligned}\Phi(t, x(t), h) &= f\left(t + \frac{h}{2}, x(t) + \frac{h}{2}f(t, x(t))\right) \\ &= f(t, x(t)) + \frac{h}{2}\frac{\partial f}{\partial t}(t, x(t)) + \frac{h}{2}f(t, x(t))\frac{\partial f}{\partial x}(t, x(t)) + O(h^2) \\ &= f(t, x(t)) + \frac{h}{2}\left(\frac{\partial f}{\partial t} + f\frac{\partial f}{\partial x}\right)(t, x(t)) + O(h^2)\end{aligned}$$

Comme $x'(t) = f(t, x(t))$ et $x''(t) = \frac{d}{dt} f(t, x(t)) = \frac{\partial f}{\partial t} + \frac{\partial f}{\partial x} x'(t) = \frac{\partial f}{\partial t} + \frac{\partial f}{\partial x} f$, on a :

$$\Phi(t, x(t), h) = x'(t) + \frac{h}{2} x''(t) + O(h^2)$$

L'erreur de consistance locale est :

$$\begin{aligned}
\epsilon(t, h) &= \frac{x(t+h) - x(t)}{h} - \Phi(t, x(t), h) \\
&= \frac{1}{h} \left(h x'(t) + \frac{h^2}{2} x''(t) + \frac{h^3}{6} x'''(t) + O(h^4) \right) - \left(x'(t) + \frac{h}{2} x''(t) + O(h^2) \right) \\
&= \left(x'(t) + \frac{h}{2} x''(t) + \frac{h^2}{6} x'''(t) + O(h^3) \right) - \left(x'(t) + \frac{h}{2} x''(t) + O(h^2) \right) \\
&= \frac{h^2}{6} x'''(t) + O(h^3)
\end{aligned}$$

(Un calcul plus détaillé montrerait que le terme $O(h^2)$ dans le développement de Φ contribue aussi). L'erreur locale est en $O(h^2)$. Le schéma du Point Milieu est donc d'ordre $p = 2$.

10.3.2 Stabilité

La stabilité concerne le comportement des erreurs au cours des itérations. Un schéma stable est un schéma pour lequel les petites perturbations (comme les erreurs d'arrondi ou l'erreur de consistance locale) ne sont pas amplifiées de manière excessive.

Definition 10.3.6 (Stabilité (au sens de Lipschitz))**.** Un schéma à un pas (P) est dit **stable** si sa fonction d'incrément Φ est uniformément Lipschitzienne par rapport à sa deuxième variable, c'est-à-dire s'il existe une constante $L_\Phi \geq 0$ (indépendante de t, y_1, y_2, h) telle que :

$$\|\Phi(t, y_1, h) - \Phi(t, y_2, h)\| \leq L_\Phi \|y_1 - y_2\|$$

pour tout $t \in [t_0, T]$, tous $y_1, y_2 \in \mathbb{R}^d$ et tout $h \in [0, h_0]$.

Cette définition implique que si l'on considère deux suites (x_n) et (y_n) générées par le même schéma mais avec des conditions initiales légèrement différentes x_0 et y_0, alors la différence $\|x_n - y_n\|$ reste contrôlée par la différence initiale $\|x_0 - y_0\|$.

Proposition 10.3.7 (Stabilité du schéma d'Euler explicite)**.** Si la fonction $f(t, x)$ est Lipschitzienne par rapport à x sur $[t_0, T] \times \mathbb{R}^d$, uniformément en t, avec une constante de Lipschitz L, i.e.,

$$\|f(t, x_1) - f(t, x_2)\| \leq L \|x_1 - x_2\|, \quad \forall t \in [t_0, T], \forall x_1, x_2 \in \mathbb{R}^d,$$

alors le schéma d'Euler explicite $x_{n+1} = x_n + h f(t_n, x_n)$ est stable pour tout $h > 0$.

Preuve. La fonction d'incrément est $\Phi(t, x, h) = f(t, x)$. Nous devons vérifier la condition de Lipschitz de la Définition 10.3.6. Pour $t \in [t_0, T]$, $y_1, y_2 \in \mathbb{R}^d$ et $h \geq 0$:

$$\|\Phi(t, y_1, h) - \Phi(t, y_2, h)\| = \|f(t, y_1) - f(t, y_2)\|$$

Par l'hypothèse de Lipschitzianité de f, on a :

$$\|f(t, y_1) - f(t, y_2)\| \leq L \|y_1 - y_2\|$$

Donc, $\|\Phi(t, y_1, h) - \Phi(t, y_2, h)\| \leq L \|y_1 - y_2\|$. La condition de la Définition 10.3.6 est satisfaite avec $L_\Phi = L$. Le schéma d'Euler explicite est stable.

De plus, considérons l'évolution de la différence entre deux solutions numériques x_n et y_n : $x_{n+1} - y_{n+1} = (x_n - y_n) + h(f(t_n, x_n) - f(t_n, y_n))$. En prenant la norme :

$$\begin{aligned}\|x_{n+1} - y_{n+1}\| &\leq \|x_n - y_n\| + h\|f(t_n, x_n) - f(t_n, y_n)\| \\ &\leq \|x_n - y_n\| + hL\|x_n - y_n\| \\ &= (1 + hL)\|x_n - y_n\|\end{aligned}$$

Soit $e_n = \|x_n - y_n\|$. On a la récurrence $e_{n+1} \leq (1 + hL)e_n$. Par induction :

$$e_n \leq (1 + hL)^n e_0$$

En utilisant l'inégalité $1 + x \leq e^x$ pour $x = hL$, on obtient :

$$e_n \leq (e^{hL})^n e_0 = e^{nhL} e_0$$

Comme $t_n = t_0 + nh$, on a $nh = t_n - t_0 \leq T - t_0$. Donc :

$$\|x_n - y_n\| \leq e^{L(T-t_0)}\|x_0 - y_0\|$$

La différence entre les solutions numériques reste bornée par la différence initiale, avec une constante $S = e^{L(T-t_0)}$ qui ne dépend pas de n ou h (pour $nh \leq T - t_0$). Ceci confirme la stabilité du schéma. □

10.3.3 Convergence

La convergence signifie que la solution numérique tend vers la solution exacte lorsque le pas h tend vers 0. Le résultat principal relie la consistance et la stabilité à la convergence.

Theorem 10.3.8 (Convergence des schémas à un pas)**.** Soit (10.1) un problème de Cauchy où f est continue et Lipschitzienne par rapport à x (avec constante L). Soit (P) un schéma numérique défini par Φ. Si le schéma est :

1. **Consistant** d'ordre $p \geq 1$ (c'est-à-dire $\|\epsilon(t, h)\| \leq Ch^p$).
2. **Stable** (c'est-à-dire Φ est Lipschitzienne par rapport à sa deuxième variable avec constante L_Φ).

Alors le schéma est **convergent** d'ordre p. C'est-à-dire que l'erreur globale $e_n = x(t_n) - x_n$ satisfait :

$$\max_{0 \leq n \leq N} \|e_n\| = \max_{0 \leq n \leq N} \|x(t_n) - x_n\| \leq S \left(\|e_0\| + (T - t_0) \max_{0 \leq k < N} \|\epsilon(t_k, h)\| \right)$$

où S est une constante liée à la stabilité (par exemple $S \approx e^{L_\Phi(T-t_0)}$). Plus précisément, il existe une constante K (indépendante de h) telle que :

$$\max_{0 \leq n \leq N} \|x(t_n) - x_n\| \leq K(\|x(t_0) - x_0\| + h^p)$$

En particulier, si $x_0 = x(t_0)$, alors l'erreur globale est en $O(h^p)$.

Preuve (Idée de la preuve)**.** L'erreur globale $e_n = x(t_n) - x_n$ satisfait une relation de récurrence. On a $x(t_{n+1}) = x(t_n) + h\Phi(t_n, x(t_n), h) + h\epsilon(t_n, h)$ par définition de l'erreur locale ϵ. Et $x_{n+1} = x_n + h\Phi(t_n, x_n, h)$. En soustrayant : $e_{n+1} = x(t_{n+1}) - x_{n+1} = (x(t_n) - x_n) + h(\Phi(t_n, x(t_n), h) - \Phi(t_n, x_n, h)) + h\epsilon(t_n, h)$. $e_{n+1} = e_n + h(\Phi(t_n, x_n + e_n, h) - \Phi(t_n, x_n, h)) + h\epsilon(t_n, h)$. En prenant la norme et en utilisant la stabilité (Lipschitzianité de Φ) : $\|e_{n+1}\| \leq \|e_n\| + hL_\Phi\|e_n\| + h\|\epsilon(t_n, h)\| = (1 + hL_\Phi)\|e_n\| + h\|\epsilon_n\|$. où $\|\epsilon_n\| = \|\epsilon(t_n, h)\| \leq Ch^p$. $\|e_{n+1}\| \leq (1 + hL_\Phi)\|e_n\| + Ch^{p+1}$. En utilisant un lemme de Gronwall discret, on peut montrer que $\|e_n\|$ est majoré par une quantité de l'ordre de $\|e_0\| + (T - t_0)Ch^p/L_\Phi$

(grossièrement), ce qui donne la convergence d'ordre p. La majoration indiquée dans le théorème provient directement de l'application de ce lemme. Par exemple, $\max_{0\leq n\leq N}\|e_n\| \leq e^{L_\Phi(T-t_0)}\|e_0\| + \frac{e^{L_\Phi(T-t_0)}-1}{L_\Phi}\max_{0\leq k<N}\|\epsilon_k\|$. □

Ce théorème est fondamental car il garantit que si l'on choisit un schéma stable qui approxime bien localement l'équation différentielle (consistant), alors la solution numérique convergera globalement vers la solution exacte lorsque le pas de discrétisation diminue.

Chapter 11

CM11

11.1 Méthodes Itératives de Résolution d'Équations

11.1.1 Introduction

Ce chapitre aborde les méthodes itératives pour la résolution numérique d'équations, en particulier les équations non linéaires de la forme $f(x) = 0$. Nous explorerons les concepts fondamentaux, les définitions clés, et des algorithmes comme la dichotomie et la méthode de la fausse position.

11.1.2 Généralités et exemples d'équations récurrentes (EO) : f(x)=0

Une équation récurrente définit une suite où chaque terme est fonction des termes précédents. Un cas fondamental est la recherche des racines d'une fonction f, c'est-à-dire la résolution de l'équation $f(x) = 0$. (EO) : Chercher $x \in \mathbb{R}$ tel que $f(x) = 0$. L'ensemble des solutions (racines) est noté $R(f) = \{x \in \mathbb{R} \mid f(x) = 0\}$.

Définitions

Definition 11.1.1 (Équation non linéaire)**.** Si $f : I \subseteq \mathbb{R} \to \mathbb{R}$ est une fonction, l'équation $f(x) = 0$ est appelée équation non linéaire. (Ici, f est au moins de classe C^0).

Definition 11.1.2 (Solution)**.** Tout $x^* \in I$ tel que $f(x^*) = 0$ est une solution de (EO) ou une racine de f.

Definition 11.1.3 (Racine simple et multiple)**.** Soit f une fonction de classe $C^r(I)$ pour $r \geq 1$.

- Si $f(a) = 0$ et $f'(a) \neq 0$, a est dite racine simple de f.
- Si $f(a) = f'(a) = ... = f^{(r-1)}(a) = 0$ et $f^{(r)}(a) \neq 0$, alors a est une racine de multiplicité r. On peut alors écrire $f(x) = (x - a)^r g(x)$ où $g(a) \neq 0$.
- Lorsque $r = 1$, on a une racine simple. C'est le cas si $f'(a) \neq 0$.
- Si $f(x) = ax + b$, l'équation est dite linéaire.

Exemples

- Schéma d'Euler implicite pour une EDO $x'(t) = F(t, x(t))$. En discrétisant le temps $t_n = n\Delta t$, on cherche $x_{n+1} \approx x(t_{n+1})$. Le schéma d'Euler implicite est donné par :

$$\frac{x_{n+1} - x_n}{\Delta t} = F(t_{n+1}, x_{n+1})$$

Pour déterminer x_{n+1}, il faut résoudre l'équation (souvent non linéaire) :

$$g_n(z) = 0 \quad \text{où} \quad g_n(z) = z - x_n - \Delta t F(t_{n+1}, z)$$

Ceci est une équation de la forme $g_n(x) = 0$ à résoudre à chaque pas de temps.

- Une suite récurrente est définie par $x_{n+1} = f(x_n)$ pour $n = 0, 1, ...$ avec x_0 donné. Si la suite $\{x_n\}$ converge vers une limite x^*, et si f est continue, alors x^* est un point fixe de f, c'est-à-dire $f(x^*) = x^*$. La recherche de points fixes est liée à la résolution de $g(x) = x - f(x) = 0$.

11.1.3 Partition correcte du problème (EO)

On se propose de vérifier si :

- (EO) admet une solution.
- Si oui, cette solution est-elle unique ?
- La solution dépend continûment des données du problème (stabilité).
- La solution est suffisamment régulière.

Pour répondre à cela, on va se placer dans le cadre macroscopique $f : \mathbb{R} \to \mathbb{R}$.

Proposition 11.1.4 (Existence - Cas $f(x) = 0$)**.** Soit $I = [a, b]$ un intervalle de $\mathbb{R}$ et $f : I \to \mathbb{R}$ une fonction continue. Si $f(a)f(b) < 0$, alors il existe $x^* \in]a, b[$ tel que $f(x^*) = 0$. De plus, si f est strictement monotone sur I, alors x^* est unique.

Preuve (Preuve (idée))**.** Théorème des valeurs intermédiaires. L'unicité découle de la stricte monotonie. □

Proposition 11.1.5 (Point fixe - Cas $x = g(x)$)**.** Soit $I \subset \mathbb{R}$ un intervalle fermé et $g : I \to \mathbb{R}$ une fonction continue.

- Si $g(I) \subseteq I$ (c'est-à-dire, si $x \in I$, alors $g(x) \in I$), alors il existe au moins un point fixe $x^* \in I$ tel que $g(x^*) = x^*$.
- Si de plus g est contractante sur I, c'est-à-dire s'il existe $k \in [0, 1)$ tel que $|g(x) - g(y)| \leq k|x - y|$ pour tous $x, y \in I$, alors le point fixe x^* est unique.

Remark 11.1.6. Lorsque la racine cherchée x^* de $f(x) = 0$ est de multiplicité $r > 1$, $f'(x^*) = 0$. Numériquement, le problème de trouver x^* peut être difficile (convergence lente, instabilité). Il faudra faire attention dans ce cas.

11.1.4 Construction de schémas pour (EO)

Méthode de dichotomie

S'applique au cas $f(x) = 0$.

Principe

On part d'un intervalle $[a_0, b_0]$ tel que $f(a_0)f(b_0) < 0$. D'après le théorème des valeurs intermédiaires, il existe au moins une racine dans $]a_0, b_0[$. L'idée est de diviser l'intervalle par 2 à chaque étape et de conserver la moitié qui contient une racine.

1. Calculer le milieu $c_0 = \frac{a_0+b_0}{2}$.
2. Évaluer $f(c_0)$.
3. Si $f(a_0)f(c_0) < 0$, alors une racine se trouve dans $]a_0, c_0[$. On pose $[a_1, b_1] = [a_0, c_0]$.
4. Si $f(c_0)f(b_0) < 0$, alors une racine se trouve dans $]c_0, b_0[$. On pose $[a_1, b_1] = [c_0, b_0]$.
5. Si $f(c_0) = 0$, alors c_0 est une racine. L'algorithme s'arrête.

On répète le processus sur l'intervalle $[a_1, b_1]$, et ainsi de suite.

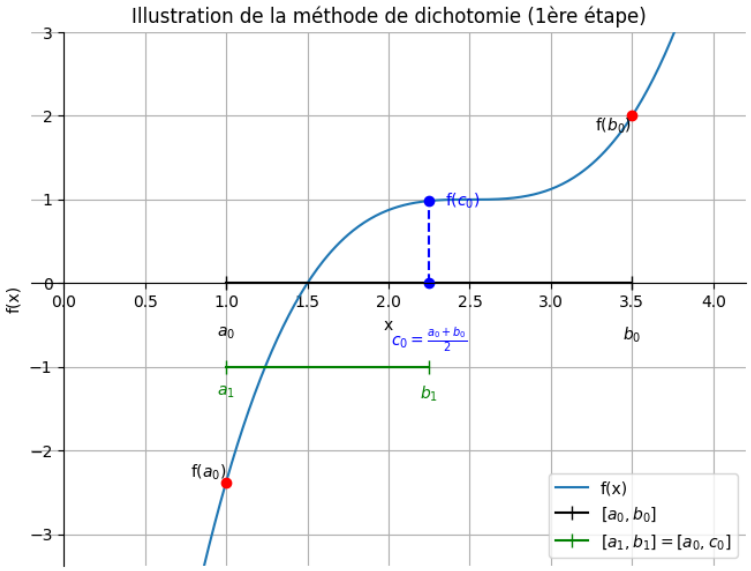

Figure 11.1: Illustration graphique de la méthode de dichotomie.

11.1.5 Algorithme de Dichotomie

Initialisation :

- Choisir a_0, b_0 tels que $f(a_0)f(b_0) < 0$.
- Choisir une tolérance $\epsilon > 0$ (critère d'arrêt).
- Choisir un nombre maximal d'itérations k_{max}.
- Poser $a = a_0, b = b_0, k = 0$.

Itération : Tant que $(b - a) > \epsilon$ et $k < k_{max}$ faire :

1. Calculer le milieu : $c = \frac{a+b}{2}$.
2. Évaluer $f(c)$.
3. Si $f(c) == 0$, arrêter (solution trouvée $x^* = c$).
4. Si $f(a)f(c) < 0$, alors la racine est dans $[a, c]$. Mettre à jour $b = c$.
5. Sinon (si $f(c)f(b) < 0$), la racine est dans $[c, b]$. Mettre à jour $a = c$.

6. Incrémenter $k = k + 1$.

Sortie : L'approximation de la racine est $\frac{a+b}{2}$.

Coût par itération :

- 1 addition et 1 division (pour calculer c).
- 1 évaluation de la fonction f.
- 1 test (comparaison de signes).

11.1.6 Convergence de la Dichotomie

Soient (a_n) et (b_n) les suites des bornes des intervalles générés par l'algorithme de dichotomie.

- $[a_{n+1}, b_{n+1}] \subset [a_n, b_n]$ pour tout n.
- La suite (a_n) est croissante et majorée par b_0.
- La suite (b_n) est décroissante et minorée par a_0.
- Les suites (a_n) et (b_n) sont donc convergentes.
- La longueur de l'intervalle $[a_n, b_n]$ est $L_n = b_n - a_n$. On a $L_{n+1} = \frac{1}{2}L_n$.
- Donc, $L_n = \frac{b_0 - a_0}{2^n}$.
- $\lim_{n\to\infty}(b_n - a_n) = \lim_{n\to\infty} \frac{b_0-a_0}{2^n} = 0$.

Puisque $\lim(b_n - a_n) = 0$, les suites (a_n) et (b_n) sont adjacentes. Elles convergent donc vers la même limite x^*. Comme $f(a_n)f(b_n) \leq 0$ pour tout n, et f est continue, en passant à la limite, on obtient $f(x^*)f(x^*) \leq 0$, soit $(f(x^*))^2 \leq 0$. Ceci implique $f(x^*) = 0$. Donc, $\lim_{n\to\infty} a_n = \lim_{n\to\infty} b_n = x^*$, où x^* est une racine de f.

Estimation de l'erreur : Si $c_n = \frac{a_n+b_n}{2}$ est l'approximation de x^* à l'étape n, alors $x^* \in [a_n, b_n]$. L'erreur est majorée par la demi-longueur de l'intervalle :

$$|c_n - x^*| \leq \frac{b_n - a_n}{2} = \frac{b_0 - a_0}{2^{n+1}}$$

La convergence est dite linéaire avec un taux de 1/2. C'est une convergence garantie mais qui peut être lente ("beaucoup de subdivisions").

11.1.7 Méthode de la Fausse Position (Regula Falsi)

Cette méthode est similaire à la dichotomie, mais au lieu de couper l'intervalle en deux, elle utilise l'intersection de la droite reliant $(a, f(a))$ et $(b, f(b))$ avec l'axe des abscisses pour déterminer le point suivant c.

Principe : L'équation de la droite passant par $(a, f(a))$ et $(b, f(b))$ est :

$$y - f(b) = \frac{f(b) - f(a)}{b - a}(x - b)$$

On cherche l'intersection avec l'axe des x, c'est-à-dire $y = 0$. On note c l'abscisse de ce point.

$$-f(b) = \frac{f(b) - f(a)}{b - a}(c - b)$$

$$c = b - f(b)\frac{b - a}{f(b) - f(a)}$$

Comme pour la dichotomie, on choisit le nouvel intervalle $[a, c]$ ou $[c, b]$ en fonction du signe de $f(a)f(c)$ ou $f(c)f(b)$.

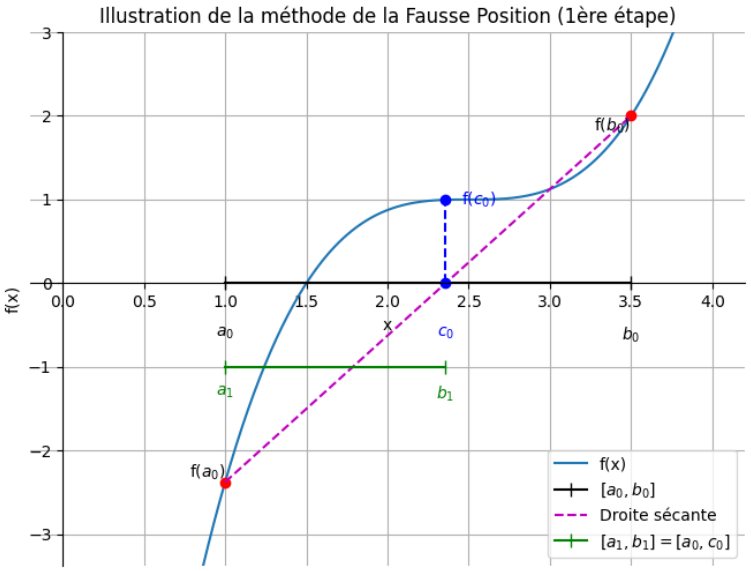

Figure 11.2: Illustration graphique de la méthode de la Fausse Position.

Algorithme : Similaire à la dichotomie, mais remplacer le calcul de c par:

$$c = b - f(b)\frac{b-a}{f(b)-f(a)}$$

Le coût par itération est plus élevé : 1 soustraction, 1 multiplication, 1 division, 2 évaluations de f (si $f(a)$ et $f(b)$ ne sont pas stockées), et les opérations arithmétiques pour c. (Note: on peut optimiser pour ne recalculer qu'une valeur de f par itération). Coût mentionné dans les notes: 1 produit, 2 additions/soustractions, 1 division, 1 évaluation de f.

11.1.8 Convergence de la Fausse Position

- Si $f \in C^2([a_0, b_0])$, $f(a_0)f(b_0) < 0$.
- Si f'' n'a aucune racine dans $]a_0, b_0[$ (i.e., f est soit strictement convexe, soit strictement concave sur l'intervalle), alors l'une des bornes (a_n) ou (b_n) deviendra stationnaire (constante à partir d'un certain rang), tandis que l'autre convergera vers la racine x^*.

Proposition 11.1.7 (Convergence linéaire)**.** Sous les hypothèses ci-dessus (f de classe C^2, $f(a_0)f(b_0) < 0$, et f'' ne s'annule pas sur $]a_0, b_0[$), la suite des approximations (c_n) générées par la méthode de la fausse position converge linéairement vers l'unique racine x^* de f dans $[a_0, b_0]$. De plus, si par exemple (a_n) est constante à partir d'un certain rang ($a_n = a$), alors (b_n) converge vers x^* et :

$$\lim_{n\to\infty} \frac{|x^* - b_{n+1}|}{|x^* - b_n|} = \left|1 - \frac{f'(x^*)}{f'(\xi_n)} \frac{x^* - a}{f(x^*) - f(a)}\right|$$

(Note: L'expression exacte du taux peut varier, celle-ci est transcrite des notes, mais semble inhabituelle. Typiquement, la convergence linéaire est établie, mais le taux dépend de la courbure et de quelle borne

est fixe). Si (b_n) est constante ($b_n = b$), alors (a_n) converge vers x^* et:

$$\lim_{n\to\infty} \frac{|x^* - a_{n+1}|}{|x^* - a_n|} = \left|1 - \frac{f'(x^*)}{f'(\eta_n)} \frac{x^* - b}{f(x^*) - f(b)}\right|$$

où ξ_n, η_n proviennent de l'application du théorème des accroissements finis.

En général, la convergence de la fausse position est linéaire, mais souvent plus rapide que la dichotomie dans les premières itérations. Cependant, elle peut devenir très lente si une des bornes reste bloquée loin de la racine.