Soutien 12 juin 2019 13h30 - 15h30

Vous venez d'achever le cours sur les thèmes listés ci-dessous.

Vous êtes supposés avoir compris certaines notions rappelées ci-dessous.

Si certaines d'entre elles ne vous semblent pas suffisamment claires, n'hésitez pas à solliciter l'enseignant en salle avec vous, et assurez-vous de les avoir bien comprises.

Contents

1 Interpolation polynomiale

1.1 Motivations

Note 1 (Vous devez avoir compris)**.**

1. à quoi sert l'interpolation polynômiale.
2. des utilisations potentielles ou concrètes de l'interpolation polynômiale.

Note 2 (Exercice d'exploration)**.**

Quelle différence y'a-t-il entre interpolation polynômiale et approximation polynômiale.

1.2 Interpolation de Lagrange définition

Note 3 (Vous devez avoir compris)**.**

1. Ce qu'on entend par polynôme d'interpolation de Lagrange.
2. Le résultat d'existence et d'unicité du polynôme d'interpolation de Lagrange.
3. L'estimation d'erreur entre une fonction suffisamment régulière et son polynôme d'interpolation de Lagrange.
   (*La relation entre le degré du polynôme d'interpolation et l'ordre de régularité de la fonction nécessaire pour cette estimation.*)

1.3 Interpolation de Lagrange construction effective

Rappel 1 (Méthodes de construction vues)**.**

Nous avons vu dans **l'ordre** les variantes suivantes du polynôme d'interpolation .

1. Méthode des coefficients inconnus dans la base de canonique :
   ce qui a conduit à l'algorithme dit **Interpolation de Vandermonde**.
2. Méthode de coefficients indéterminés dans la base de Lagrange :
   ce qui a conduit à l'algorithme dit **Interpolation de Lagrange**.
3. Approche basée sur la recherche de récurrence dans les polynômes d'interpolation :
   ce qui a conduit à l'algorithme dit **Interpolation de Néville**.
4. Méthode des coefficients indéterminés dans la base de Newton :
   ce qui a conduit à l'algorithme dit **Interpolation de Newton**.
5. Méthode basée sur le choix convenable des points d'interpolation :
   ce qui a conduit à l'algorithme dit **Interpolation de Tchebychev**.

Un Notebook rappellant ces algorithmes est accessible sur Dokeos dans un répertoire dedié.

Note 4 (Acquis théoriques)**.**

D'un point de vue théorique, vous devez avoir compris :

1. Les raisons de cet ordre de présentation des méthodes de construction du polynôme d'interpolation
2. Pour chaque méthode
   - un algorithme de construction
   - identifier l'effort mis en place pour réduire le coût d'une évaluation du polynôme construit.
   - les avantages de la méthode
   - les inconvénients ou limites de la méthode
   - l'insuffisance de la méthode précédente à laquelle celle-ci apporte une solution.

Note 5 (Acquis pratiques)**.**

D'un point de vue pratique, vous devez pouvoir :

1. Calculer à la main, pour chaque type d'interpolation et pour des degrés bas, la valeur en un point du polynôme d'interpolation de Lagrange d'une fonction.
2. Pour chaque méthode vous devez pouvoir :
   - décrire l'algorithme de calcul et sa complexité.
   - fournir un script (en Python, non nécessairement optimisé) la mettant en oeuvre.
3. Fournir un script Python, permettant de représenter sur un même graphique, avec légende et titre :
   - une fonction donnée
   - les points d'interpolation donnés
   - le polynôme d'interpolation construit
4. Fournir un script Python, permettant de représenter graphiquement l'évolution en fonction du nombre n de points d'interpolation (qu'ils soient uniformes ou de Tchebychev) de l'erreur en norme $\|\|_\infty$ sur un intervalle $[a, b]$, entre une fonction et son polynôme d'interpolation.

2 Intégration numérique

2.1 Motivations

Note 6 (Vous devez avoir compris)**.**

1. à quoi sert l'intégration numérique,
2. des utilisations potentielles ou concrètes.

Note 7 (Exercice d'exploration)**.**

Expliquez comment l'interpolation polynomiale peut servir dans l'intégration numérique.

2.2 Construction de formules de quadratures

Note 8 (Vous devez avoir compris)**.**

1. La différence qu'il y a entre une formule de quadrature élémentaire et une formule de quadrature composite,
2. Comment une analyse d'une formule de quadrature est menée (*théoriquement, numériquement*).

2.3 Construction effective

Rappel 2 (Méthodes de construction vues)**.**

Vous devez pour chaque formule de quadrature élémentaire ci-dessous:

- expliquer la démarche de leur obtention,
- estimer l'erreur commise, en précisant la régularité nécessaire de la fonction à intégrer,
- fournir la formule composite associée ainsi que sont estimation d'erreur.

Voici ces formules élémentaires par leur nom :

- Formule des rectangles à gauche
- Formule des rectangles à droite
- Formule du point milieu
- Formule des trapèzes
- Formule de Simpson,
- Formule de Gauss-Legendre à 2 points,
- Formule de Gauss-Legendre à 3 points.

Un notebook rappellant ces algorithmes est accessible sur Dokeos dans un répertoire dedié.

Note 9 (Acquis théoriques)**.**

D'un point de vue théorique, vous devez avoir compris :

1. Les raisons de cet ordre de présentation des méthodes de construction du polynôme d'interpolation
2. Pour chaque méthode
   - un algorithme de construction
   - identifier l'effort qu'on peut mettre en place pour réduire le coût.
   - les avantages de la méthode
   - les inconvénients ou limites de la méthode
   - l'insuffisance de la méthode précédente à laquelle celle-ci apporte une solution.

Note 10 (Acquis pratiques)**.**

D'un point de vue pratique, vous devez pouvoir :

1. calculer à la main une formule de quadrature sur $[-1, 1]$ puis écrire la formule composite associée,
2. pour toute formule de quadrature donnée, vous devez pouvoir :
   - décrire un algorithme de calcul et sa complexité
   - et fournir un script (en Python, non nécessairement optimisé) le mettant en oeuvre.
3. Fournir un script Python, permettant de représenter graphiquement l'évolution en fonction du nombre n de points de quadrature (qu'ils soient uniformes ou non) de l'erreur en norme $\|\|_\infty$ sur un intervalle $[a, b]$, entre la valeur exacte de l'intégrale et son calcul approché.

3 Résolution numérique des équations différentielles ordinaires (EDO)

3.1 Motivations

Note 11 (Vous devez avoir compris)**.**

1. la nécessité de résoudre numériquement les EDOs.
2. des utilisations potentielles ou concrètes.

Note 12 (Exercice d'exploration)**.**

Expliquer pourquoi on peut se limiter à l'étude des systèmes d'EDO d'ordre 1.

3.2 Position correcte d'une EDO

Note 13 (Vous devez avoir compris)**.**

1. ce qu'on entend par problème de Cauchy pour une EDO,
2. comment est-ce qu'on résout un problème de Cauchy pour une EDO.

3.3 Construction effective de schémas numériques

Rappel 3 (Méthodes de construction vues)**.**

Vous devez pour chaque schéma numérique ci-dessous:

- expliquer la démarche de son obtention,
- réécrire (dans le cas explicite à un pas) le schéma sous la forme générale
- pouvoir étudier sa consistance,
- pouvoir étudier sa stabilité (dans le cas où celui-ci est explicite).

Voici ces schémas par leur nom:

- Schéma d'Euler explicite
- Schéma d'Euler implicite
- Schéma du Point Milieu
- Schéma de Crank-Nicolson
- Schéma de Heun

Un notebook rappellant ces algorithmes est accessible sur Dokeos dans un répertoire dedié.

Note 14 (Acquis théoriques)**.**

D'un point de vue théorique, vous devez avoir compris :

1. Les raisons de cet ordre de présentation des méthodes de construction
2. Pour chaque méthode
   - un algorithme de construction
   - identifier l'effort à mettre en place pour réduire le coût.
   - les avantages de la méthode
   - les inconvénients ou limites de la méthode
   - l'insuffisance de la méthode précédente à laquelle celle-ci apporte une solution.

Note 15 (Acquis pratiques)**.**

D'un point de vue pratique, vous devez pouvoir :

1. Pour chaque méthode vous devez pouvoir :
   - décrire l'algorithme de calcul et sa complexité.
   - fournir un script (en Python, non nécessairement optimisé) la mettant en oeuvre.
2. Fournir un script Python, permettant de représenter graphiquement l'évolution en fonction du nombre n de points de discrétisation (qu'ils soient uniformes ou non) de l'erreur en norme $\|\|_\infty$ sur un intervalle $[a, b]$, entre la valeur exacte de la solution approchée.

4 Résolution numérique des équations ordinaires (EO) : $f(x) = 0$

4.1 Motivations

Note 16 (Vous devez avoir compris)**.**

1. la nécessité de résoudre numériquement les EOs.
2. des utilisations potentielles ou concrètes.

Note 17 (Exercice d'exploration)**.**

Rappeler ce qu'on entend par racine multiples et expliquer en quoi la recherche numérique de telles racine est difficile.

4.2 Position correcte d'une EO

Note 18 (Vous devez avoir compris)**.**

1. Les conditions sous lesquelles on est certain de l'existence d'une racine.
2. Les conditions sous lesquelles l'unicité de cette racine est assurée.

4.3 Construction effective de schémas numériques

Rappel 4 (Méthodes de construction vues)**.**

Vous devez pour chaque schéma numérique ci-dessous:

- expliquer la démarche de leur obtention,
- pouvoir étudier sa (vitesse de) convergence.

Voici ces schémas par leur nom:

- Méthode de dichotomie
- Méthode de fausse position
- Méthode du point fixe
- Méthode de Newton
- Méthode de la sécante

Un notebook rappellant ces algorithmes est accessible sur Dokeos dans un répertoire dedié.

Note 19 (Acquis théoriques)**.**

D'un point de vue théorique, vous devez avoir compris :

1. Les raisons de cet ordre de présentation des méthodes de construction
2. Pour chaque méthode
   - un algorithme de construction
   - identifier l'effort mis en place pour réduire le coût.
   - les avantages de la méthode
   - les inconvénients ou limites de la méthode
   - l'insuffisance de la méthode précédente à laquelle celle-ci apporte une solution.

Note 20 (Acquis pratiques)**.**

D'un point de vue pratique, vous devez pouvoir :

1. Calculer à la main, la constante asymptotique de chacune de ces méthodes.
2. Fournir une interprétation géométrique de la méthode (et ceci graphiquement).
3. Pour chaque méthode vous devez pouvoir :
   - décrire l'algorithme de calcul et sa complexité.
   - fournir un script (en Python, non nécessairement optimisé) la mettant en oeuvre.
4. Fournir un script Python, permettant de représenter graphiquement l'évolution en fonction du nombre n d'itérations l'erreur en norme $\|\|_\infty$ sur un intervalle $[a, b]$, entre la solution exacte et la solution calculée.