Macs 205, partie 1 : interpolation et quadrature
mini-projet : fonction 'Height'

Consignes

L'ensemble de votre travail devra être rassemblé dans un seul dossier portant votre nom, et archivé au format .zip ou tar.gz. Ce dossier doit contenir

- Un format Notebook R compilé sous forme html ou pdf, obtenu avec Rstudio ou jupyter notebook, contenant les réponses rédigées au questions, les figures et résultats numériques. Tous les résultats doient êre commentés.
- Toutes les fonctions utilisées (à l'exceptions des fonctions de test) dans un fichier 'mesfonctions.R'
- Trois fichiers scripts exécutable en `R` contenant votre travail sur les questions numériques : un fichier par partie, nommé `Script-partieN.R`, pour le script correspondant à la partie N. Le correcteur doit pouvoir retrouver tous vos résultats (y compris les figures, dotées d'une légende sommaire et d'un titre) en appelant le script depuis `R`. Pour cela pensez à ajouter une ligne

```
source('mesfonctions.R')
```

au début de chaque script.

Les dossiers doivent être déposés sur l'espace de partage sur Eole avant le dimanche 29 mars 2020, 23h59. Passé cette limite, chaque heure de retard vous coûtera deux points (sur vingt).

Introduction

L'étude de cas consiste à utiliser les résultats théoriques et les méthodes numériques vus en cours pour étudier une fonction `Height` , dont l'expression est inconnue. Vous disposez seulement des valeurs de la fonction `Height`, échantillonnée à intervalles réguliers, d'espacement 2^{-12}, entre $a = -0.5 + 2^{-12}$ et $b = +0.5$ inclus. On vous fournit aussi une vraie fonction, appelée `evalHeight`, qui pour n'importe réel x passé en argument, renvoie la valeur de `Height` au point connu le plus proche. Le but général du TP est double : d'une part, approcher cette fonction correctement par un polynome avec un budget limité d'évaluations ; d'autre part de calculer la surface sous la courbe, c'est-à dire d'intégrer numériquement cette fonction, toujours avec un budget limité d'évaluations.

Données, prise en main : Téléchargez le dossier `donneesEtudeCas.zip` sur Eole.

Placez-le dans votre répertoire de travail pour ce TP et décompressez l'archive, qui contient :

- Un fichier de données : `projectData.Rdata`

- Un fichier contenant des fonctions de base pour les manipuler : `evalTools.R`

Pour charger les données et les fonctions dans votre environnement de travail : assurez vous que le répertoire courant de votre console `R` est le bon (commandes `getwd()` et `setwd('chemin-d-acces')`), nettoyez votre environnement avec la commande `rm(list=ls())` puis :

```
load('projectData.Rdata')
source('evalTools.R')
```

Inspectez votre environnement de travail avec la commande `ls()`. Vous devez obtenir

```
[1] "evalHeight" "EXPO2"    "getAbsc"   "getIndex"  "GRILLE"
[6] "MESURES"
```

Ces 6 objets ne doivent pas être modifiés

- `GRILLE` et `MESURES` sont des vecteurs de taille 2^{12}. Le premier est vecteur d'abscisses : les points
$$\{-0.5 + k\,.2^{-12}, k \in \{1, \ldots, 2^{12}\}\}.$$
Le second contient les valeurs de la fonction `Height` en chaque point du vecteur `GRILLE`, c'est à dire entre $-0.5 + 2^{-12}$ et 0.5.
- `EXPO2` $= 12$ est l'exposant du pas de discrétisation (en base 2)
- `getAbsc, evalHeight, getIndex` sont des fonctions : `getIndex(x)` renvoie l'indice i de l'élément du vecteur `GRILLE` le plus proche de x ; `getAbsc(x)` renvoie l'élément `GRILLE[i]` en question. Autrement dit : si x est un nombre entre a et b, `evalX(x)` renvoie le nombre le plus proche de x qui s'écrive $-0.5+k*2^{-\,\mathrm{BETA}}$ (où BETA $= 12$, k un entier entre 1 et 2^{BETA}). Enfin `evalHeight(x)` renvoie la valeur de la fonction `Height` au point `GRILLE[i]`, c'est-à-dire `MESURES[i]`.

1 Interpolation polynomiale

1. Interpolation polynomiale simple entre $a = -0.5 + 2^{-12}$ et $b = 0.5$.
   (a) Affichez quatre exemples mettant en évidence les défauts de l'interpolation avec un degré trop bas, ou trop haut, en utilisant des noeuds équidistants et de Tchebychev. Vous tracerez sur chaque graphique le résultat de l'interpolation et le graphe de la fonction de référence.
   (b) construisez et affichez, comme précédemment, un interpolateur qui parait satisfaisant (visuellement)
   (c) Pour chaque choix de noeuds (équi-distants ou de tchebychev), quel est le rang n de la méthode qui minimise la norme infinie de l'erreur d'interpolation ? On évaluera cette erreur en appelant la fonction `evalHeight` en 10^3 points de l'intervalle $[a, b]$. On demande ici de coder une méthode pour trouver le minimiseur
2. On s'autorise maintenant à utiliser une interpolation par morceaux.
   (a) Proposez un interpolateur par morceaux qui vous parait (visuellement) raisonnable. Affichez le résultat comme dans les questions précédentes.

(b) On vous donne un budget de 90 évaluations de la fonction pour construire votre interpolateur. Proposez un jeu de paramètres (type de noeuds, ordre, nombre de sous-intervalles) qui minimise l'erreur d'interpolation sous cette contrainte de budget.
Comme à la question (1-c), dans la phase de test de l'interpolateur on a le droit à 10^3 évaluations de la fonctions en des points équi-répartis $[a, b]$ et on demande de coder une méthode qui trouve automatiquement l'interpolateur optimal.

2 Méthodes de quadrature

1. Utilisons la méthode de quadrature composite de Cavalieri-Simpson pour estimer la valeur de l'intégrale
$$I = \int_a^b f.$$
On appelle $\hat{I}_M^{simp}$ le résultat obtenu par une méthode de simpson composite avec M intervalles de même taille.
   (a) En appliquant la méthode de Simpson pour une plage de valeurs appropriée de M, donnez une première estimation "à vue d'oeil", de I et un ordre de grandeur de l'erreur commise. On affichera le graphe de $\hat{I}_M^{simp}$ en fonction de M.
   (b) On vous donne un budget de 71 évaluations de la fonction $Height$.
      i. À combien de sous-intervalles d'intégration ce budget vous donne-t-il droit ?
      ii. Donnez une approximation a posteriori de l'erreur commise avec ce budget ainsi que la valeur de $\hat{I}_M^{simp}$ associée, au niveau de précision correspondant à l'approximation de l'erreur.
2. On voudrait une méthode automatique pour choisir M aussi petit que possible étant donné une contrainte sur la précision. Ici on souhaite une précision de 10^{-4}. Écrire un algorithme utilisant une méthode d'évaluation de l'erreur a posteriori pour résoudre ce problème.
indication : on pourra initialiser la recherche à une valeur de M assez faible puis doubler M tant qu'une condition n'est pas satisfaite. Donnez alors une estimation de I à 10^{-4} près.

3 Extrapolation de Richardson

Dans cette partie, on cherche à évaluer la valeur de `Height` en -0.5.

1. Estimez la valeurs de `Height` en -0.5 en utilisant la méthode de Richardson avec un choix qui vous parait adapté de t et δ et de n (justifiez au vu de la précision de vos données). Tracez sur le même graphique en fonction de n, l'interpolation de Richardson et le résultat de l'estimateur naïf renvoyant la valeur de la fonction au plus petit point d'interpolation utilisé par Richardson.