ECN 6338 Cours 1

Introduction

William McCausland

2022-01-05

Documents et Communication

Site Github du cours

1. Diapos (code source, pdf)
2. Démonstrations
3. Lectures, exercices
4. Devoirs avec computation

Site StudiUM du cours

1. Messages
2. Forums (possiblement)
3. Documents avec droit d'auteur
4. Chargement des devoirs

Notation pour les dérivées multivariées

- Soit x un vecteur $n \times 1$, $y = f(x)$ un vecteur $m \times 1$.
- La *matrice jacobienne* $(m \times n)$ contient toutes les dérivées de première ordre:

$$f_x = \frac{\partial y}{\partial x}, \qquad \text{où} \qquad \left[\frac{\partial y}{\partial x}\right]_{ij} = \frac{\partial y_i}{\partial x_j}.$$

- Le *gradient* est un cas spécial du Jacobien où y est scalaire, un vecteur ligne $1 \times n$.
- La *matrice hessienne* $(n \times n)$ contient toutes les dérivées de deuxième ordre quand y est scalaire:

$$f_{xx} = \frac{\partial}{\partial x}\left(\frac{\partial y}{\partial x}\right)^\top = \frac{\partial^2 y}{\partial x \partial x^\top}.$$

- La notation ci-haut suit la convention "numerator layout" ici

Quelques propriétés des dérivées multivariées

À la même page il y a des tableaux de propriétés, telles que les suivantes:

- Pour les matrices A constantes, $m \times n$,

$$\frac{\partial A x}{x} = A,$$

- Pour les vecteurs $u(x)$ et $v(x)$, $n \times 1$,

$$\frac{\partial u^\top v}{\partial x} = u^\top \frac{\partial v}{\partial x} + v^\top \frac{\partial u}{\partial x}.$$

- Pour $z = g(y)$,

$$\frac{\partial z}{\partial x} = \frac{\partial g(y)}{\partial y} \frac{\partial y}{\partial x}$$

Analyse de l'erreur

- Deux sources d'erreur numérique.
  - Précision finie des nombres réels
  - Troncation de calculs infinie
- Les deux sources d'erreur propagent et

La représentation virgule flottante

L'ordinateur représente un nombre réel x comme

$$x = s \times m \times 2^e,$$

où

- $s \in \{-1, 1\}$ est la signe,
- $m \in \mathbb{N}$ est la mantisse et
- $e \in \mathbb{N}$ est l'exposant.

Le nombre d'octets utilisés pour représenter m gouverne la précision numérique.

Le nombre d'octets utilisés pour représenter e détermine les points de dépassement et soupassement numérique.

Quatre constantes méchanique

Pour une machine donnée, les constantes suivantes décrivent les points de dépassement et soupassement, ainsi que la précision.

Constante	description
`double.xmax`	$x > 0$ le plus grand distinct de ∞.
`double.xmin`	$x > 0$ le plus petit distinct de 0.
`double.eps`	$x > 0$ le plus petit tel que $1 + x$ et 1 sont distincts.
`double.neg.eps`	$x > 0$ le plus petit tel que $1 - x$ et 1 sont distincts.

Trouver ces constantes avec R

```r
m = .Machine
m$double.eps
```

```
## [1] 2.220446e-16
```

```r
m$double.neg.eps
```

```
## [1] 1.110223e-16
```

```r
m$double.xmin
```

```
## [1] 2.225074e-308
```

```r
m$double.xmax
```

```
## [1] 1.797693e+308
```

Expansions de Taylor et de Mercator de la fonction $\ln x$

L'expansion de Taylor autour de $x = 1$:

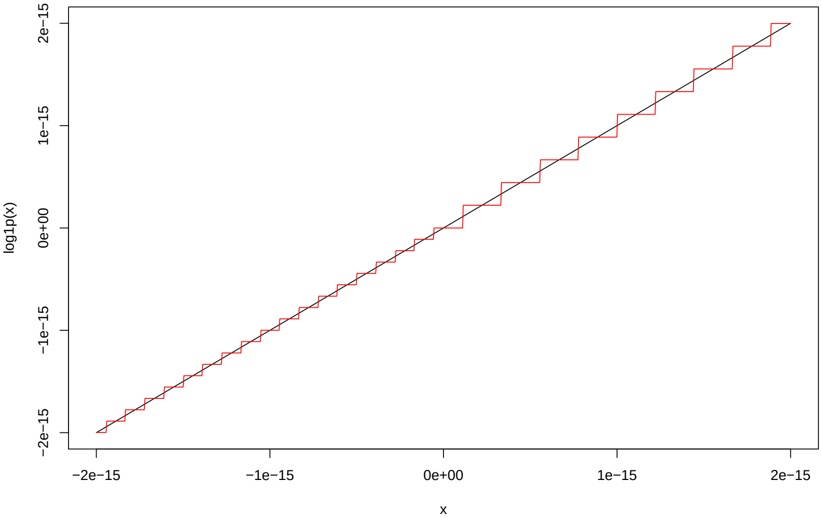
$$\ln x = \sum_{k=1}^{\infty} \frac{(-1)^k (x-1)^k}{k} = (x-1) - \frac{(x-1)^2}{2} + \frac{(x-1)^3}{3} - \ldots$$

L'expansion de Mercator :

$$\ln(1+x) = \sum_{k=1}^{\infty} \frac{(-1)^k x^k}{k} = x - \frac{x^2}{2} + \frac{x^3}{3} - \ldots$$

La fonction log1p

```
x = seq(-2e-15, 2e-15, length.out=1000)
plot(x, log1p(x), 'l')
lines(x, log(1+x), col='red')
```



Troncation des sommes infinies

Analyse (de la complexité) d'algorithmes

Notation $O(\cdot)$

- Pour des fonctions f et g sur $\mathbb{N}$, on écrit $f(n) = O(g(n))$ s'il existe $M > 0$ tel que $|f(n)| \leq Mg(n)$.
- Par exemple $f(n) = 6n^2 + 8n + 2 = O(n^2)$

La complexité de certains algorithmes

- $O(1)$: trouver l'élément $f(i)$ dans un tableau.
- $O(\log n)$: nombre de comparaisons pour trouver un élément dans une liste triée de n éléments, par recherche binaire.
- $O(n)$: nombre de comparaisons pour trouver un élément dans une liste non-triée de n éléments.
- $O(n^2)$: nombre de multiplications scalaires pour multiplier une matrice $n \times n$ et un vecteur $n \times 1$.
- $O(n^3)$: nombre de multiplications scalaires pour multiplier deux matrices $n \times n$ (Algorithme évident)
- $O(n^{2.8074})$: nombre de multiplications scalaires pour multiplier deux matrices $n \times n$ (Algorithme de Strassen)

L'importance d'algorithmes polynôme

- Un algorithme est polynôme si le temps de computation est $O(g(n))$ en n, le nombre de scalaires dans l'intrant, pour une polynôme $g(n)$.
- Il y a une distintinction importante entre les problèmes facile (pour lesquels il y a un algorithme polynôme connue pour le résoudre) et les problèmes plus difficile.
- Fonctions à sens unique.
- Si un algorithme est polynôme, il est habituellement facile à prouver qu'il l'est.
  - Les polynômes sont stable pour l'addition, la multiplication et la composition.
  - Si $p(x)$ est $q(x)$ sont des fonctions polynômes, $p(x) + q(x)$, $p(x)q(x)$ et $p(q(x))$ le sont aussi.
  - Les boucles, l'appel aux fonctions, ne sont pas

P, NP et NP Complet

- Une réduction : problème A n'est pas plus difficile que B ($A \leq_P B$) si A peut être réduit à B—résolu par un algorithme pour résoudre B plus une quantité polynôme de computation supplémentaire.
- Classes de problèmes :
  - P : problèmes de décision (oui/no) résoluble en temps polynôme.
  - NP : problèmes de décision (oui/non) ou une réponse affirmative peut être prouvé correcte en temps polynôme.
  - NP-complet : les problèmes en NP les plus difficile (une classe d'équivalence avec plusieurs exemples connus)
  - NP-difficile : les problèmes qui sont au moins aussi difficile que les problèmes dans NP-complet.
- P=NP? est une question non-résolue : il n'y a pas d'algorithmes polynômes connus pour résoudre les problèmes en NP.

Quelques problèmes

(polynomial, NP-complete, NP-hard)

Problème du voyageur de commerce (Travelling Salesman)

- Trouvez le trajet le plus court qui relie un ensemble de villes.
- Y a-t-il un trajet plus court que L qui relie les villes?

Optimisation linéaire en nombres entiers (Integer programming)

- Trouvez la solution optimale ou montrez qu'il n'y a pas de solution.
- (Cas spécial de variables binaires) Y a-t-il une solution faisable?

Optimisation linéaire (Linear programming)

- Trouvez la solution optimale ou montrez qu'il n'y a pas de solution.

L'évaluation des polynomes avec la méthode de Horner

Le problème est l'évaluation du polynôme $a_0 + a_1 x + \cdots + a_n x^n$.

Trois solutions

1. Évaluation naïve, $O(n^2)$ opérations, $O(1)$ registres :

$$a_0 + a_1 * x + a_2 * x * x + a_3 * x * x * x + \cdots$$

2. Meilleure, avec $O(n)$ opérations, $O(n)$ registres :
   a. Calculer $x^i = x^{i-1} * x$, $i = 2, \ldots, n$.
   b. Calculer $a_0 + a_1 * x + \cdots + a_n * x^n$.
3. La méthode de Horner, $O(n)$ opérations, $O(1)$ registres :

$$a_0 + x * (a_1 + x * (a_2 + x * (a_3 + \cdots + x * (a_{n-1} + x * a_n) \ldots)))$$

Parallelisme

Vous allez vous habituer à reconnaître deux types de problème où vous pouvez profitez des processeurs en parallèle.

Problèmes

L'embarras du parallelisme

Problèmes avec l'embarras du parallelisme

1. Évaluation d'une fonction sur une grille de points
2. Intégration numérique
3. Simulation Monte Carlo indépendent
4. Évaluation d'une fonction de vraisemblance
5. Multiplication des matrices

Problèmes sans l'embarras du parallelisme

1. Méthodes iteratives d'optimisation
2. Méthodes iteratives pour trouver un point fixe
3. Simulation Markov chain Monte Carlo

Problèmes SIMD (Single Instruction, Multiple Data)