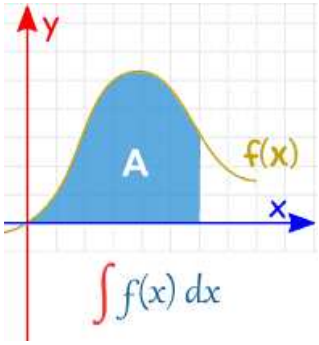
Calcul d'intégrales

Alessandro Torcini et Andreas Honecker

LPTM

Université de Cergy-Pontoise



Calculer la superficie d'une surface

Supposons que nous voulons calculer ¸ue la superficie A_F d'une surface $F \subset \mathbb{R}^2$. En employant son fonction caractéristique χ_F de F,

$$\chi_F(\vec{x}) = \begin{cases} 1 & \text{si } \vec{x} \in F \\ 0 & \text{si } \vec{x} \notin F \end{cases}$$

Par conséquent, il faut évaluer un intégrale bi-dimensionnel. En peut approximer cette quantité par une somme de Riemann

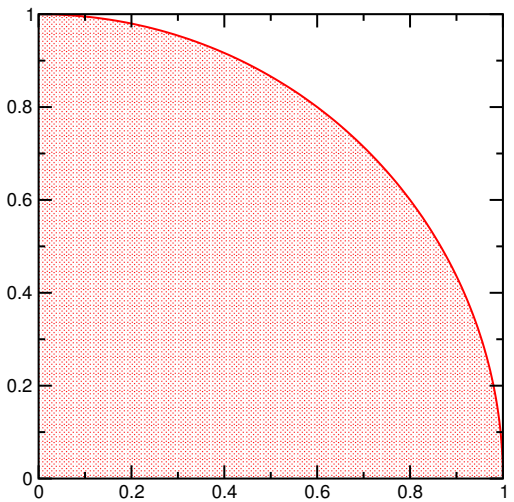
$$A_F = \int_{\mathbb{R}^2} \mathrm{d}^2 x \, \chi_F(\vec{x}) \approx \sum_i \chi_F(\vec{x}_i) \, A_i \, .$$

où $\vec{x}_i$ sont des points discrètes sur le plan, chacun entouré par une superficie A_i. Probablement vous pensez maintenant à un placement régulier des points $\vec{x}_i$, mais ceci n'est pas nécessaire. En effet, un peut prendre une distribution complètement aléatoire

La méthode de Monte-Carlo

Pratiquement, il faut inclure la surface F p.-ex. dans un rectangle R.
La superficie A_R du rectangle est connue et si nous prenons N points dans R, la superficie moyenne est $A_i = A_R/N$.
Par conséquent, pour $F \subset R$ nous trouvons l'approximation

$$A_F = \int_R \mathrm{d}^2x\, \chi_F(\vec{x}) \approx \frac{A_R}{N} \sum_{i=1}^{N} \chi_F(\vec{x}_i)\,.$$



Exercice (*Estimer π en lançant des fléchettes*) **:** Tirez $N = 100\ 000$ couples (x_i, y_i) avec $x_i, y_i, \in [0, 1)$. Comptez le nombre de cas quand $x_i^2 + y_i^2 < 1$ et estimez la probabilité de trouver (x_i, y_i) à l'intérieur de cet cercle, voir Figure. Estimez l'erreur de cette quantité et comparez-la avec $\pi/4$.
Utilisez `np.random.random_sample()` . Générateur aléatoire entre $[0 : 1)$ avec distribution uniforme

La méthode de Monte-Carlo

```
# pi et la méthode de Monte-Carlo
import numpy as np                  # importer le module comme "np"
N=1000                             # nombre de points
cnt = 0
var = 0

for i in range(0,N):
  x = np.random.random_sample() # tirer le couple (x,y)
  y = np.random.random_sample()
  if x*x + y*y < 1:                # le couple est-il dans le cercle ?
    cnt += 1                       # si oui : compter
    var += 1**2

moy = cnt/float(N)                 # la moyenne
var=  var/float(N) -moy*moy        #  la variance
ecart = np.sqrt(var/N)             #  ecart type de la moyenne

print "compte =", moy, "+/-", ecart
print "pi/4   =", np.pi/4
print "(erreur =", np.abs(moy-np.pi/4), ")"
```

L'intégrale d'une fonction

Évidemment, on peut facilement généraliser la méthode au calcul de l'intégrale d'une fonction f — il faut seulement remplacer χ_F avec f :

$$\int_R \mathrm{d}^2x\, f(\vec{x}) \approx \frac{A_R}{N} \sum_{i=1}^{N} f(\vec{x}_i)\,.$$

Plus généralement, si R est une région en d dimensions avec volume V_R, on a l'approximation

$$\int_R \mathrm{d}^d x\, f(\vec{x}) \approx \frac{V_R}{N} \sum_{i=1}^{N} f(\vec{x}_i)\,.$$

En une dimension ($d = 1$) l'intervalle $[a, b]$ est une région avec « volume » $V_{[a,b]} = b - a$. Par conséquent, en une dimension la prescription Monte Carlo devient

$$\int_a^b \mathrm{d}x\, f(x) \approx \frac{b-a}{N} \sum_{i=1}^{N} f(x_i)\,,$$

où x_i sont N points tirés avec distribution uniforme dans l'intervalle $[a, b]$.

L'intégrale en une dimension

```
import numpy as np                    # importer le module comme "np"
# si on passe la fonction f comme parametre, on peut definir une methode
# "universelle" pour le calcul de l'integrale Monte Carlo
#  N le nombre de points on peut preciser N comme intMC.N
# L'intervalle de confiance se trouve dans intMC.erreur

def intMC(f, a, b): # a est la borne inferieure et b le borne superieur
  somme, somme2 = 0, 0
  i = 0
  while i<intMC.N:
    xi = (b-a)*np.random.random_sample()+a  # tirer un point aleatorie dans
    fi = (b-a)*f(xi)                        # la fonction
    somme += fi
    somme2 += fi*fi
    i += 1
  somme /= float(intMC.N)                   # <x>  l'integral
  somme2 /= float(intMC.N)                  # <x^2>
  var = somme2-somme*somme                  # variance = <x^2>-<x>^2
  intMC.erreur = np.sqrt(var/float(intMC.N)) #ecart type sur la moyenne
  return somme
```

L'intégrale en une dimension

```
# la fonction :

def f1(x):                                          # x^2
  return x*x

# maintenant on peut utliser une boucle pour realisier les valeurs de N
for N in [1e2, 1e3, 1e4]:
  print "N =", N
  intMC.N = N

  print "I1 = ", intMC(f1, 0, 1), "+/-", intMC.erreur, "(exact : 1/3)"
```

Formules de quadrature

Une approche déterministe utilise une décomposition du domaine en morceaux. Pour l'illustrer, nous discuterons le cas d'une dimension.

Approximation simple

L'approximation la plus simple d'une intégrale d'une fonction f sur un intervalle $[a, b]$ est donné par la superficie d'une rectangle de $(b - a)$ et la valeur de la fonction à la borne inférieure :

$$\int_a^b \mathrm{d}x\, f(x) \approx (b-a)\, f(a) =: F_1\,.$$

Pour estimer l'erreur faite avec telle approximation, nous utilisons une exapnsion en série de Taylor :

$$f(x) = f(a) + (x-a)\, f'(a) + \ldots\,,$$

soit

$$\int_a^b \mathrm{d}x\, f(x) = \int_a^b \mathrm{d}x\, \left[f(a) + (x-a)\, f'(a)\right] + \ldots = F_1 + \frac{(b-a)^2}{2}\, f'(a) + \ldots\,.$$

L'erreur de l'approximation pour $F1$ est donc proportionnel à $(b-a)^2$.

Méthode du point médian

On arrive très facilement à une approximation meilleure ; il faut seulement prendre le milieu d'intervalle $[a, b]$ pour évaluer f et pas la borne inférieure :

$$\int_a^b \mathrm{d}x\, f(x) \approx (b-a)\, f\left(\frac{a+b}{2}\right) =: F_2\,.$$

Avec l'expansion de Taylor autour de $x_m = \frac{a+b}{2}$

$$f(x) = f\left(x_m\right) + \left(x - x_m\right)\, f'\left(x_m\right) + \frac{\left(x - x_m\right)^2}{2}\, f''\left(x_m\right) + \ldots$$

on trouve

$$\begin{aligned}
\int_a^b \mathrm{d}x\, f(x) &= \int_a^b \mathrm{d}x \left(f\left(x_m\right) + \left(x - x_m\right)\, f'\left(x_m\right) + \frac{\left(x - x_m\right)^2}{2}\, f''\left(x_m\right) \right) + \ldots \\
&= F_2 + \frac{(b-a)^3}{24}\, f''(x_m) + \ldots\,.
\end{aligned}$$

L'erreur de l'approximation F2 est donc proportionnel à $(b-a)^3$. Par conséquent, si $b-a$ est petit, l'approximation F2 est meilleure de l'approximation F1.

Méthode de Simpson

Si on utilise les bornes inférieures et supérieures et aussi le milieu de l'intervalle, on arrive à une formule toujours simple, mais encore meilleure :

$$\int_a^b \mathrm{d}x\, f(x) \approx \frac{b-a}{6}\left(f(a) + 4\, f\left(\frac{a+b}{2}\right) + f(b)\right) =: F_3\,.$$

On peut toujours estimer l'erreur de cette approximation en utilisant des séries de Taylor. Comme c'est un peu longue, je ne vous donne pas de détails, mais seulement le résultat final :

$$\int_a^b \mathrm{d}x\, f(x) = F_3 + \frac{(b-a)^5}{2880}\, f^{(4)}(\xi)$$

avec un $\xi \in [a, b]$.

Abscisses équidistantes

Les approximations sont bonnes quand $b - a$ est petit. Il faut donc subdiviser un intervalle $[a, b]$ donné en plusieurs intervalles petits. Si on prend N intervalles équidistantes, chacun a la longueur

$$\Delta x = \frac{b-a}{N}$$

et les bornes $a_i := a + i\,\Delta x\,, \qquad b_i := a + (i+1)\,\Delta x$ pour $i = 0, \ldots, N-1$.
Évidemment, $a_0 = a$, $b_{N-1} = b$ et $a_{i+1} = b_i$. Finalement, on peut utiliser les formules pour l'approximation F1, F2 ou F3, pour l'intervalle $[a_i, b_i]$.
Je précise la prescription pour la méthode du point médian :

$$\int_a^b \mathrm{d}x\, f(x) \approx \Delta x \sum_{i=0}^{N-1} f\left(a + \left(i + \frac{1}{2}\right)\,\Delta x\right) =: F_2(N)\,.$$

Pour chaque intervalle, l'erreur est proportionnel à Δx^3. Comme nous avons N intervalles, l'erreur total est proportionnel à $N\,\Delta x^3 = (b-a)^3/N^2 \propto 1/N^2$.

Erreurs d'intégration

1. Approximation simple

$$\left| \int_a^b \mathrm{d}x\, f(x) - F_1(N) \right| \propto \frac{1}{N}$$

2. Méthode du point médian

$$\left| \int_a^b \mathrm{d}x\, f(x) - F_2(N) \right| \propto \frac{1}{N^2}$$

3. Méthode de Simpson

$$\left| \int_a^b \mathrm{d}x\, f(x) - F_3(N) \right| \propto \frac{1}{N^4}$$

Plusieurs dimensions

Si on a un hypercube $Q = [a_1, b_1] \times \ldots \times [a_d, b_d]$ en d dimensions, on peut écrire l'intégrale d-dimensionnel comme plusieurs intégrales unidimensionnelles :

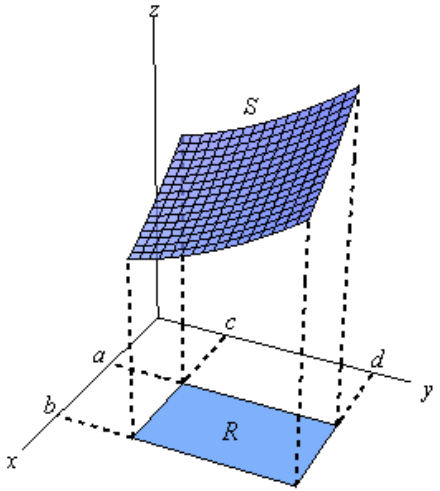
$$\int_Q \mathrm{d}^d x\, f(\vec{x}) = \int_{a_d}^{b_d} \mathrm{d}x_d \ldots \int_{a_1}^{b_1} \mathrm{d}x_1\, f(x_1, \ldots, x_d)\,.$$

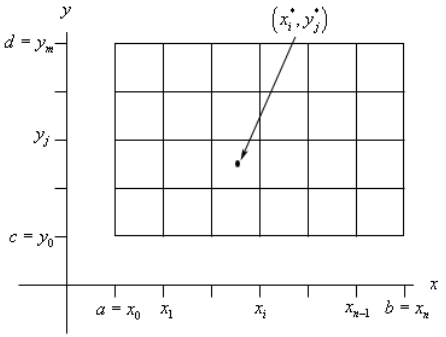
Après on peut utiliser une des quadratures que nous avons discutées. Il faut se seulement rendre compte que maintenant nous avons des points discrèts pour tous les coordonnées. Supposons que nous avons N points par coordonnée, nous avons $N_T = N^d$ points total, ou $N = \sqrt[d]{N_T} = (N_T)^{1/d}$.
L'erreur pour la méthode du point médian devient donc

$$\left| \int_Q \mathrm{d}^d x\, f(\vec{x}) - F_2(N) \right| \propto N_T^{-2/d}\,.$$

De la même façon, l'erreur est proportionnel à $N_T^{-1/d}$ pour l'approximation simple et proportionnel à $N_T^{-4/d}$ pour la méthode de Simpson.
Évidement, en plusieurs dimensions il faut (beaucoup) plus de points N_T pour un résultat précis.

Méthode du point médian pour 2 dimensions





Nous voulons intégrer la fonction bidimensionnelle $f(x, y)$ sur le rectangle $R = [a, b] \times [c, d]$ avec le méthode du point médian. Comme première étape nous allons diviser le intervalle $[a, b]$ in N intervalles équidistantes et $[c, d]$ in M intervalles équidistantes, chacun a la longueur

$$\Delta x = \frac{b-a}{N} \qquad \Delta y = \frac{d-c}{M}$$

et les bornes

$a_i := a + i\,\Delta x\,, \qquad b_i := a + (i+1)\,\Delta x \qquad c_j := c + j\,\Delta y\,, \qquad d_j := c + (j+1)\,\Delta x$

pour $i = 0, \ldots, N-1$ et $j = 0. \ldots . M-1$. Évidemment, $a_0 = a$, $b_{N-1} = b$ et $a_{i+1} = b_i$; $c_0 = c$ et $d_{M-1} = d$.

Méthode du point médian pour 2 dimensions

La prescription pour la méthode du point médian :

$$\int_a^b \mathrm{d}x \int_c^d \mathrm{d}y\, f(x,y) \approx \Delta x \times \Delta y \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} f(x_i^*, y_j^*) =: F_2^{2d}(N,M)\,.$$

ou $x_i^* = a + \left(i + \frac{1}{2}\right)\, \Delta x$ et $y_j^* = c + \left(j + \frac{1}{2}\right)\, \Delta y$

Pour la méthode de Simpson, l'expression devient extrêmement compliquée, voir
http ://mathfaculty.fullerton.edu/mathews/n2003/SimpsonsRule2DMod.html

Exercise : Évaluer l'intégrale suivante avec la méthode de Monte Carlo et la méthode du point médian :

$$I_{2d} = \int_3^5 dx \int_2^9 dy(x^2 + 10y)$$

Méthode du point médian pour 2 dimensions

$$I_{2d} = \int_3^5 dx \int_2^9 dy(x^2 + 10y) = \int_2^9 dy \left[\frac{x^3}{3} + 10xy\right]_{x=3}^{x=5} =$$

$$= \int_2^9 dy \frac{98}{3} + 20y = \left[\frac{98}{3}y + 10y^2\right]_{y=2}^{y=9} = \frac{686}{3} + 770 \simeq 998.666$$

Comparaison Monte Carlo et quadratures

Comme nous avons vu l'erreur de la méthode Monte Carlo est toujours proportionnel à l'inverse de la racine de nombre de points $1/\sqrt{N_T}$.
De l'autre côté, selon la discussion précédente, l'erreur des quadratures augmente avec la dimension d.
Si nous comparons l'erreur de la méthode du point médian avec la méthode Monte Carlo, nous trouvons que la méthode Monte Carlo devient plus efficace quand

$$\frac{1}{2} > \frac{2}{d},$$

soit quand

$$d > 4$$

De la même façon, la méthode Monte Carlo devient plus efficace de la méthode de Simpson quand

$$\frac{1}{2} > \frac{4}{d},$$

soit quand

$$d > 8$$