Informatique — M2 Physique Médicale

Introduction aux méthodes Monte-Carlo

Stéphane Perries

e-mail : perries@ipnl.in2p3.fr

12 novembre 2009

La simulation d'événements s'effectue à l'aide de nombres aléatoires.

1.1 Génération de nombres aléatoires

La génération de nombres aléatoires (ou plutôt de nombres pseudo-aléatoires [1]) est la clé de toutes les techniques de simulation. Un bon générateur doit pouvoir justifier d'un ensemble de qualités avant de pouvoir être utilisé intensivement [2]. Les tests statistiques standards de moyenne, écart-type, etc. sont évidemment nécessaires, mais ils sont notoirement insuffisants pour garantir des résultats fiables.

De très nombreux tests destinés à valider un générateur ont été proposés, tests dont la description est hors de propos ici. Il faut en particulier connaître *le cycle* du générateur ou du moins savoir si ce générateur possède un cycle qui est supérieur au nombre de tirages envisagés. Il existe *de bonnes sources*, citons en particulier : *Numerical Recipes*, les bibliothèque CERN, NAG, GNU Scientific Library, Root, etc.

En règle générale, les nombres aléatoires distribués *uniformément* sont générés à partir de congruences :

$$r_{j+1} = ar_j + c \quad (\text{modulo } m) \tag{1.1}$$

m est le *module*, et a et c sont le *multiplicateur* et *l'incrément* respectivement. La difficulté réside évidemment dans le choix de ces paramètres.

1.1.1 Utilisation des nombres aléatoire avec Root

Voir la classe **TRandom**.

1.1.2 Distribution uniforme

Dans root, la génération de nombres aléatoires selon une distribution uniforme peut s'effectuer de la manière suivante :

```
TRandom r;
r.Uniform();
```

- Ecrire une macro qui effectuer un tirage de 10 nombres aléatoires entre 0 et 100 en initialisant l'objet TRandom avec un entier de votre choix. Exécuter le programme plusieurs fois. Conclusion ?
- La commande time(NULL) renvoie le nombre de secondes écoulées depuis le 1er janvier 1970 (essayez dans root). Ce nombre peut servir de graine pour le générateur de nombre aléatoire. Remplacer `TRandom r;` par `TRandom r(time(NULL));` dans votre macro. Exécuter le programme plusieurs fois. Conclusion ?

1. En théorie, pour générer une vraie suite aléatoire il faudrait, par exemple, se servir des intervalles de temps entre les désintégrations d'un matériau radioactif.

2. Un générateur peut-être biaisé par bien des aspects et peut alors engendrer nombre de résultats curieux. Un générateur fiable en dimension 1 peut être catastrophique en dimension 2 !

1.2 Simulation et méthode de Monte Carlo

Les méthode de Monte-Carlo (notée MC) sont des techniques utilisant des nombres aléatoires pour résoudre un problème. La simulation de modèles stochastiques par MC est devenue d'une énorme importance ; elle facilite en particulier l'utilisation de théories incomplètement formalisées, et dont on ne saurait déduire par calcul les conséquences pratiques.

1.2.1 Détermination de la valeur de π

Ecrire une macro root permettant d'estimer la valeur de π à partir d'une méthode monte-carlo. Observer la dépendance de la précision en fonction du nombre de tirages aléatoires.

1.2.2 Simulation Monte-Carlo d'une désintégration radioactive

La désintégration d'une substance radioactive est un vrai processus aléatoire, la probabilité de désintégration étant une constante. La probabilité qu'un noyau se désintègre pendant l'intervalle de temps $\mathrm{d}t$ est : $p = \lambda\ \mathrm{d}t$ pour $\lambda\ \mathrm{d}t \ll 1$. La constante de désintégration λ est la probabilité de désintégration par unité de temps pour chaque noyau.

Le coeur de l'algorithme est le suivant :

```
Determine N0   (initial number of parents)
Determine lambda    (decay constant)
Determine Tmax (any time)
Determine dt   (time step)

N = N0

LOOP from t=dt to Tmax, step dt
   LOOP over each remaining parent nucleus
      Generate a random number R between [0,1] from a uniform distrib
      # Decide if the nucleus decays:
      # reduce the number of parents by 1
      IF( R < lambda * dt ) N = N-1
   END LOOP over nuclei
END LOOP over time
```

1. Ecrire un programme permettant de simuler la désintégration d'une source radioactive à l'aide de l'algorithme précédent. Stocker les valeurs de la population N à chaque instant t, dans un histogramme. Tracer cet histogramme
2. Modifier le programme précédent pour "fitter" la distribution obtenue par une fonction adéquate pour retrouver les paramètres de cette loi de désintégration. Etudier la dépendance en fonction du nombre de désintégration.
3. Modifier le programme précédent pour traiter du cas d'une chaine radioactive à 3 éléments, dans les deux premiers sont radioactifs de constante de désintégration λ_1 et λ_2.

1.2.3 Intégration par la méthode de Monte Carlo

L'évaluation des intégrales multiples est un domaine où la méthode de Monte Carlo (notée MC) rend des services inestimables. Cependant, nous nous limiterons ici au calcul d'une intégrale simple.

Formulation

Supposons que nous disposions de n points, distribués de manière aléatoire dans un volume V (supposé de dimension d quelconque). Le théorème de base de MC consiste à estimer que l'intégrale d'une fonction f sur le volume V est approchée par la relation,

$$\int f\, dV \approx V < f > \pm V\sqrt{\frac{< f^2 > - < f >^2}{n}}, \tag{1.2}$$

La première partie du résultat se déduit du théorème de la moyenne. Les quantités notées $< u >$ représentent la moyenne des quantités u sur les n points,

$$< f >\equiv \frac{1}{n}\sum_{i=1}^{n} f(x_i), \quad < f^2 >\equiv \frac{1}{n}\sum_{i=1}^{n} f^2(x_i). \tag{1.3}$$

Le terme d'erreur, proportionnel à la variance de f, est une *estimation*, à une erreur standard, et n'est absolument pas une borne rigoureuse. A partir de cette formulation, on réalise que la précision augmente seulement comme *la racine carrée* du nombre n de tirages. La convergence est donc, à priori, très mauvaise dans le cas à une dimension.
Considérons une méthode d'intégration où n est le nombre de points d'intégration et v la dimension de l'intégrale, le nombre d'évaluation de la fonction à intégrer est alors égal à n^v. Ce nombre peut devenir rapidement prohibitif [3]. Or dans MC la précision dépend du nombre de tirages pas de la dimension ! C'est pourquoi le domaine privilégié d'application de MC est celui des intégrales multiples.

Une intégrale simple

Comme exemple de l'évaluation d'un intégrale, par MC, nous prendrons l'intégrale très simple :

$$\int_0^1 \frac{1}{1+x^2}\, dx = \frac{\pi}{4} = 0.78540\ldots \tag{1.4}$$

– Ecrire le programme qui permet de calculer cette intégrale.
– Étudiez la convergence du calcul en fonction du nombre de tirages N (prendre $N = 2^i$, $i = 2, 3, \ldots$
– Vérifier que cette précision augmente comme $\sqrt{N}$.
– Comment peut-on imaginer d'améliorer la convergence de la méthode ?

3. Imaginer une méthode de Gauss à 20 points appliquée à une intégrale à 9 dimensions, il y alors $9^{20} = 1.2$ milliards évaluations de la fonction !