Université Paris Descartes
UFR de Mathématiques et Informatique
45, rue des Saints-Pères 75270 Paris cedex 06

PARIS DESCARTES

Introduction aux probabilités - Licence MIA

Contrôle continu du 17/12/2008- Durée : 1 heure

Les documents ne sont pas autorisés. **Tous les codes demandés devront être écrits très soigneusement afin que chaque caractère soit lisible.**

Exercice 1

1. Que renvoient, en R, les commandes `dgeom(n,p)` et `rgeom(n,p)` ?
2. Ecrire une fonction R équivalente à la fonction `dgeom`.
3. Expliquer pourquoi la fonction suivante est équivalente à la fonction `rgeom` :

```
SimulGeom = function(n,p)
{
  X = rep(0,n);
  for (i in 1:n)
    while (runif(1) >= p)
      X[i] = X[i] + 1;
  return(X);
}
```

4. Que se passe-t'il si on lance la commande `SimulGeom(1,0)` ?

Exercice 2 Le jeu de Pachinko consiste à laisser tomber une bille sur un plan incliné sur lequel sont fixés des clous. La bille rebondit sur les clous et suit un chemin aléatoire jusqu'à tomber dans l'un des trous situés sous les rangées de clous. On suppose qu'il y a cinq rangées de clous et six trous (voir dessin de gauche). A chaque étape, la bille a la même probabilité de tomber à droite ou à gauche du clou sur lequel elle rebondit, et tous les rebonds sont indépendants.

On peut représenter la trajectoire décrite par la bille par un vecteur $(X_1, X_2, X_3, X_4, X_5)$ où X_i vaut -1 si la bille va à gauche à l'étape i, ou 1 si elle va à droite. Par exemple sur le dessin de gauche, la trajectoire est $(-1, 1, -1, -1, 1)$.

1. Ecrire la formule donnant le numero N du trou dans lequel tombe la bille en fonction des $X_1, \ldots, X_5$.
2. Ecrire une fonction `Pachinko1` sans argument et en utilisant une boucle `for` qui simule la descente d'une bille et renvoie le numero du trou dans lequel elle est tombée.

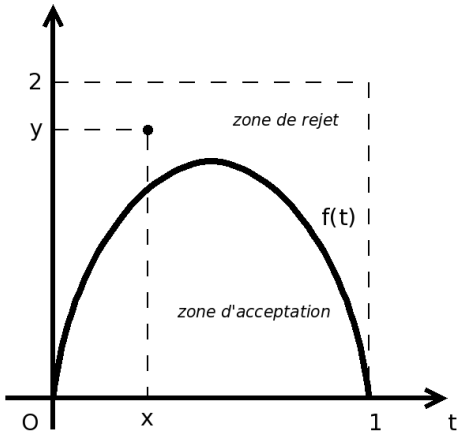
3. Ecrire une fonction `Pachinko2` qui réalise la même opération que la première mais ne comporte pas de boucle.
4. On modifie légèrement le jeu en disposant un septième trou à la place d'un des clous. Sa position est indiquée sur le dessin de droite. Modifier la fonction `Pachinko1` afin de simuler ce nouveau jeu.

Exercice 3 On considère une variable X à valeurs dans $[0,1]$, admettant une densité de probabilité $f(t) = 6t(1-t)$. Le but de cet exercice est d'écrire une fonction pour simuler cette variable X. La *méthode du rejet* est la suivante :

a) On tire au sort un nombre x simulé suivant une loi uniforme sur $[0,1]$, et un nombre y simulé suivant une loi uniforme sur $[0,2]$.

b) Si $y < f(x)$ on renvoie la valeur x (on dit que x est "accepté"), sinon on recommence à l'étape a) (x est "rejeté")

L'algorithme s'arrête donc lorsqu'une valeur x a été acceptée, et on peut montrer que cette valeur acceptée x est une simulation de la variable X.



algorithme du rejet : ici la valeur x est rejetée.

1. Ecrire une fonction `MethRejet` sans argument qui réalise cette méthode et renvoie la simulation x obtenue.

En fait la valeur 2, borne de l'intervalle pour le tirage de y, est arbitraire : on peut fixer cette valeur différemment sans changer la validité de la méthode. Le seul impératif est que cette valeur soit un majorant de la fonction f.

2. Pourquoi vaut-il mieux choisir une valeur de cette borne qui soit la plus petite possible ? Quelle est la valeur optimale ?