Modélisation & Simulation

Chapitre 1: Génération des nombres aléatoires

Mohamed Essaied Hamrita

Institut Supèrieur de Mathématiques Appliqués & Informatique - Kairouan

Octobre 2012

M2: Ingénierie Financière

Plan du chapitre

Définitions

Définitions

Définition

Une séquence pseudo-aléatoire (Pseudo Random Sequence en anglais) est une suite de nombres entiers $x_0, x_1, x_2, ...$ prenant ses valeurs dans l'ensemble $M = 0, 1, 2, ..., m-1$. Le terme x_n $(n > 0)$ est le résultat d'un calcul sur le ou les termes précédents. Le premier terme x_0 est appelé le germe (seed en anglais).

Définitions

Définition

Une séquence pseudo-aléatoire (Pseudo Random Sequence en anglais) est une suite de nombres entiers $x_0, x_1, x_2, ...$ prenant ses valeurs dans l'ensemble $M = 0, 1, 2, ..., m - 1$. Le terme x_n $(n > 0)$ est le résultat d'un calcul sur le ou les termes précédents. Le premier terme x_0 est appelé le germe (seed en anglais).

Afin de générer des séquences pseudo-aléatoires, on fait recours à des algorithmes qui se reposent à des fonctions déterministes.

Algorithme

Algorithme

Algorithme général de récurrence

1: Choisir x_0 dans M.
2: Poser $x_{n+1} = f(x_n)$, puis $Y_k = x_k/m$ pour $n = 0, 1, ..., m-1$.
Où f est une application de M dans M.

Algorithme

Algorithme général de récurrence

1: Choisir x_0 dans M.
2: Poser $x_{n+1} = f(x_n)$, puis $Y_k = x_k/m$ pour $n = 0, 1, ..., m-1$.
Où f est une application de M dans M.

Une suite construite comme indiqué dans l'algorithme ci-dessus contient toujours un cycle de nombres se répétant à partir d'un certain endroit. Pour éviter de devoir employer plusieurs fois le même nombre au cours d'une simulation, on cherchera à rendre la longueur de ce cycle, appelée **période**, aussi grande que possible.

Qualités d'un générateur

Tout générateur des nombres pseudo-aléatoires doit avoir certaines qualités, à savoir :

Qualités d'un générateur

Tout générateur des nombres pseudo-aléatoires doit avoir certaines qualités, à savoir :

- Uniformité et stochasticité : La distribution doit être uniforme et aléatoire.

Qualités d'un générateur

Tout générateur des nombres pseudo-aléatoires doit avoir certaines qualités, à savoir :

- Uniformité et stochasticité : La distribution doit être uniforme et aléatoire.
- Longueur de la période : Le générateur doit avoir une période longue. Si n est le nombre de bits dans un mot d'ordinateur, la longueur maximale de la séquence est de l'ordre de 2^{n-1}.

Qualités d'un générateur

Tout générateur des nombres pseudo-aléatoires doit avoir certaines qualités, à savoir :

- Uniformité et stochasticité : La distribution doit être uniforme et aléatoire.
- Longueur de la période : Le générateur doit avoir une période longue. Si n est le nombre de bits dans un mot d'ordinateur, la longueur maximale de la séquence est de l'ordre de 2^{n-1}.
- Reproductibilité : Il suffit de partir des mêmes valeurs initiales "seed" pour reproduire les mêmes séquences de nombres pseudo-aléatoires.

Suite

Suite

- Longueurs de sous séquences disjointes : Certains générateurs à plusieurs "seed" permettent de démarrer les travaux de simulation en parallèle. La séquence produite par une première "seed" est assez longue pour un travail, alors que la valeur donnée à une autre "seed" garantit la génération de séquences disjointes.

Suite

- Longueurs de sous séquences disjointes : Certains générateurs à plusieurs "seed" permettent de démarrer les travaux de simulation en parallèle. La séquence produite par une première "seed" est assez longue pour un travail, alors que la valeur donnée à une autre "seed" garantit la génération de séquences disjointes.
- Portabilité : La portabilité est essentielle pour le travail en collaboration.

Suite

- Longueurs de sous séquences disjointes : Certains générateurs à plusieurs "seed" permettent de démarrer les travaux de simulation en parallèle. La séquence produite par une première "seed" est assez longue pour un travail, alors que la valeur donnée à une autre "seed" garantit la génération de séquences disjointes.
- Portabilité : La portabilité est essentielle pour le travail en collaboration.
- Rapidité : Un générateur doit être très rapide.

Proposé par D.H Lehmer en 1948.

Proposé par D.H Lehmer en 1948.

Forme générale :

- Soient 3 constantes a, c et m. En choisissant aléatoirement une graine "seed", nous obtenons la récurrence :

$$x_n = ax_{n-1} + c \bmod m.$$

Proposé par D.H Lehmer en 1948.

Forme générale :

- Soient 3 constantes a, c et m. En choisissant aléatoirement une graine "seed", nous obtenons la récurrence :

$$x_n = ax_{n-1} + c \bmod m.$$

- Elle est périodique, de période maximale T_{max} telle que :

Proposé par D.H Lehmer en 1948.

Forme générale :

- Soient 3 constantes a, c et m. En choisissant aléatoirement une graine "seed", nous obtenons la récurrence :

$$x_n = ax_{n-1} + c \bmod m.$$

- Elle est périodique, de période maximale T_{max} telle que :

cas 1 $T_{max} = m - 1$ si $c = 0$.

Proposé par D.H Lehmer en 1948.

Forme générale :

- Soient 3 constantes a, c et m. En choisissant aléatoirement une graine "seed", nous obtenons la récurrence :

$$x_n = ax_{n-1} + c \bmod m.$$

- Elle est périodique, de période maximale T_{max} telle que :

cas 1 $T_{max} = m - 1$ si $c = 0$.
cas 2 $T_{max} = m$ si $c \neq 0$.

Proposé par D.H Lehmer en 1948.

Forme générale :

- Soient 3 constantes a, c et m. En choisissant aléatoirement une graine "seed", nous obtenons la récurrence :

$$x_n = ax_{n-1} + c \bmod m.$$

- Elle est périodique, de période maximale T_{max} telle que :

cas 1 $T_{max} = m - 1$ si $c = 0$.
cas 2 $T_{max} = m$ si $c \neq 0$.

- m est premier pour obtenir un bon comportement aléatoire des bits de poids faibles.

Le choix des paramètres a, c et m

Le choix des paramètres a, c et m

La qualité du générateur dépend du choix de a, c et m. On cherche à maximiser la période de la séquence qu'on va produire pour avoir un bon générateur.

Le choix des paramètres a, c et m

La qualité du générateur dépend du choix de a, c et m. On cherche à maximiser la période de la séquence qu'on va produire pour avoir un bon générateur.

Si $c \neq 0$ la période est **maximale** si :

Le choix des paramètres a, c et m

La qualité du générateur dépend du choix de a, c et m. On cherche à maximiser la période de la séquence qu'on va produire pour avoir un bon générateur.

Si $c \neq 0$ la période est **maximale** si :

- c est premier avec m i.e $\text{Pgcd}(c, m) = 1$.

Le choix des paramètres a, c et m

La qualité du générateur dépend du choix de a, c et m. On cherche à maximiser la période de la séquence qu'on va produire pour avoir un bon générateur.

Si $c \neq 0$ la période est **maximale** si :

- c est premier avec m i.e $\text{Pgcd}(c, m) = 1$.
- Tout diviseur **premier** de m divise $(a - 1)$.

Le choix des paramètres a, c et m

La qualité du générateur dépend du choix de a, c et m. On cherche à maximiser la période de la séquence qu'on va produire pour avoir un bon générateur.

Si $c \neq 0$ la période est **maximale** si :

- c est premier avec m i.e $\text{Pgcd}(c, m) = 1$.
- Tout diviseur **premier** de m divise $(a-1)$.
- $(a-1)$ est un multiple de 4, si m l'est.

Le choix des paramètres a, c et m

La qualité du générateur dépend du choix de a, c et m. On cherche à maximiser la période de la séquence qu'on va produire pour avoir un bon générateur.

Si $c \neq 0$ la période est **maximale** si :

- c est premier avec m i.e $\text{Pgcd}(c, m) = 1$.
- Tout diviseur **premier** de m divise $(a-1)$.
- $(a-1)$ est un multiple de 4, si m l'est.

Si $c = 0$ la période **maximale** si :

Le choix des paramètres a, c et m

La qualité du générateur dépend du choix de a, c et m. On cherche à maximiser la période de la séquence qu'on va produire pour avoir un bon générateur.

Si $c \neq 0$ la période est **maximale** si :

- c est premier avec m i.e $\text{Pgcd}(c, m) = 1$.
- Tout diviseur **premier** de m divise $(a-1)$.
- $(a-1)$ est un multiple de 4, si m l'est.

Si $c = 0$ la période **maximale** si :

- m est premier.

Le choix des paramètres a, c et m

La qualité du générateur dépend du choix de a, c et m. On cherche à maximiser la période de la séquence qu'on va produire pour avoir un bon générateur.

Si $c \neq 0$ la période est **maximale** si :

- c est premier avec m i.e $\text{Pgcd}(c, m) = 1$.
- Tout diviseur **premier** de m divise $(a-1)$.
- $(a-1)$ est un multiple de 4, si m l'est.

Si $c = 0$ la période **maximale** si :

- m est premier.
- $a^{m-1} - 1$ est multiple de m.

Le choix des paramètres a, c et m

La qualité du générateur dépend du choix de a, c et m. On cherche à maximiser la période de la séquence qu'on va produire pour avoir un bon générateur.

Si $c \neq 0$ la période est **maximale** si :

- c est premier avec m i.e $\text{Pgcd}(c, m) = 1$.
- Tout diviseur **premier** de m divise $(a-1)$.
- $(a-1)$ est un multiple de 4, si m l'est.

Si $c = 0$ la période **maximale** si :

- m est premier.
- $a^{m-1} - 1$ est multiple de m.
- $a^j - 1$ n'est pas divisible par m, pour $j = 1, 2, ..., m-2$.

Exemple

Exemple

Soit l'exemple simple suivant : si $a = c = x_0 = 3$ et $m = 5$, alors la séquence obtenue à partir de la formule de récurrence $x_n = (ax_{n-1} + c) \bmod m$ est :

$$3, 2, 4, 0, 3, 2, 4, 0, 3, ...$$

qui est de période égale à 4.

Générateur MRG

Générateur MRG

Expression de ce générateur :

Générateur MRG

Expression de ce générateur :

$$x_n = (a_1 x_{n-1} + a_2 x_{n-2} + ... + a_k x_{n-k}) \bmod m.$$

Générateur MRG

Expression de ce générateur :

$$x_n = (a_1 x_{n-1} + a_2 x_{n-2} + \ldots + a_k x_{n-k}) \bmod m.$$

Un tel générateur a une période de longueur $m^k - 1$.

Méthode LFG

Méthode LFG

C'est la suite bien connue :

$$x_n = (x_{n-j} + x_{n-l}) \bmod m, \quad 0 < j < l.$$

C'est un cas particulier du MRG.

Méthode LFG

C'est la suite bien connue :

$$x_n = (x_{n-j} + x_{n-l}) \bmod m, \quad 0 < j < l.$$

C'est un cas particulier du MRG.

Inconvénients :

- Ces générateurs (MRG et LFG) nécessite un emplacement mémoire type registre.
- Les nombres consécutifs obtenus sont fortement corrélés.

Générateur du carré median

Générateur du carré median

Ce générateur est proposé par John Von Neumann en 1946.

Principe

Générateur du carré median

Ce générateur est proposé par John Von Neumann en 1946.

Principe

On élève au carré un nombre de $2k$ chiffres et on prend comme terme suivant de la séquence pseudo-aléatoire le nombre formé des $2k$ chiffres du milieu du résultat (de $4k$ chiffres).

Exemple : $k = 1$ et $X_0 = 12$

n	0	1	2	3	4	5	6	7	8
x_n	12	14	19	36	29	84	05	02	00
x_{n+1}	0144	0196	0361	1296	0841	7056	0025	0004	0000

Générateur de Mersenne Twister

Générateur de Mersenne Twister

Développé par Makoto Matsumoto et Takuji Nishimura en 1997.

Générateur de Mersenne Twister

Développé par Makoto Matsumoto et Takuji Nishimura en 1997.

- Généralisation du MRG.

Générateur de Mersenne Twister

Développé par Makoto Matsumoto et Takuji Nishimura en 1997.

- Généralisation du MRG.
- Très performant mais architecture séquentielle.

Générateur de Mersenne Twister

Développé par Makoto Matsumoto et Takuji Nishimura en 1997.

- Généralisation du MRG.
- Très performant mais architecture séquentielle.
- Le générateur utilisé (par défaut) par le logiciel `R`.

Tests d'adéquation

Tests d'adéquation

On donne ici quelques tests statistiques afin de vérifier la stochasticité (Run test) et l'uniformité (test chi-deux et test de Kolmogorov-Smirnov) des séquences générées. trois tests seront exposés :

- Run test : test de stochasticité.

Tests d'adéquation

On donne ici quelques tests statistiques afin de vérifier la stochasticité (Run test) et l'uniformité (test chi-deux et test de Kolmogorov-Smirnov) des séquences générées. trois tests seront exposés :

- Run test : test de stochasticité.
- Test de chi-deux : test de conformité des lois.

Tests d'adéquation

On donne ici quelques tests statistiques afin de vérifier la stochasticité (Run test) et l'uniformité (test chi-deux et test de Kolmogorov-Smirnov) des séquences générées. trois tests seront exposés :

- Run test : test de stochasticité.
- Test de chi-deux : test de conformité des lois.
- Test de Kolmogorov-Smirnov : idem

Run test

Run test

C'est un test non paramétrique. Il s'agit de tester :

Run test

C'est un test non paramétrique. Il s'agit de tester :

$$\begin{cases} H_0 : \text{La séquence est } \textit{iid} \text{ (aléatoire)} \\ H_1 : \text{La séquence est non } \textit{iid} \text{ (non aléatoire)} \end{cases}$$

Run test

C'est un test non paramétrique. Il s'agit de tester :

$$\begin{cases} H_0 : \text{La séquence est } \textit{iid} \text{ (aléatoire)} \\ H_1 : \text{La séquence est non } \textit{iid} \text{ (non aléatoire)} \end{cases}$$

Principe du test

Ce test est basé sur le nombre de séquences croissantes et décroissantes. Soit R est le nombre de séquences dans l'échantillon de n nombres. Théoriquement, la moyenne et la variance sont :

$$E(R) = \frac{(2n-1)}{3} \quad \text{et } V(R) = \frac{(16n-29}{90}$$

Run test

Run test

Règle de décision

Sous H_0 et pour R grand, la statistique R est normale.

Run test

Règle de décision

Sous H_0 et pour R grand, la statistique R est normale.

Soit $Z = \dfrac{R - E(R)}{\sqrt{V(R)}}$. Au seuil $\alpha = 5\%$, si $|Z| > 1.96$, on rejette H_0.

Run test

Règle de décision

Sous H_0 et pour R grand, la statistique R est normale.

Soit $Z = \dfrac{R - E(R)}{\sqrt{V(R)}}$. Au seuil $\alpha = 5\%$, si $|Z| > 1.96$, on rejette H_0.

Exemple

Run test

Règle de décision

Sous H_0 et pour R grand, la statistique R est normale.

Soit $Z = \frac{R - E(R)}{\sqrt{V(R)}}$. Au seuil $\alpha = 5\%$, si $|Z| > 1.96$, on rejette H_0.

Exemple

Prenons l'échantillon suivant : 12 14 65 18 33 77 89 72 76 43 70 81 94 98 3.

Les séquences croissants décroissants sont comme suit :

Run test

Règle de décision

Sous H_0 et pour R grand, la statistique R est normale.

Soit $Z = \frac{R - E(R)}{\sqrt{V(R)}}$. Au seuil $\alpha = 5\%$, si $|Z| > 1.96$, on rejette H_0.

Exemple

Prenons l'échantillon suivant : 12 14 65 18 33 77 89 72 76 43 70 81 94 98 3.

Les séquences croissants décroissants sont comme suit :

$++ - + + + - + - + + ++ -$.

$R = 8$ séquences (4 croissantes (en rouge) et 4 décroissantes (en blue)). $E(R) = 9.66$, $V(R) = 2.34$ et $Z = -1.089$

$|Z| < 1.96$, on accepte H_O, c.à.d au seuil de 5% l'échantillon est **aléatoire**.

Run test avec R

Run test avec R

Dans R, il existe deux fonctions (à mes connaissances) pour faire ce test : `runs.test{tseries}` et `runs.test{lawstat}`.

Run test avec R

Dans R, il existe deux fonctions (à mes connaissances) pour faire ce test : `runs.test{tseries}` et `runs.test{lawstat}`.
En utilisant la deuxième fonction, on a :

Run test avec R

Dans R, il existe deux fonctions (à mes connaissances) pour faire ce test : `runs.test{tseries}` et `runs.test{lawstat}`.
En utilisant la deuxième fonction, on a :

```
library(lawstat) # charger l'extension lawstat.
x=c(12, 14, 65, 18, 33, 77, 89, 72, 76,
     43, 70, 81, 94, 98, 3)
rt=runs.test(x)
rt
```

```
 Runs Test - Two sided

data:  x Standardized Runs Statistic = -1.8667, p-value =0
```

Test de chi-deux

Test de chi-deux

Il s'agit de tester :

$$\begin{cases} H_0 : p_i^{\text{obs}} = p_i^{\text{theo}} \\ H_1 : \exists\ i \text{ tel que } p_i^{\text{obs}} \neq p_i^{\text{theo}} \end{cases}$$

Test de chi-deux

Il s'agit de tester :

$$\begin{cases} H_0 : p_i^{\text{obs}} = p_i^{\text{theo}} \\ H_1 : \exists\ i \text{ tel que } p_i^{\text{obs}} \neq p_i^{\text{theo}} \end{cases}$$

Le principe du test On regroupe la série simulée en k classes possibles avec des probabilités p_k, puis on calcule la statistique suivante :

Test de chi-deux

Il s'agit de tester :

$$\begin{cases} H_0 : p_i^{\text{obs}} = p_i^{\text{theo}} \\ H_1 : \exists\ i \text{ tel que } p_i^{\text{obs}} \neq p_i^{\text{theo}} \end{cases}$$

Le principe du test On regroupe la série simulée en k classes possibles avec des probabilités p_k, puis on calcule la statistique suivante :

La statistique utilisée

$$\chi_c^2 = \sum_{i=1}^{k} \frac{(f_i - np_i)^2}{np_i} \overset{H_0}{\sim} \chi^2(k-1)$$

où f_i est la fréquence observée de la classe c_i

Test de chi-deux

Test de chi-deux

Règle de décision

Test de chi-deux

Règle de décision Sous l'hypothèse nulle, la statistique χ_c^2 suit la loi $\chi^2(k-1)$. On accepte H_0 si $\chi_c^2 > \chi_\alpha^2(k-1)$ au seuil α.
En utilisant des logiciels statistiques, on accepte H_0 si `p-value` $> \alpha$.

Test de chi-deux

Règle de décision Sous l'hypothèse nulle, la statistique χ^2_c suit la loi $\chi^2(k-1)$. On accepte H_0 si $\chi^2_c > \chi^2_\alpha(k-1)$ au seuil α.
En utilisant des logiciels statistiques, on accepte H_0 si `p-value` $> \alpha$.

Avec `R`

Avec `R`, on peut utiliser la fonction `chisq.test` pour effectuer le test d'ajustement de chi-deux.

Test de chi-deux

Règle de décision Sous l'hypothèse nulle, la statistique χ_c^2 suit la loi $\chi^2(k-1)$. On accepte H_0 si $\chi_c^2 > \chi_\alpha^2(k-1)$ au seuil α.
En utilisant des logiciels statistiques, on accepte H_0 si `p-value` $> \alpha$.

Avec `R`

Avec `R`, on peut utiliser la fonction `chisq.test` pour effectuer le test d'ajustement de chi-deux.

Remarque : Ce test est utilisé pour les variables aléatoires discrètes.

Exemple

Exemple

Reprenant l'exemple précédent :

```
x=c(12, 14, 65, 18, 33, 77, 89, 72, 76,43,
   70, 81, 94, 98, 3)
p.theo=rep(1/length(x), length(x))
chisq.test(x,p=p.theo)
```

```
  Chi-squared test for given probabilities
data: x
X-squared = 265.3018, df = 14, p-value < 2.2e-16
```

Test de K-S

Test de K-S

Il s'agit de tester :

$$\begin{cases} H_0 : F_n(x) = F_0(x), \\ H_1 : F_n(x) \neq F_0(x) \end{cases}$$

Test de K-S

Il s'agit de tester :

$$\begin{cases} H_0 : F_n(x) = F_0(x), \\ H_1 : F_n(x) \neq F_0(x) \end{cases}$$

Statistique utilisée :

Test de K-S

Il s'agit de tester :

$$\begin{cases} H_0 : F_n(x) = F_0(x), \\ H_1 : F_n(x) \neq F_0(x) \end{cases}$$

Statistique utilisée : La statistique reliée au test de K-S, désignée par $\mathbf{D}$, se calcule comme suit :

Test de K-S

Il s'agit de tester :

$$\begin{cases} H_0 : F_n(x) = F_0(x), \\ H_1 : F_n(x) \neq F_0(x) \end{cases}$$

Statistique utilisée : La statistique reliée au test de K-S, désignée par $\mathbf{D}$, se calcule comme suit :

$$\mathbf{D} = \max_{\mathbb{R}} |F_n(x) - F_0(x)|$$

Test de K-S

Il s'agit de tester :

$$\begin{cases} H_0 : F_n(x) = F_0(x), \\ H_1 : F_n(x) \neq F_0(x) \end{cases}$$

Statistique utilisée : La statistique reliée au test de K-S, désignée par $\mathbf{D}$, se calcule comme suit :

$$\mathbf{D} = \max_{\mathbb{R}} |F_n(x) - F_0(x)|$$

où $F_n(x)$ est la fonction de répartition de la série simulée et $F_0(x)$ est la fonction de répartition de la loi uniforme.
Sous H_0, $\sqrt{n}\mathbf{D}$ suit la loi définit par :

$$K(x) = \sum_{n=-\infty}^{n=+\infty} \exp(-2(nx)^2)$$

Règle de décision

Sous H_0 pour $n \to +\infty$, $\mathbf{D} \to 0$. On accepte H_0 si $\mathbf{D} < c$ où c est déterminée par $P(\mathbf{D} > c|H_0) = \alpha$ avec α seuil de signification du test.

Règle de décision

Sous H_0 pour $n \to +\infty$, $\mathbf{D} \to 0$. On accepte H_0 si $\mathbf{D} < c$ où c est déterminée par $P(\mathbf{D} > c|H_0) = \alpha$ avec α seuil de signification du test.

Pour $n \leq 100$ et quelques valeurs de α les valeurs critiques c sont tabulées comme suit :

Règle de décision

Sous H_0 pour $n \to +\infty$, $\mathbf{D} \to 0$. On accepte H_0 si $\mathbf{D} < c$ où c est déterminée par $P(\mathbf{D} > c|H_0) = \alpha$ avec α seuil de signification du test.

Pour $n \leq 100$ et quelques valeurs de α les valeurs critiques c sont tabulées comme suit :

α \ n	5	10	15	20	25	30	40
$\alpha = 1\%$	0.6685	0.4864	0.4042	0.3524	0.3165	0.2898	0.2521
$\alpha = 5\%$	0.5633	0.4087	0.3375	0.2939	0.2639	0.2417	0.2101

α \ n	50	60	70	80	90	100	> 100
$\alpha = 1\%$	0.2260	0.2067	0.1917	0.1795	-	-	$1.63/\sqrt{n}$
$\alpha = 5\%$	0.1884	0.1723	0.1597	0.1496	0.1412	0.1340	$1.36/\sqrt{n}$

Exemple

Exemple

Avec `R`, on utilise la fonction `ks.test` pour effectuer le test K-S.

Exemple

Avec `R`, on utilise la fonction `ks.test` pour effectuer le test K-S.

Reprenons notre exemple :

Exemple

Avec R, on utilise la fonction `ks.test` pour effectuer le test K-S.
Reprenons notre exemple :

```
 ks.test(x,"punif")
data:  x
D = 1, p-value < 2.2e-16
alternative hypothesis: two-sided
```