IMIS : Master 1
Université Paris Est – Marne la Vallée

TP Bonus : Simulation de variables aléatoires

1. Simulation de lois

Dans les applications, on a souvent besoin de générer de façon artificielle (à l'aide d'un ordinateur) une suite $X_1, \cdots, X_n$ de nombres aléatoires i.i.d. suivant la loi donnée F. Les méthodes de simulation permettent seulement d'obtenir une valeur pseudo-aléatoire X_i, au lieu d'une valeur aléatoire. Cela signifie que les nombres $X_1, \cdots, X_n$ simulés sont déterministes (ils sont obtenus par un algorithme déterministe) mais les propriétés de la suite $X_1, \cdots, X_n$ sont proches d'une suite aléatoire iid de loi donnée. Par exemple pour les X_i pseudo-aléatoires on a la propriété de Glivenko-Cantelli :

$$\sup_x |F_n(x) - F(x)| \to 0 \text{ quand } n \to \infty,$$

mais il s'agit ici de la convergence au sens déterministe.

1.1. **Simulation des variables uniformément distribuées.** La f.d.r. $F^U(\cdot)$ de la loi uniforme $\mathcal{U}[0,1]$ s'écrit sous la forme

$$F^U(x) = x\mathbf{1}_{[0,1]}(x) + \mathbf{1}_{]1,+\infty]}(x)\,.$$

Le programme-générateur d'un échantillon pseudo-aléatoire $U_1, \cdots, U_n$ de cette loi est disponible dans de nombreux logiciels. Le principe de son fonctionnement est le suivant : on se donne un réel $a > 1$ et un entier m (a et m seront choisis très grands). On commence par une valeur Z_0 fixe. Pour tout $i \in \{1, \cdots, n\}$, on définit

$$\begin{aligned} z_i &= \text{ le reste de la division de } az_{i-1} \text{ par } m \\ &= az_{i-1} - \big[\frac{az_{i-1}}{m}\big]m\,. \end{aligned}$$

Pour tout $i \in \{1, \cdots, n\}$, nous avons toujours $0 \le z_i < m$. On définit

$$U_i = \frac{z_i}{m} = \frac{az_{i-1}}{m} - \big[\frac{az_{i-1}}{m}\big]\,.$$

Ainsi, pour tout $i \in \{1, \cdots, n\}$, $0 \le U_i < 1$. La suite $U_1, \cdots, U_n$ est considérée comme un échantillon de la loi uniforme $\mathcal{U}[0,1]$. Bien que cette suite n'est pas aléatoire, on peut montrer que sa f.d.r. empirique

$$F_n^U(x) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{U_i \le x}$$

est telle que $\sup_x |F_n^U(x) - F(x)| = \sup_{x\in[0,1]} |F_n^U(x) - x| \le \epsilon(n,m)$ avec $\epsilon(n,m)$ qui converge très vite vers 0 quand $m \to \infty$ et $n \to \infty$.

1. Comme la plupart des logiciels, SAS dispose d'un générateur de nombres pseudo-aléatoires. Afin de simuler un échantillon de $n = 50$ v.a. iid de loi uniforme $\mathcal{U}[0,1]$, exécuter le programme suivant :

```
DATA TableCree;
        DO i=1 TO 50;
          y=ranuni(10);
          OUTPUT;
        END;
        KEEP y;
RUN;
PROC PRINT; RUN;
```

   Le paramètre (10) est la graine ou semence du générateur. Réexécuter le programme avec la même valeur, une autre valeur, plusieurs fois avec la valeur -1. Une valeur négative comme -1 provoque l'utilisation de l'horloge interne comme semence "aléatoire". Cette dernière semence semble généré des nombres "plus aléatoires" mais avec un inconvénient : il est impossible de regénérer le même échantillon à un autre instant et donc de comparer des méthodes de façon rigoureuse.

1.2. **Simulation des variables de loi générale.** Etant donné un échantillon iid $U_1, \cdots, U_n$ d'une loi uniforme, on peut obtenir un échantillon d'une loi générale $F(\cdot)$ par la méthode d'inversion. Elle est opérationnelle si F^{-1} est disponible sous forme explicite. Cette méthode est basée sur la proposition suivante :

Proposition 1.1. *Soit F une f.d.r. continue et strictement croissante et soit U une v.a. uniformément distribuée sur $[0,1]$. Alors la v.a.*

$$X = F^{-1}(U)$$

suit la loi F.

Il en découle l'algorithme de simulation suivant : si F est une f.d.r. continue et strictement croissante, on pose

$$X_i = F^{-1}(U_i)\,,$$

où les U_i sont des nombres pseudo-aléatoires uniformément distribués sur $[0,1]$ générés comme expliqué précédemment. On obtient ainsi un échantillon simulé $(X_1, \cdots, X_n)$ de loi F.
Si F n'est pas continue ou strictement croissante, il faut modifier la définition de l'inverse de F et considérer l'inverse généralisée définie par : pour tout $u \in [0,1]$,

$$F^{-1}(u) = \inf\{x \in \mathbf{R} \text{ t.q. } F(x) \geq u\}\,,$$

avec la convention $\inf \emptyset = +\infty$. On a alors que $y \geq F^{-1}(u) \iff F(y) \geq u$. Cela implique que si X a pour f.d.r. F et si $U \sim \mathcal{U}[0,1]$, alors $F^{-1}(U)$ a la même loi que X.

Exemple 1. Simulation d'un échantillon de loi exponentielle $\mathcal{E}(1)$.
On a

$$f(x) = e^{-x}\mathbf{1}_{(x>0)} \text{ et } F(x) = (1-e^{-x})\mathbf{1}_{(x>0)}\,.$$

Ainsi $F^{-1}(y) = -\ln(1-y)$ pour $y \in (0,1)$. Posons alors $X_i = \ln(1-U_i)$ où les U_i sont des nombres pseudo-aléatoires uniformément distribués sur $[0,1]$.

Exemple 2. Simulation d'un échantillon de loi de Bernoulli. Soit

$$P(X=1) = p \text{ et } P(X=0) = 1-p\,,\ 0<p<1\,.$$

On a alors pour $y \in [0,1]$,

$$F^{-1}(y) = \inf\{x \in \mathbf{R} \text{ t.q. } F(x) \geq y\} = \begin{cases} 0 & \text{si } y \in [0, 1-p] \\ 1 & \text{si } y \in]1-p, 1]\,. \end{cases}$$

Si U_i est une v.a. de loi uniforme, alors $X_i = F^{-1}(U_i)$ suit la loi de Bernoulli. On pose alors

$$X_i = \begin{cases} 0 & \text{si } U_i \in [0, 1-p] \\ 1 & \text{si } U_i \in]1-p, 1]\,. \end{cases}$$

2. A l'aide de la méthode d'inversion, simuler un échantillon de taille $n = 100$ de v.a. iid de loi exponentielle de paramètre 1. Refaire la même démarche en utilisant l'instruction `ranexp(10)`. Les instructions `rannor(10)` et `rancau(10)` permettent respectivement de simuler des lois normales et des lois de Cauchy. A l'aide de l'instruction `rannor(10)`, simuler un $n = 50$ échantillon de loi $\mathcal{N}(1,4)$.

1.3. **Simulation des variables de loi gaussienne.** On présente ci-dessous deux méthodes connues pour générer des variables (pseudo) aléatoires suivant une loi gaussienne.

1.3.1. *Via le théorème central limite.* Pour $U \sim \mathcal{U}[0,1]$, on a $E(U) = 1/2$ et $\mathrm{Var}(U) = 1/12$. Vu le TCL, si les U_i sont iid et de loi $\mathcal{U}[0,1]$, on a que

$$\frac{U_1 + \cdots + U_N - N/2}{\sqrt{N/12}} \to_{N\to\infty} \mathcal{N}(0,1) \text{ en loi.}$$

La valeur $N = 12$ est déjà suffisante pour obtenir une bonne approximation de la loi normale. On en déduit la méthode de simulation suivante : on génère $U_{11}, \cdots, U_{nN}$, une suite de variables pseudo-aléatoires de loi $\mathcal{U}[0,1]$ et on pose ensuite

$$X_i = \frac{U_{i1} + \cdots + U_{iN} - N/2}{\sqrt{N/12}} \to_{N\to\infty} \mathcal{N}(0,1)\,, i = 1, \cdots, n.$$

On obtient ainsi un échantillon simulé $(X_1, \cdots, X_n)$ de la loi approximativement $\mathcal{N}(0,1)$.

3. Exécuter le programme suivant

```
DATA TableCree;
        ARRAY sim{12} y1-y12;
        DO i=1 TO 1000;
          DO j=1 TO 12;
            sim{j}=ranuni(-1);
          END;
          S=sum(of y1-y12);
          OUTPUT;
        END;
        KEEP S;
RUN;
PROC PRINT; RUN;
```

Lancer SAS/INSIGHT :
`solutions/Analyse/Analyse interactive des données`.
Charger la table créée ci-dessus :
`work/TableCree/OK`
Pour obtenir de belles sorties graphiques et des estimations de la densité :
`Analyze/Distribution /S/Y/OK`
`Curves/ kernel density/OK`

1.3.2. *Via la méthode de Box et Müller.* Nous renvoyons au TP5 pour un descriptif de cette méthode.

2. Simulations d'intervalles de confiance

On considère la loi normale de moyenne 5 et d'écart type 2 et on prendra pour seuil $\alpha = 0,05$. En supposant $\sigma = 2$ connu, nous tirons un échantillon de taille 25 par exemple. Celui-ci nous permet d'avoir une estimation par intervalle de μ (que l'on sait valoir 5). Cet intervalle peut contenir ou non le paramètre à estimer : tout dépend de l'échantillon obtenu. Nous allons considérer 100 échantillons et examiner les intervalles de confiance.

4. Exécuter les programmes suivants :

```
DATA simul;
        n=25; nechantillon=100; mu=5; sigma=2;
        DO j=1 TO nechantillon;
          DO i=1 TO n;
            x=sigma*rannor(0)+mu;
            OUTPUT;
          END;
        END;
        DROP i nechantillon;
RUN;

PROC MEANS Noprint;
        VAR x;
        OUTPUT Out=moyennes mean=m;
        BY j;
RUN;

DATA intervalles;
        SET moyennes;
        sigma=2;n=25; mu=5;
        a=m-sigma*1.96/sqrt(n);
        b=m+sigma*1.96/sqrt(n);
RUN;

SYMBOL1 i=join c=red width=1;
SYMBOL2 i=join c=blue width=1;
SYMBOL3 i=join c=green width=1;
TITLE 'Bornes de confiance de mu=5 au niveau alpha=0.05';

PROC GPLOT Data=intervalles;
        PLOT a*j=1 mu*j=2 b*j=3 / Overlay;
RUN;
TITLE;
QUIT;
```