L2 - Langage de programmation - Aller à l'alea

Jean-Paul CHEHAB

LAMFA UMR CNRS 7352, Univ. Picardie Jules Verne

30 avril 2020

Plan

Plan

Plan

Plan

Plan

La génération de nombres aléatoires s'effectue à l'aide d'algorithmes déterministes, par exemple basés sur des critères arithmétiques, mais le résultat obtenu vérifie des caractéristiques de l'aléa, comme la non-reproductibilité d'une simulation à l'autre.

Premières commandes

- `rand()` fournit un résultat choisi au hasard dans l'intervalle suivant la loi uniforme dans $[0, 1]$ mais d'autres loi sont disponibles. .Par exemple `d=rand(1,10000,'normal')` construira une suite de 1000 nombre suivant la loi normale.
- `rand(m,n)` fournit une matrice de $m \times n$ dont chaque élément est tiré suivant la loi uniforme dans $[0, 1]$.
- Si X est une matrice, `rand(X)` fournit une matrice de mêmes dimensions
- `rand("seed",getdate("s"))` permet d'initialiser le générateur de nombres aléatoires.

Illustration

```
rand()  :
ans =
0.2113249

rand()
ans =
0.7263507
```

Cela change à chaque fois ! 5 d'un coup maintenant

```
rand(1,5) ans =
0.6325745 0.4051954 0.9184708 0.0437334 0.4818509
```

ou bien

```
rand(7,5)
ans =

           0.3435337 0.5376230 0.2017173 0.1205996 0.9923191
           0.3760119 0.1199926 0.3911574 0.2855364 0.0500420
           0.7340941 0.2256303 0.8300317 0.8607515 0.7485507
           0.2615761 0.6274093 0.5878720 0.8494102 0.4104059
           0.4993494 0.7608433 0.4829179 0.5257061 0.6084526
           0.2638578 0.0485566 0.2232865 0.9931210 0.8544211
           0.5253563 0.6723950 0.8400886 0.6488563 0.0642647
```

Marche aléatoire 1D

Principe

Avant d'effectuer chaque pas, un individu tire à pile ou face : pile il avance d'un pas (+1), face, il recule d'un pas (-1). Sa position au k-ième pas est la somme algébrique des pas (+1) et (-1) effectués. Sa marche est l'historique de ses positions.

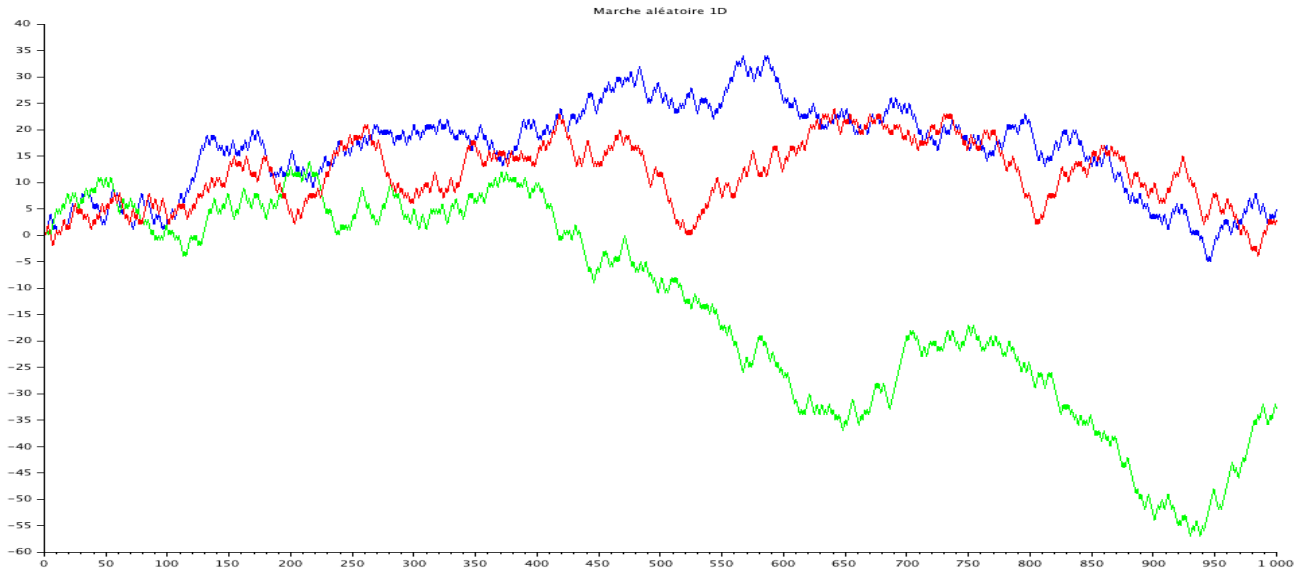


Figure – 3 réalisations de marches aléatoires 1D

Marche aléatoire 2D

Principe

Le principe est similaire qu'un 1D mais ici 4 possibilités s'offrent : un pas vers le Nord, le Sud, l'Est ou l'Ouest.

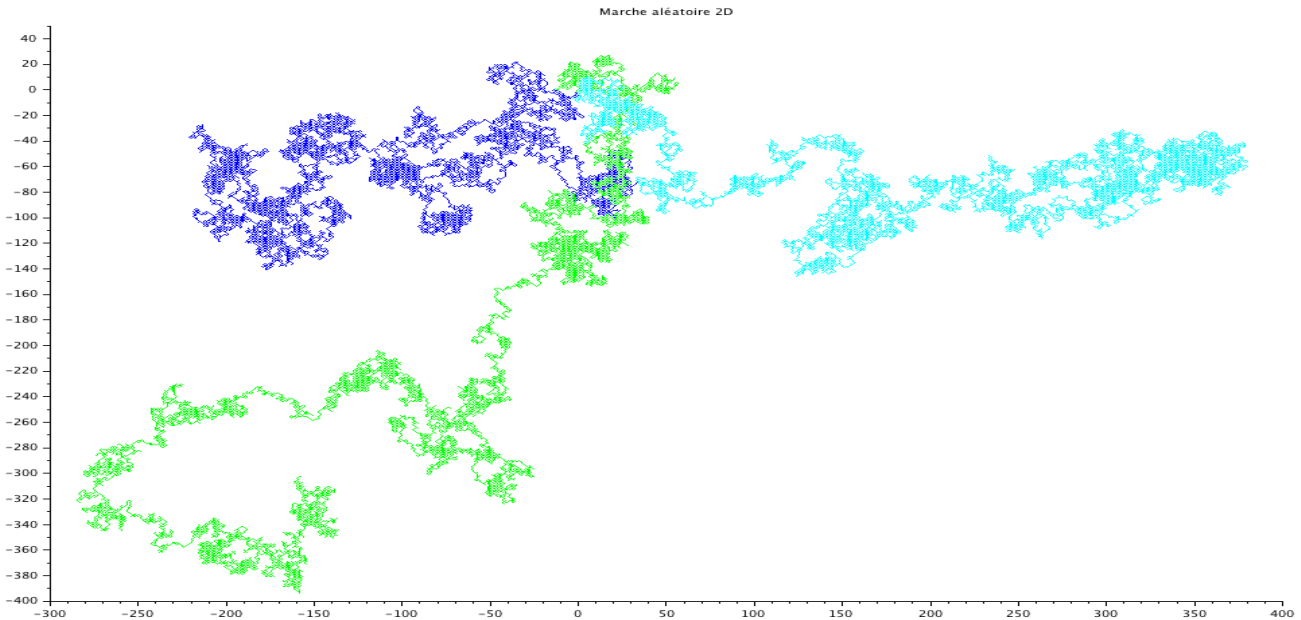


Figure – 3 réalisations de marches aléatoires 2D

Marche aléatoire 3D

Principe

Le principe est similaire qu'un 2D mais ici 6 possibilités s'offrent : un pas vers le Nord, le Sud, l'Est ou l'Ouest, en haut, en bas.

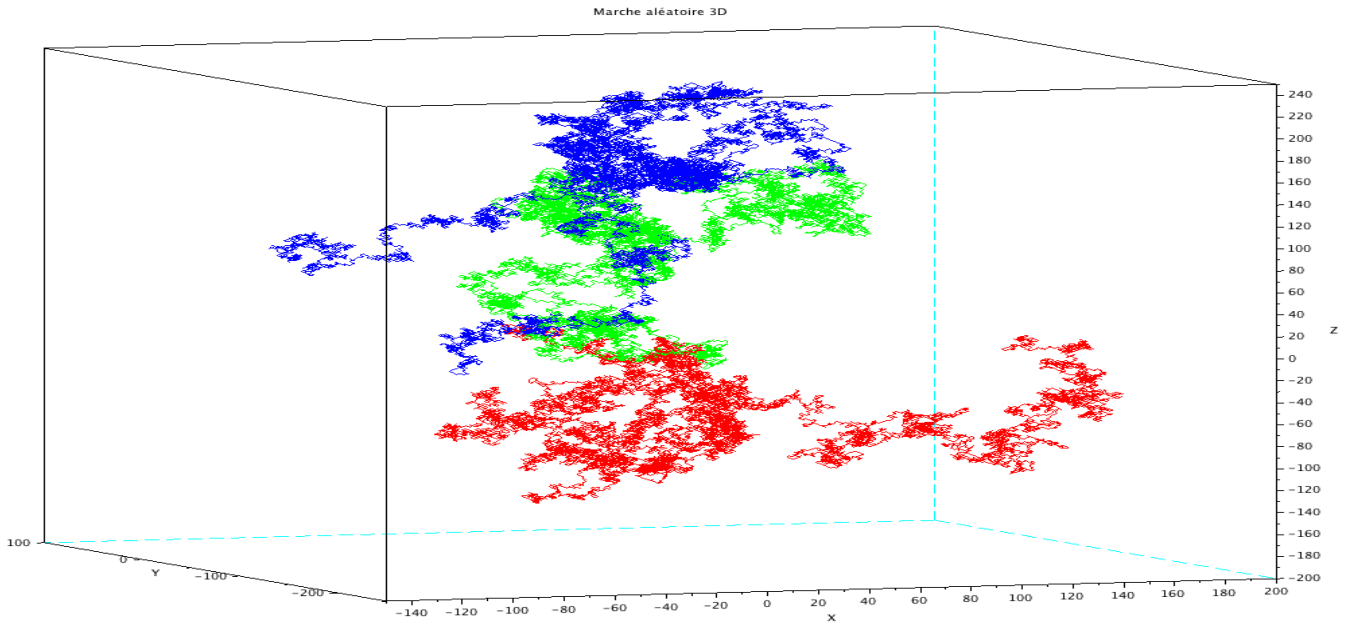


Figure – 3 réalisations de marches aléatoires 3D

La loi binomiale : on effectue n tirages avec remise dans une famille de N individus, de deux type, avec $N1$ individus du type 1 et $N2$ de type 2, $N = N1 + N2$. On note X la variable aléatoire correspondant au nombre d'individus de type 1 obtenus.

Theorem

(Simulation de variables aléatoires suivant des lois discrètes usuelles)

- *Loi uniforme* $U([a; b])$ *:* `X=a+floor((b-a+1)*rand())`
- *Loi binomiale* $B(n; p)$ *:*

```
X=0 ;
for k=1:n
if rand()<p
X=X+1 ;
end
end
```

- *Loi géométrique* $G(p)$ *:*

```
X=1 ;
while rand()>p
X=X+1 ;
end
```

Illustration

Uniforme sur $[-1, 1]$ `1-2*rand(1,5)`
On peut également définir sa propre fonction

```
function x=drand(m,n,p)
//matrice mxn de nombres pseudo-aléatoires de loi p sur
[1,...,length(p)]
cp=cumsum(p);
y=rand(m,n);// ou : y=grand(m,n,'def')
x=ones(y);
for k=1:length(p)-1,
ind=find(y>cp(k));
x(ind)=x(ind)+1;
end;
endfunction;
```

$\Rightarrow$ Voir la commande `grand` qui permet de générer des nombres aléatoires suivant une loi donnée

grand

Avec `rand`

- `rand(n,m)` est une matrice *nxm* à entrées indépendantes de loi uniforme sur $[0,1]$ (par défaut)
- On peut présélectionner la loi en avec `rand("normal")` alors `rand(n,m)` retournera une matrice *nxm* à entrées indépendantes de loi $\mathcal{N}(0,1)$. D'autres possibilités sont : `rand("normal"), rand("exp") ...`

Avec `grand(n,m, arguments supplémentaires)`

On construit `X` matrice *nxm* à entrées distribuées selon une loi spécifiée par les arguments supplémentaires.

- `X = grand(n,m,"bin",N,p)` : v.a. de loi binomiale de paramètres N,p
- `X = grand(n,m,"poi",mu)` : v.a. de loi de Poisson de moyenne mu
- `X = grand(n,m,"exp",mu)` : v.a. exponentielles de moyenne mu
- `X = grand(n,m,"nor",mu,s)` : v.a. gaussiennes de moyenne mu et d'écart-type s

Exemples

- Loi de Bernouilli : $P(X=-1)=p,\ P(X=1)=1-p$ (tirage de pile ou face) : `B=(rand(1,n)<p)`
- Loi uniforme discrète : $P(x=k)=\frac{1}{n},\ k=1,\cdots,n$: `U=rand()`
- Loi Binomiale (on tire n fois à pile ou face et on regarde le nombre de pile) $P(x=k)=C_n^k p^k(1-p)^{n-k}\ k=0,\cdots,n$
- Loi de Poisson $P(x=k)=e^{-\lambda}\frac{\lambda^k}{k!}, k=1,\cdots,+\infty,\ \lambda>0$

Espérance, variance

- Espérance $E(x)=\sum_{k=1}^{n} kP(x=k)$ ou $E(x)=\sum_{k=1}^{+\infty} kP(x=k)$
- Variance $Var(x)=E((x-E(x))^2)=E(x^2)-(E(x))^2=$ $\sum_{k=1}^{n} k^2P(x=k)-(sum_{k=1}^{n}kP(x=k))$

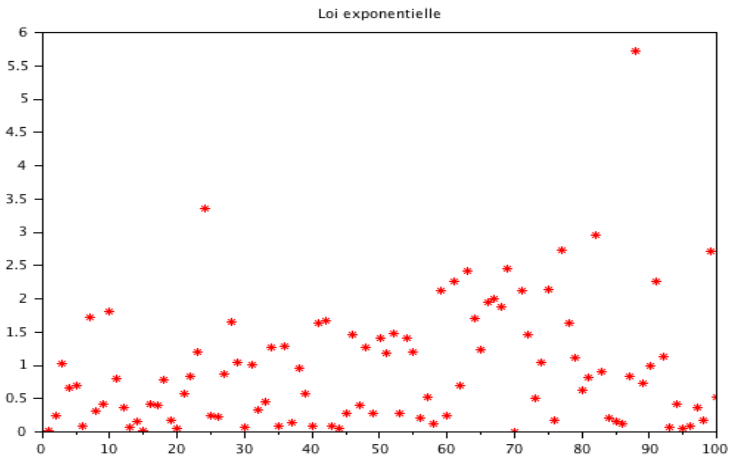
Densité, fonction de répartition

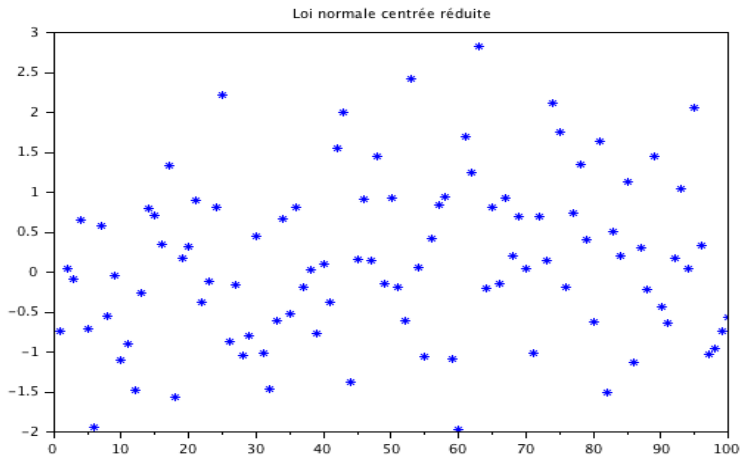
- Densité de loi de probabilité P sur $[a, b]$: $f(x) \geq 0$ telle que $\int_a^b f(x)dx = 1$
- Fonction de répartition : $F(y) = \int_a^y f(x)dx$, $F(y) = P(x < y)$.

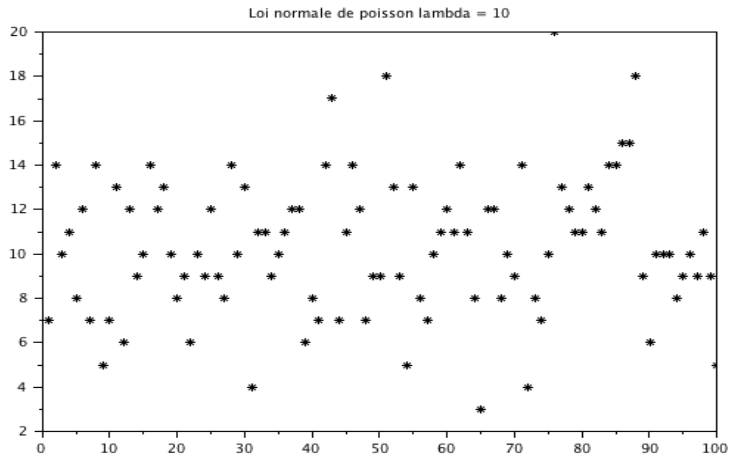
Exemples

- Loi uniforme sur $[0, 1]$: $f(x) = \begin{cases} 1 & \text{si } x \in [0, 1], \\ 0 & \text{sinon} \end{cases}$
- Loi normale sur $\mathbb{R}$: $f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma}}$
- Loi exponentielle sur $\mathbb{R}^+$: $f(x) = \lambda e^{-\lambda x}$, $\lambda > 0$

Comment représenter des suites de nombres aléatoires ?







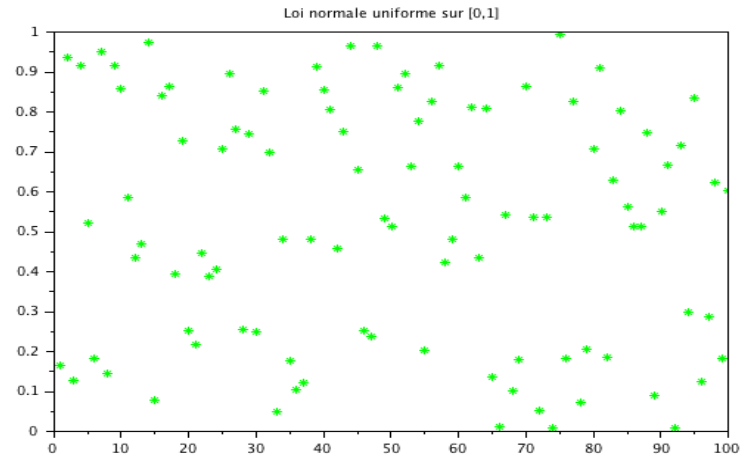


Figure – Représentation brute de nbs aléatoires

Histogramme

Il s'agit de représenter graphiquement la répartition d'une variable continue (ou discrète) avec des colonnes verticales. En rangeant les fréquences d'apparition.

hisplot(NbBar,DimEchantillon,Loi)

```
r=rand(1000,1,"normal");
scf();
histplot(10,r);
xtitle("Distribution normale","X","Fréquence")
```

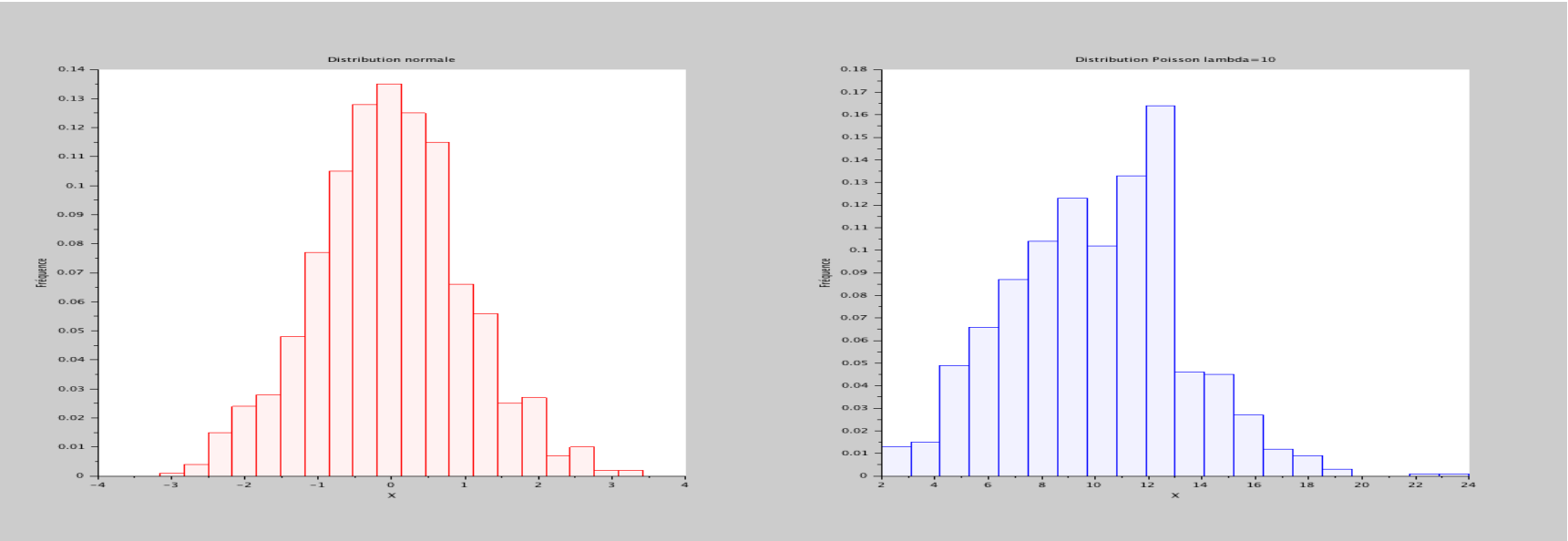


Figure – Histogrammes loi normale et Poisson ($\lambda = 10$)

Illustration

Une loi discrète sur une partie finie de $\mathbb{R}$ peut se représenter par un diagramme en bâtons ou par un diagramme en barres. On dispose pour cela des commande `plot2d3` et `bar` :

```
x=binomial(0.6,20);
subplot(121);
plot2d3(0:20,x);
subplot(122);
bar(0:20,x)
```

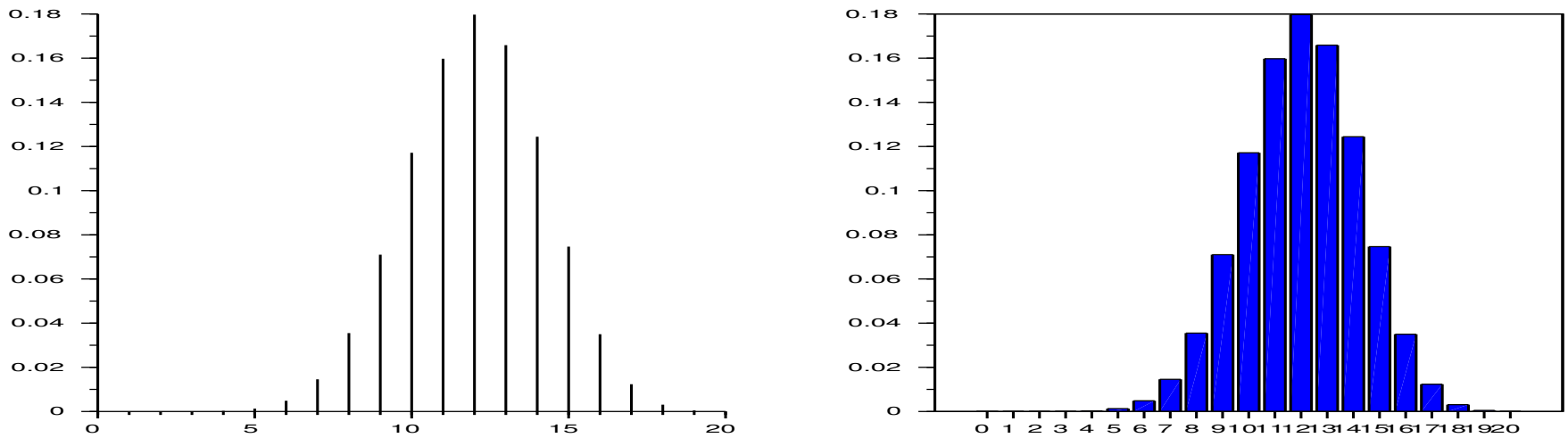


Figure – Simulation de la loi binomiale - histogramme

Illustration 2

La représentation graphique d'une loi de densité sur $\mathbb{R}$, s'effectue à l'aide de la commande `plot2d`, comme par exemple ci-dessous avec une gaussienne :

```
x1=linspace(0,20);m=12;v=4.8;
plot2d(x1,exp(-(x1-m).^2/2/v)/sqrt(2*%pi*v),style=2);
```

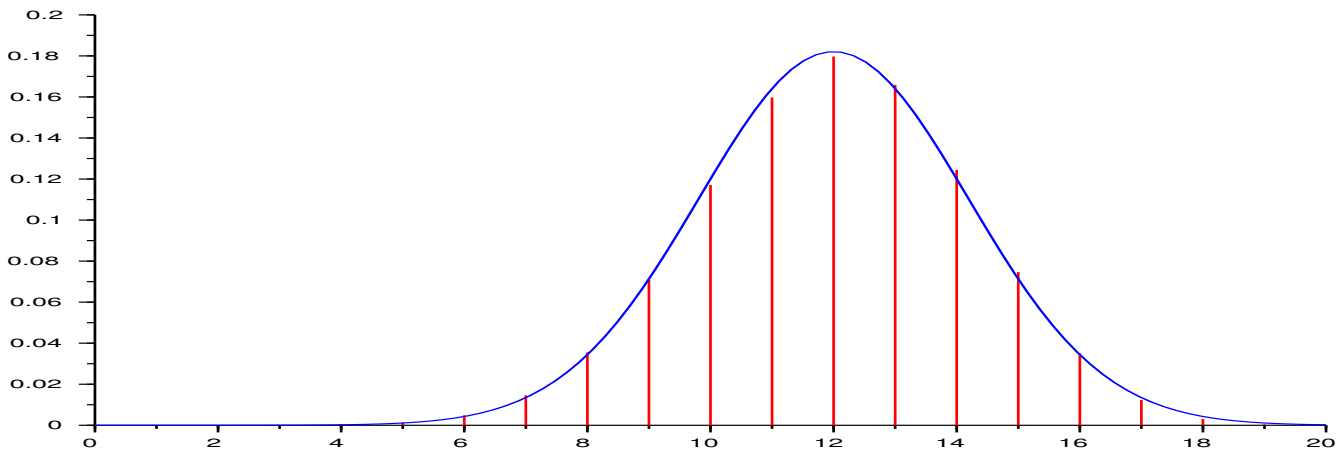


Figure – Simulation de la loi binomiale - densité

- Moyenne empirique

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

A l'aide de `mean` : `mean(x)` ou à l'aide de `sum` : `sum(x)/length(x)`.

- Variance

$$Var(x) = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

la commande `variance(x)` effectue ce calcul. ATTENTION il s'agit ici de l'estimateur sans biais d'où le $n-1$ (sinon on divise par n)

La fonction de répartition empirique d'un échantillon $x = (x_1, ..., x_n)$ est la fonction de répartition de la loi empirique

$$F_n(y) = \frac{1}{n} \sum_{i=1}^{n} \chi_{y \geq x_i}$$

avec

$$\chi_{y \geq x} = \begin{cases} 1 & \text{si } y \geq x \\ 0 & \text{sinon} \end{cases}$$

Exemple avec une loi normale réduite `n=100;`

```
x=rand(1,n,'n');
xx=gsort(x,'g','i');
plot2d2(xx,(1:n)/n);
y=cdfnor('PQ',xx,zeros(xx),ones(xx));
plot2d(xx,y,style=3)
```

$\Rightarrow$ Voir les commandes `cdf...` *cummulative distribution* qui permettent de construire les fonction de répartition de certaines lois ; `gsort` trie par ordre croissant les éléments de `x`

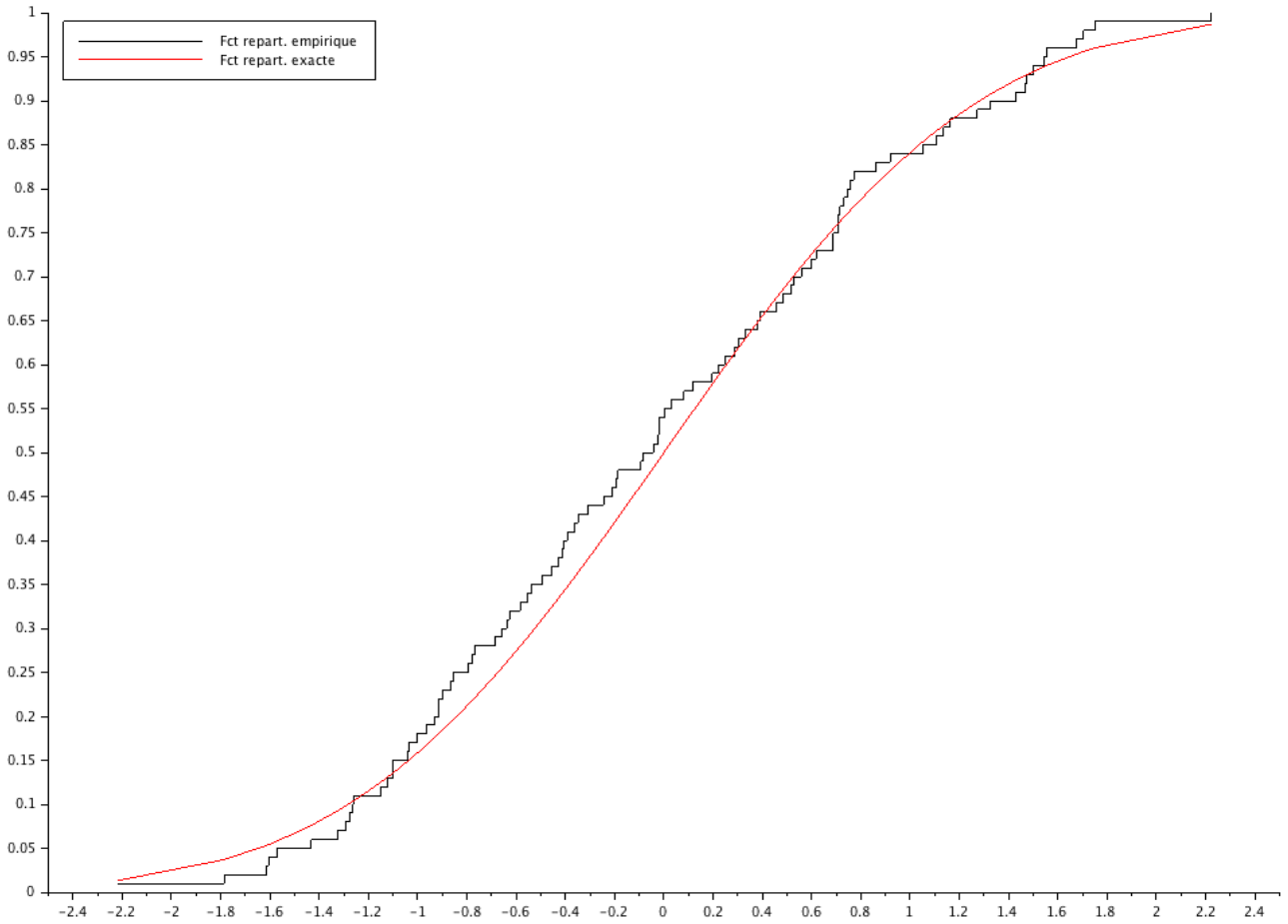

Fct repart. empirique
Fct repart. exacte
1
0.95
0.9
0.85
0.8
0.75
0.7
0.65
0.6
0.55
0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
-2.4
-2.2
-2
-1.8
-1.6
-1.4
-1.2
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
2.2
2.4

Figure –

Régression

On se donne un nuage de points (x_i, y_i), on cherche à déterminer la meilleur droite $y = ax + b$ qui "résume" ce nuage, i.e. a et b tels que

$$\phi(a,b) = \sum_{i=1}^{n} (y_i - (ax_i + b))^2$$

soit minimale. On montre que ceci équivaut aux relations

$$\begin{cases} a\sum_{i=1}^{n} x_i^2 + b\sum_{i=1}^{n} x_i &= \sum_{i=1}^{n} x_i y_i \\ a\sum_{i=1}^{n} x_i + nb &= \sum_{i=1}^{n} y_i \end{cases}$$

ce système 2×2 se résout facilement, on trouve

$$\begin{cases} a = \dfrac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{\delta} \\ b = \dfrac{-n\sum_{i=1}^{n} x_i y_i + \sum_{i=1}^{n} y_i \sum_{i=1}^{n} x_i^2}{\delta} \end{cases} \quad \text{avec } \delta = n\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2.$$

Régression

On se donne un nuage de points (x_i, y_i), on cherche à déterminer la meilleur droite $y = ax + b$ qui "résume" ce nuage, i.e. a et b tels que

$$\phi(a,b) = \sum_{i=1}^{n} (y_i - (ax_i + b))^2$$

soit minimale. On montre que ceci équivaut aux relations

$$\begin{cases} a\sum_{i=1}^{n} x_i^2 + b\sum_{i=1}^{n} x_i &= \sum_{i=1}^{n} x_i y_i \\ a\sum_{i=1}^{n} x_i + nb &= \sum_{i=1}^{n} y_i \end{cases}$$

ce système 2×2 se résout facilement, on trouve

$$\begin{cases} a = \dfrac{co(x,y) - \mu_x \mu_y}{Var(x)} \\ b = \dfrac{co(x,y) + \mu_x co(x,x)}{Var(x)} \end{cases}$$

avec $\mu_x = \frac{1}{n}\sum_{i=1}^{n} x_i, \mu_y = \dfrac{1}{n}\sum_{i=1}^{n} y_i, \delta = n^2 Var(x), co(x,y) = \dfrac{1}{n}\sum_{i=1}^{n} x_i y_i.$

Programme

```
x=rand(n,1);
y=2*x+1 +0.1*(1-2*rand(n,1));
//calcul des valeurs statistiques
mx=sum(x);
my=sum(y); mxy=sum(x'*y);
mxx=sum(x'*x);
//construction de la matrice et du second membre
A=[[mxx, mx];[mx, n]];
F=[mxy;my];
// Résolution du système linéaire
U=A\F;
a=U(1);b=U(2);
z=a*x+b;
plot(x,y,'.b')
plot(x,z,'-r')
```

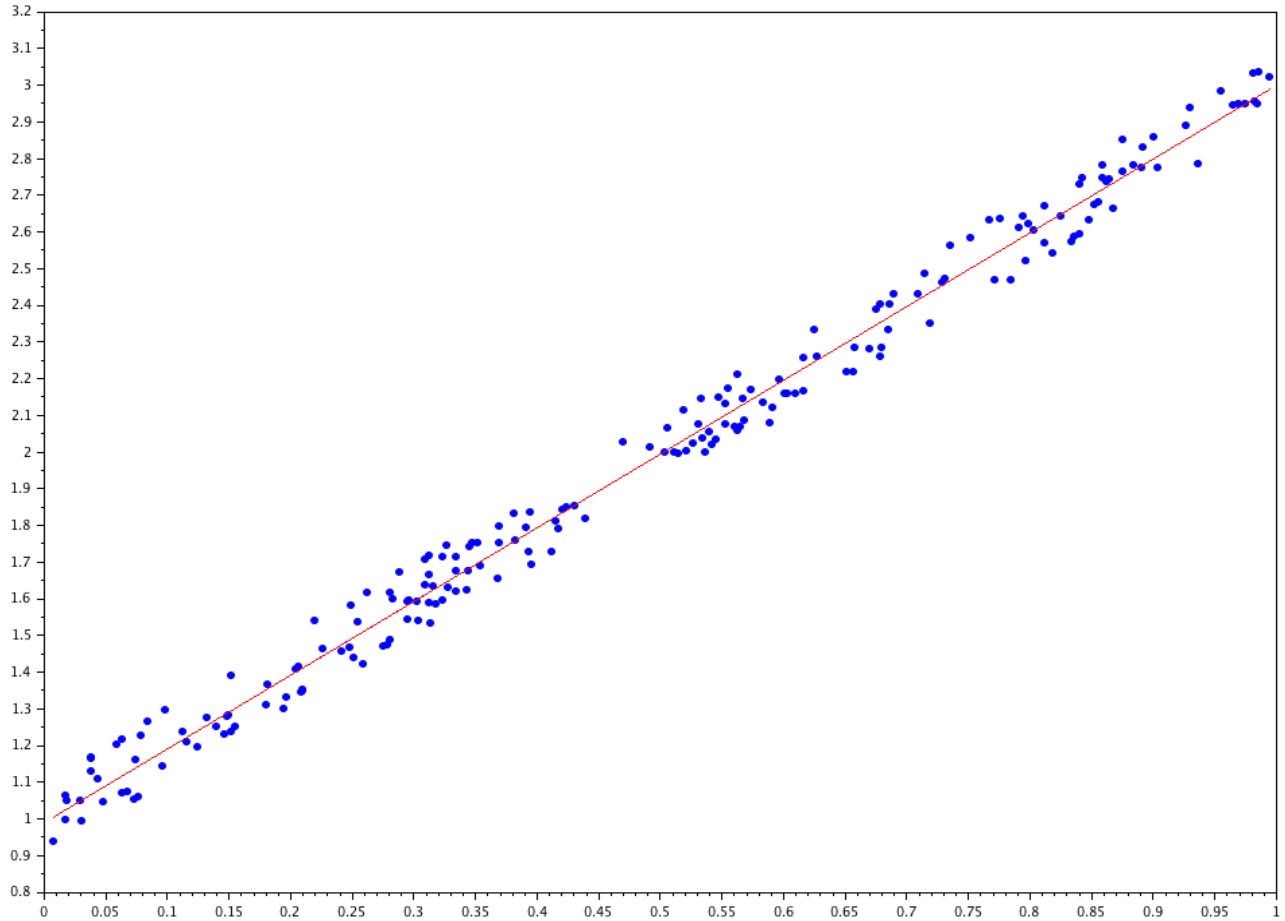


Figure – Régression $N = 200$. $a = 2.009888$, $b = 0.9897046$

Theorem

Soit (X_k) une suite de variables aléatoires réelles indépendantes et de même loi admettant une espérance μ . La moyenne empirique $\bar{X}_n = \frac{1}{n}\sum_{k=1}^{n} X_k$ converge en probabilité vers l'espérance μ : $\forall \epsilon > 0 \lim_{n\to+\infty} P(|\bar{X}_n - \mu| > \epsilon) = 0$

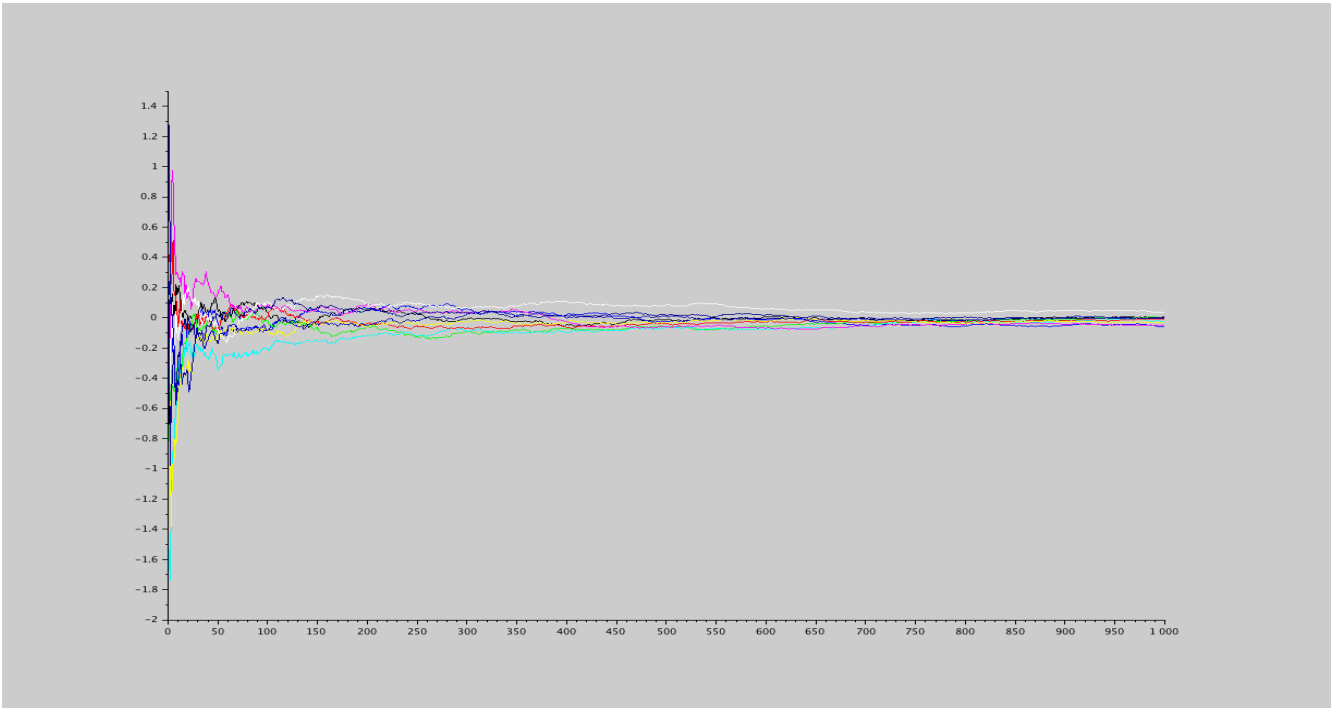


Figure – Illustration Loi G nombres