Informatique et Algorithmique – Chapitre 6

Probabilités avec Scilab

I Générateur de nombres (pseudo) aléatoires : la fonction rand

Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin.
John von Neumann [1]

1) Introduction

Simuler une variable aléatoire réelle d'une loi donnée à l'aide d'un ordinateur consiste à construire un nombre réel que l'on peut assimiler à une réalisation $X(\omega)$ d'une variable aléatoire X ayant cette loi.

Ce procédé repose en général sur deux étapes :

- Simuler des variables aléatoires indépendantes [2] de loi uniforme sur $]0,1[$.
- Réaliser des opérations sur les simulations afin d'obtenir la réalisation d'une variable aléatoire de loi voulue (ce sera l'objet du paragraphe suivant).

La première étape pose des problèmes à la fois conceptuels et pratiques... notamment comment demander à un ordinateur de faire un choix aléatoire, alors que celui-ci est programmé pour n'effectuer qu'une suite d'instructions déterministes ? Aucune solution totalement satisfaisante n'a été proposée à ce jour. La plupart des générateurs sont fondés sur des calculs de congruences sur des grands nombres de façon déterministe (mais que l'on peut initialiser avec l'horloge de l'ordinateur pour ajouter une dose de hasard). On parle alors de générateurs de nombres pseudo-aléatoires : les nombres générés possèdent les propriétés apparentes d'une suite de variables aléatoires indépendantes et de loi uniforme sur $[0,1]$:

- La suite de nombres se comporte de façon chaotique, de sorte que les éléments successifs de cette suite de nombres semblent imprévisibles. Plus précisément, pour tout $k \in \mathbb{N}^*$, la connaissance des $k-1$ premiers termes de la suite ne semble pas nous donner d'information sur le $k^{\text{ième}}$ terme.
- La moyenne des premiers nombres de la suite semble se concentrer asymptotiquement (quand le nombre de termes moyennés tend vers $+\infty$) vers une quantité fixe.

Il existe plusieurs générateurs basés sur des phénomènes physiques (et certains sont brevetés et disponibles dans le commerce). Ils reposent par exemple sur des capteurs de bruit thermique dans les résistances de circuits électroniques, ou sur d'autres mécanismes basés sur la physique quantique, etc. Ces générateurs contiennent du *vrai* hasard mais ils sont encombrants, pas toujours fiables, pas facilement reproductibles et ne sont pas accessibles à une analyse mathématique rigoureuse.

2) La fonction `rand`

Scilab possède une fonction `rand` qui renvoie un nombre compris entre 0 et 1 et que l'on peut assimiler à la réalisation d'une variable aléatoire de loi uniforme sur $]0,1[$.

`rand()`	simule une variable aléatoire de loi uniforme sur $]0,1[$.
`rand(1,n)`	renvoie un vecteur de taille `n` contenant des simulations de variables aléatoires de loi uniforme sur $]0,1[$.
`rand(m,n)`	renvoie une matrice de taille `m`×`n` contenant des simulations de variables aléatoires de loi uniforme sur $]0,1[$.

1. John von Neumann (1903-1957) est un mathématicien et physicien américano-hongrois.
2. La notion de variables aléatoires indépendantes est au programme de deuxième année d'ECS.

Exemples :

```
-->rand()
 ans  =
    0.2113249
-->rand()
 ans  =
    0.7560439
```

```
-->rand(3,4)
 ans  =
    0.0002211   0.6283918   0.8782165   0.6623569
    0.3303271   0.8497452   0.0683740   0.7263507
    0.6653811   0.6857310   0.5608486   0.1985144
```

Il est à noter que le générateur de Scilab produit toujours la même séquence [1] de nombre (ce n'est pas étonnant : il est déterministe !). En effet, si on ferme Scilab et qu'on l'ouvre à nouveau, la séquence de nombres donnée par `rand()` commencera encore par `0.2113249`, puis `0.7560439`, etc. Pour éviter cela, on peut modifier la graine du générateur (c'est-à-dire la valeur initiale utilisée, qui vaut 0 par défaut). On peut réinitialiser aléatoirement en utilisant par exemple la date en secondes via les commandes `n=getdate('s'); rand('seed',n);`. On peut revenir à la graine par défaut en utilisant la commande `rand('seed',0);`.

Exemples :

```
-->n=getdate('s'); rand('seed',n);
-->rand()
 ans  =
    0.8475400
```

```
-->rand('seed',0);
-->rand()
 ans  =
    0.2113249
```

II Simulation de variables aléatoires

1) ... à partir de la fonction `rand`

a) Variable aléatoire de loi uniforme sur $[\![a,b]\!]$

Proposition 1. *Soit U une variable aléatoire de loi uniforme sur $]0,1[$. Soit $(a,b)\in\mathbb{Z}^2$ avec $a<b$. Posons $X=\lfloor (b-a+1)U\rfloor+a$. Alors X suit une loi $\mathcal{U}([\![a,b]\!])$.*

Démonstration. La variable aléatoire $(b-a)U$ prend ses valeurs dans $]0,b-a+1[$ donc $\lfloor (b-a+1)U\rfloor$ prend ses valeurs dans $[\![0,b-a]\!]$. Ainsi X prend ses valeurs dans $[\![a,b]\!]$. De plus, pour tout $k\in[\![a,b]\!]$,

$$\begin{aligned}\mathbb{P}(X=k)=\mathbb{P}(\lfloor (b-a+1)U\rfloor=k-a)&=\mathbb{P}(k-a\leqslant (b-a+1)U<k-a+1)\\&=\mathbb{P}\left(\frac{k-a}{b-a+1}\leqslant U<\frac{k-a+1}{b-a+1}\right)\\&=F_U\left(\frac{k-a+1}{b-a+1}\right)-F_U\left(\frac{k-a}{b-a+1}\right)\\&=\frac{k-a+1}{b-a+1}-\frac{k-a}{b-a+1}=\frac{1}{b-a+1}=\frac{1}{\mathrm{card}([\![a,b]\!])}\end{aligned}$$

Ainsi X suit une loi $\mathcal{U}([\![a,b]\!])$. □

Soit $(a,b)\in\mathbb{Z}^2$ avec $a<b$. La commande `X=a+floor((b-a+1)*rand())` simule la réalisation d'une variable aléatoire de loi $\mathcal{U}([\![a,b]\!])$.

1. La documentation de Scilab nous apprend que le générateur de nombres aléatoires utilisé par `rand` est de type URAND (Universal Random Number Generator). Il utilise par défaut la suite de nombres $(u_n)_{n\in\mathbb{N}}$ définie par

$$\forall n\in\mathbb{N},\qquad x_{n+1}\equiv (ax_n+c)\,[m]\,,\qquad u_n=\frac{x_n}{m},$$

avec $x_0=0$, $m=2^{31}$, $a=843314861$, $c=453816693$. On dit qu'il s'agit d'un générateur linéaire à congruence. On remarque que $x_1=\dfrac{c}{m}\approx 0.2113249$.

Plus généralement `X=a+floor((b-a+1)*rand(m,n))` renvoie une matrice de taille $m \times n$ contenant des réalisations indépendantes de variables aléatoires de loi $\mathcal{U}(\llbracket a, b \rrbracket)$.

Exemples :

```
-->X=2+floor(5*rand())
 X  =
    4.
```

```
-->X=-3+floor(4*rand(2,7))
 X  =
  - 2.  - 2.  - 1.  - 3.    0.  - 1.  - 2.
  - 2.  - 1.  - 3.  - 1.  - 3.  - 3.    0.
```

b) Variable aléatoire de loi de Bernoulli

Proposition 2. *Soit U est une variable aléatoire de loi uniforme sur $]0,1[$. Soient $p \in]0,1[$ et X la variable aléatoire telle que*

$$\forall \omega \in \Omega, \qquad X(\omega) = \begin{cases} 1 & \text{si} \quad U(\omega) < p \\ 0 & \text{si} \quad U(\omega) \geqslant p. \end{cases}$$

Alors X suit une loi $\mathcal{B}(p)$.

Démonstration. La variable aléatoire X prend deux valeurs 0 et 1. De plus

$$\mathbb{P}(X = 1) = \mathbb{P}(U < p) = \mathbb{P}(U \leqslant p) = F_U(p) = p.$$

Ainsi X suit une loi $\mathcal{B}(p)$. □

Soit $p \in]0,1[$. La commande `X=(rand()<p)` simule la réalisation d'une variable aléatoire de loi $\mathcal{B}(p)$. Plus précisément la variable `X` contient la valeur 1 (ou `T`) avec probabilité p et 0 (ou `F`) avec probabilité $1-p$.

Plus généralement `X=(rand(m,n)<p)` renvoie une matrice de taille $m \times n$ contenant des réalisations de variables aléatoires de loi $\mathcal{B}(p)$.

Exemples :

```
-->rand()<0.7, rand()<0.7, rand()<0.7,
 ans  =
  T
 ans  =
  T
 ans  =
  F
```

```
-->X=(rand(5,4)<0.3)
 X  =
  T F F F
  F F T F
  T F F T
  F T T F
  T F F F
```

c) Variable aléatoire de loi binomiale

Rappelons qu'une variable aléatoire de loi binomiale de paramètres $n \in \mathbb{N}^*$ et $p \in]0,1[$ compte le nombre de succès (de 1) lors de n répétitions indépendantes d'une épreuve de Bernoulli de paramètre p. On peut donc voir [1] une variable aléatoire de loi $\mathcal{B}(n,p)$ comme la somme de n variables aléatoires indépendantes de loi $\mathcal{B}(p)$.

Nous en déduisons que la commande `X=sum((rand(1,n)<p))` simule la réalisation d'une variable aléatoire de loi $\mathcal{B}(n,p)$.

Exemples :

```
-->Z=(rand(1,15)<0.6)
 Z  =
  T T T F F T T F T F T T T F F
-->X=sum(Z)
 X  =
    9.
```

```
-->sum(rand(1,1000)<0.9)
 ans  =
    894.
-->sum(rand(1,1000)<0.2)
 ans  =
    231.
```

1. Ce sera montré rigoureusement en deuxième année.

d) Variable aléatoire de loi géométrique

Rappelons qu'une variable aléatoire de loi géométrique de paramètre $p \in]0,1[$ compte le nombre de répétitions d'une épreuve de Bernoulli de paramètre p nécessaires pour obtenir un succès.

Nous en déduisons que la commande

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

simule la réalisation d'une variable aléatoire de loi $\mathcal{G}(p)$. En effet cet algorithme consiste à simuler des variables aléatoires de loi $\mathcal{B}(p)$ jusqu'à ce qu'on obtienne un succès (un 1) et à compter le nombre d'essais.

Exemples :

```
-->X=1; while (rand()>=0.5);
                    X=X+1; end; X
 X  =
    3.
```

```
-->Y=1; while (rand()>=0.1);
                    Y=Y+1; end; Y
 Y  =
    8.
```

e) Variable aléatoire de loi uniforme

Soit $(a,b) \in \mathbb{R}^2$ avec $a < b$. Rappelons que, si U suit une loi uniforme sur $]0,1[$, alors $X = (b-a)U+a$ suit une suit une loi uniforme sur $]a,b[$.

Nous en déduisons que la commande `X=a+(b-a)*rand()` simule la réalisation d'une variable aléatoire de loi $\mathcal{U}(]a,b[)$.

Plus généralement `X=a+(b-a+1)*rand(m,n)` renvoie une matrice de taille $m \times n$ contenant des réalisations indépendantes de variables aléatoires de loi $\mathcal{U}(]a,b[)$.

Exemples :

```
-->a=-100; b=2; a+(b-a+1)*rand()
 ans  =
  - 68.731307
-->a=-4; b=10; a+(b-a+1)*rand()
 ans  =
  - 2.4738892
-->a=3; b=7; a+(b-a+1)*rand()
 ans  =
    6.9479155
```

```
-->a=-5; b=5; X=a+(b-a+1)*rand(7,3)
 X  =
    1.0560969 - 0.3350905   0.4304299
  - 2.1940774 - 0.0364293   2.3119927
    3.9005907 - 1.9463221   1.2518854
    3.3862387   5.080067    1.8904607
    5.0058329   4.3093667   3.7820935
    1.3356004 - 1.3903816 - 2.188643
    2.3416096   1.6902812   0.2503213
```

f) Variable aléatoire de loi exponentielle

Proposition 3. *Soit U une variable aléatoire de loi uniforme sur $]0,1[$. Soit $a \in \mathbb{R}_+^*$.*
Alors $1-U>0$ presque sûrement et $X = -\dfrac{1}{a}\ln(1-U)$ est une variable aléatoire de loi $\mathcal{E}(a)$.

Démonstration. Soit $t \in \mathbb{R}$. On a $F_X(t) = \mathbb{P}(X \leqslant t) = \mathbb{P}(1-U \geqslant e^{-at}) = \mathbb{P}(U \leqslant 1-e^{-at})$.

- Si $t<0$, alors $1-e^{-at}<0$ donc $F_X(t) = \mathbb{P}(U \leqslant 1-e^{-at}) = 0$.
- Si $t \geqslant 0$ alors $1-e^{-at} \in [0,1[$ et donc $F_X(t) = \mathbb{P}(U \leqslant 1-e^{-at}) = 1-e^{-at}$.

Ainsi F_X est la fonction de répartition d'une loi $\mathcal{E}(a)$. Nous en déduisons que X est une variable aléatoire de loi $\mathcal{E}(a)$. □

Soit $a > 0$. La commande `X=-log(1-rand())/a` simule ainsi la réalisation d'une variable aléatoire de loi $\mathcal{G}(a)$.

Exemples :

```
-->a=8; X=-log(1-rand())/a
 X  =
    0.2989328
-->-log(1-rand())
 ans  =
    0.2254493
```

```
-->-log(1-rand(3,3))/0.25
 ans  =

    1.6379744   10.693333   11.133384
    2.2612048   5.7148055   4.8033538
    1.8506441   5.1256179   3.9588708
```

2) ... avec la fonction `grand`

`grand(1,1,'***',###)`	simule une variable aléatoire de loi *** avec les paramètres ###
`grand(1,n,'***',###)`	renvoie un vecteur de taille n contenant des simulations de variables aléatoires de loi *** avec les paramètres ###
`grand(m,n,'***',###)`	renvoie une matrice de taille m×n contenant des simulations de variables aléatoires de loi *** avec les paramètres ###

a) Le cas des variables discrètes

`grand(1,1,'uin',a,b)`	simule une v.a.r de loi $\mathcal{U}(\llbracket \mathtt{a}, \mathtt{b} \rrbracket)$
`grand(1,1,'bin',1,p)`	simule une v.a.r de loi $\mathcal{B}(\mathtt{p})$
`grand(1,1,'bin',n,p)`	simule une v.a.r de loi $\mathcal{B}(\mathtt{n}, \mathtt{p})$
`grand(1,1,'geom',p)`	simule une v.a.r de loi $\mathcal{G}(\mathtt{p})$
`grand(1,1,'poi',a)`	simule une v.a.r de loi $\mathcal{P}(\mathtt{a})$

Exemples :

```
-->grand(4,4,'uin',-2,2)
 ans  =
    2.  - 2.    2.    0.
  - 2.  - 2.  - 2.  - 1.
    1.  - 2.    2.    2.
    1.  - 1.    2.    2.
-->grand(2,4,'bin',1,0.4)
 ans  =
    1.    0.    1.    0.
    1.    1.    0.    1.
```

```
-->grand(1,1,'bin',100,0.75)
 ans  =
    69.
-->grand(1,1,'geom',0.5)
 ans  =
    3.
-->grand(1,1,'poi',10)
 ans  =
    6.
```

b) Le cas des variables continues

`grand(1,1,'unf',0,1)`	simule une v.a.r de loi $\mathcal{U}(]0,1[)$
`grand(1,1,'unf',a,b)`	simule une v.a.r de loi $\mathcal{U}(]\mathtt{a}, \mathtt{b}[)$
`grand(1,1,'exp',1/a)`	simule une v.a.r de loi $\mathcal{E}(\mathtt{a})$
`grand(1,1,'nor',0,1)`	simule une v.a.r de loi $\mathcal{N}(0,1)$
`grand(1,1,'nor',m,s)`	simule une v.a.r de loi $\mathcal{N}(\mathtt{m}, \mathtt{s})$

Exemples :

```
-->grand(4,2,'unf',-100,100)
 ans  =
  - 57.958185 - 92.71175
    53.103358 - 62.625479
  - 89.756715 - 18.253768
    59.039981 - 2.047121
```

```
-->grand(2,3,'unf',0,1)
 ans  =
    0.4747587   0.4220877   0.1738652
    0.3922270   0.6554779   0.1711867
-->grand(1,1,'exp',1/3)
 ans  =
    0.1959833
```

```
-->grand(1,1,'nor',5,0.5)
 ans  =
    5.4286616
```

```
-->grand(2,3,'nor',0,1)
 ans  =
    0.4218018   0.1112611 - 0.1806437
    1.8865999   0.7886150   0.2364817
```

III Représentation graphique

1) Représentation graphique de loi de variables aléatoires

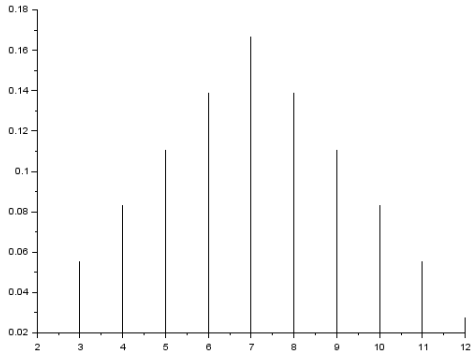
a) Diagramme en bâtons de lois discrètes

Si la variable aléatoire X suit une loi finie, alors on peut la représenter par un diagramme en bâtons via la commande `plot2d3(val,prob)` ou `bar(val,prob)` où `val` est un vecteur contenant la liste des valeurs possibles de X et `prob` est un vecteur contenant les probabilités respectives de ces valeurs.

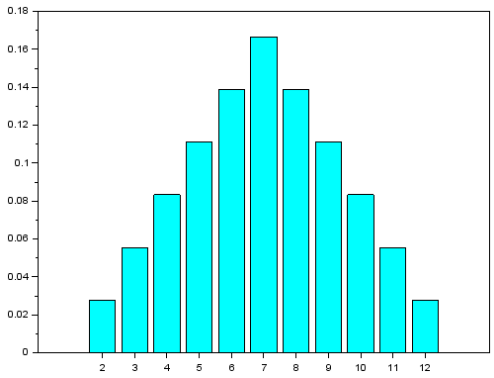
Exemples : *Si X représente la somme des chiffres lorsqu'on lance deux dés, alors la loi de X peut être représenter par le tableau suivant :*

k	2	3	4	5	6	7	8	9	10	11	12
$\mathbb{P}(X=k)$	$\frac{1}{36}$	$\frac{1}{18}$	$\frac{1}{12}$	$\frac{1}{9}$	$\frac{5}{36}$	$\frac{1}{6}$	$\frac{5}{36}$	$\frac{1}{9}$	$\frac{1}{12}$	$\frac{1}{18}$	$\frac{1}{36}$

```
-->val=2:12;
-->prob=[1,2,3,4,5,6,5,4,3,2,1]/36;
-->clf(); plot2d3(val,prob)
```



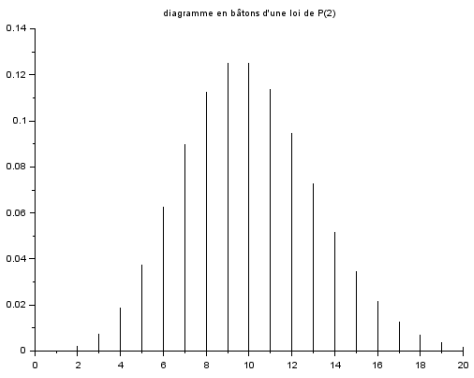
```
-->val=2:12;
-->prob=[1,2,3,4,5,6,5,4,3,2,1]/36;
-->clf(); bar(val,prob,'c')
```



Si la variable aléatoire X suit une loi discrète infinie, alors on peut la représenter par un diagramme en bâtons en se limitant aux valeurs dont les probabilités sont significatives.

Exemple : *Si X suit une loi de Poisson de paramètre 10, alors on peut montrer que $\mathbb{P}(X=k) < 0,001$ dès que $k \geqslant 21$. On peut donc limiter la représentation graphique de la loi aux 21 premières valeurs :*

```
-->val=0:20; a=10; x=exp(-a); prob=[x];
-->for k=1:20; x=x*a/k; prob=[prob,x]; end
//On remplit tour à tour le vecteur prob
//des valeurs successives des probabilités
-->plot2d3(val,prob)
-->title('diagramme en bâtons
       d''une loi de P('+string(a)+')')
```

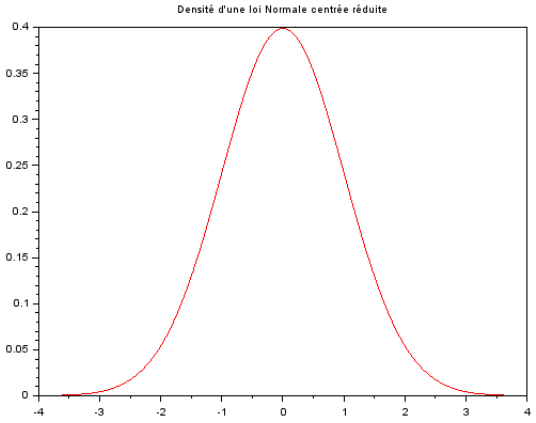


b) Fonctions de densité

On se réfère au TP n° 7.

Exemple :

```
-->x=linspace(-4,4,1000);
-->y=exp(-x.*x/2)/sqrt(2*%pi);
-->clf(); plot(x,y,'r')
-->title('densité d'une loi Normale
                    centrée réduite')
```

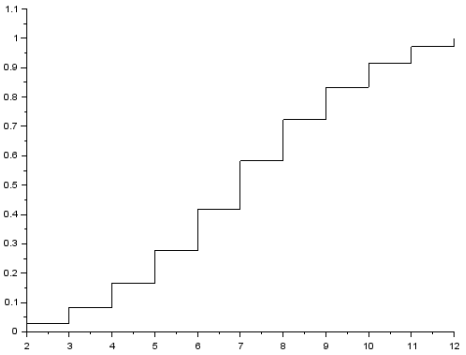


c) Fonctions de répartition

Pour tracer la fonction de répartition d'une variable aléatoire réelle finie X, on utilise la commande `plot2d2(val,cprob)` où `val` est un vecteur contenant la liste des valeurs possibles de X (des sauts dans la courbe de la fonction de répartition) et `cprob` est un vecteur contenant les probabilités respectives d'être inférieur ou égale à ces valeurs. Si `prob` est un vecteur contenant les probabilités respectives de ces valeurs, la fonction `cumsum` appliquée à `prob` renvoie le vecteur `cprob`.

Exemple : *Reprenons l'exemple où X représente la somme des chiffres lorsqu'on lance deux dés.*

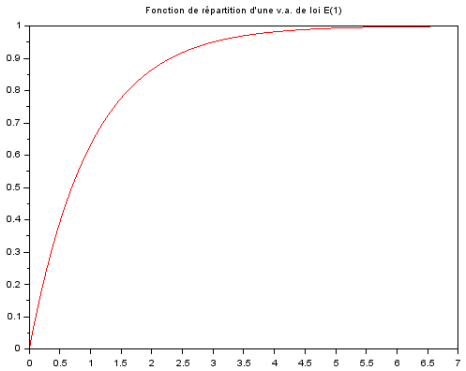
```
-->cumsum([1,2,3,1])
 ans  =
    1.    3.    6.    7.
-->val=2:12;
-->prob=[1,2,3,4,5,6,5,4,3,2,1]/36;
-->cprob=cumsum(prob);
-->clf(); plot2d2(val,cprob)
```



Pour les fonctions de répartitions de lois continues, on se réfère au TP n° 7.

Exemple :

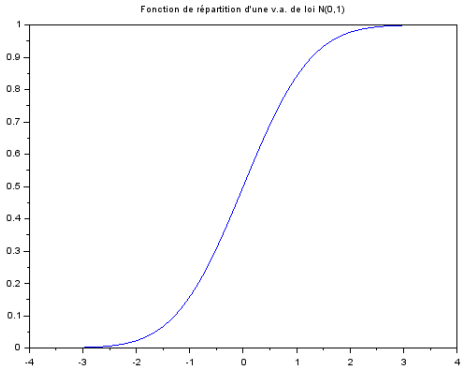
```
-->a=1; x=linspace(0,7,1000);
-->y=1-exp(-a*x);
-->clf(); plot(x,y,'r')
-->title('Fonction de répartition d''une
         v.a. de loi E('+string(a)+')')
```



La fonction de répartition d'une loi Normale est prédéfinie sans Scilab. Il s'agit de la fonction `cdfnor`. Nous renvoyons à l'aide Scilab pour tous les détails sur son utilisation.

Exemple :

```
-->m=0; s=1; clf();
-->x=linspace(-4,4,1000);
-->y=cdfnor("PQ",x,m*ones(x),s*ones(x));
-->plot(x,y,'b')
-->title('Fonction de répartition d''une
v.a. de loi N('+string(m)+','+string(s^2)+')')
```



2) Représentation graphique de phénomènes aléatoires

a) Diagramme en batôn

Supposons que l'on dispose de réalisations indépendantes d'une variable aléatoire réelle finie X, stockées dans un vecteur Scilab `X`. La commande `tabul(X,'i')` va renvoyer une matrice dont

- la première colonne range par ordre croissant les valeurs prises par X,
- la deuxième colonne correspond aux effectifs respectifs des valeurs prises par X.

La commande `T=tabul(X,'i'); T(:,2)/sum(T(:,2))` remplace les effectifs respectifs par les fréquences.

La commande `T=tabul(X,'i'); plot2d2(T(:,1),T(:,2))` construit le diagramme en bâtons (dit empirique) des effectifs des réalisations de X, stockées dans `X`.

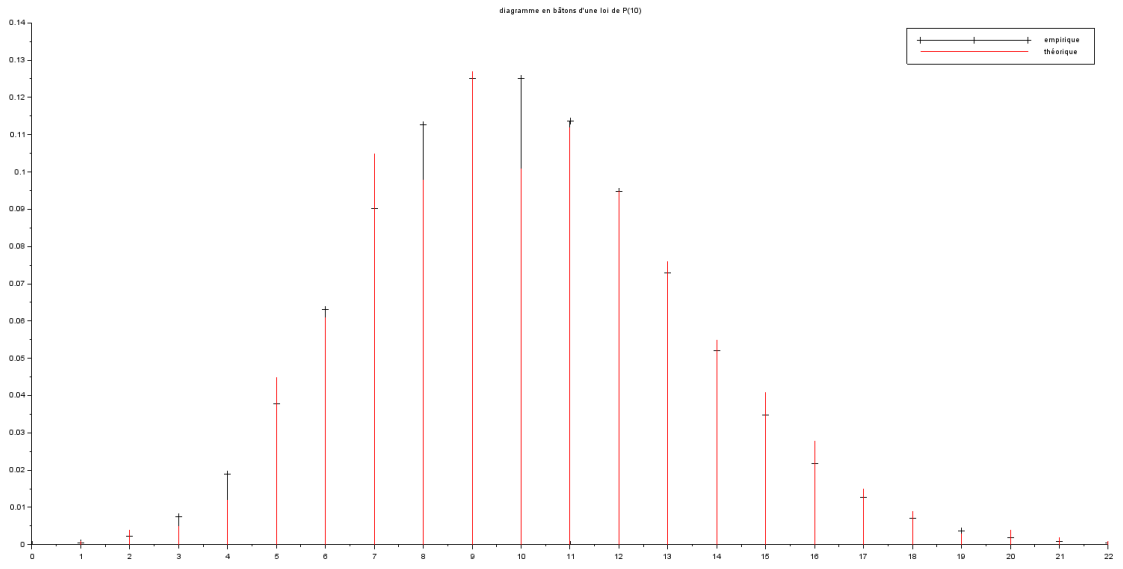
La commande `T=tabul(X,'i'); plot2d2(T(:,1),T(:,2)/sum(T(:,2)))` construit le diagramme en bâtons (dit empirique) des effectifs des réalisations de X, stockées dans `X`.

Si le nombre de réalisations est élevé, alors on peut superposer le diagramme en bâtons des réalisations avec celui de la loi théorique et observer une forte similitude (c'est une conséquence de la loi des grands nombres).

Cette méthode fonctionne aussi si X prend des valeurs dénombrables (mais on ne représente que les bâtons correspondants aux valeurs de probabilités significatives).

Exemple :

```
//On simule n=1000 réalisations d'une loi de Poisson de paramètre a=10
-->n=500; a=10; X=grand(1,n,'poi',a);
//On trace le diagramme théorique d'une loi de Poisson de paramètre a
//(limité aux premières valeurs)
-->M=max(T(:,1)); val=0:M; x=exp(-a); prob=[x];
-->for k=1:M; x=x*a/k; prob=[prob,x]; end
-->clf(); plot2d3(val,prob,style=-1)
//On lui superpose le diagramme empirique
-->T=tabul(X,'i');
-->plot2d3(T(:,1),T(:,2)/sum(T(:,2)),style=5);
-->title('diagramme en bâtons d''une loi de P('+string(a)+')')
-->legend('empirique','théorique')
```

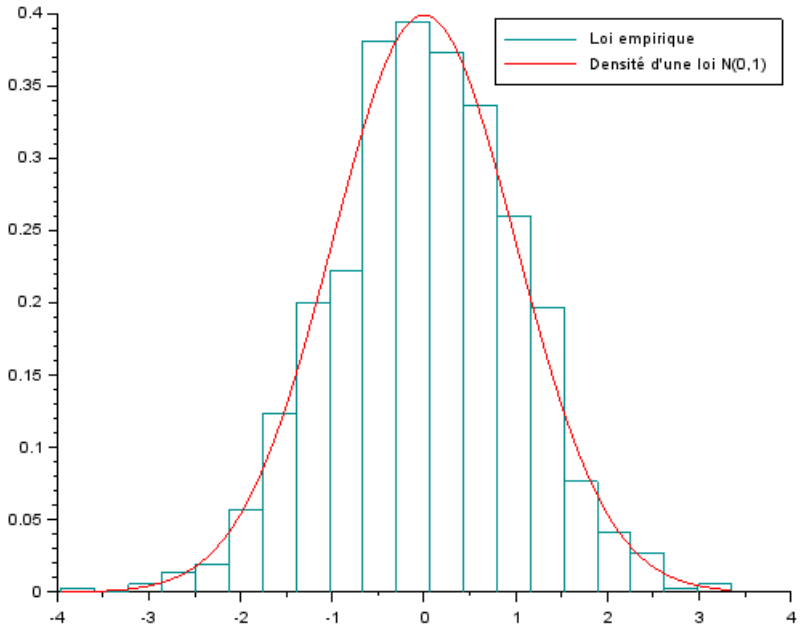


b) Histogramme

Supposons que l'on dispose de réalisations indépendantes d'une variable aléatoire réelle X à densité stockées dans un vecteur Scilab `X`. On réalise un histogramme à partir de ses données (à r classes) `histplot(r,X)`. Si le nombre de réalisations est élevé, alors on peut superposer l'histogramme des réalisation avec la courbe de la densité de la loi théorique et observer une forte similitude (c'est une conséquence de la loi des grands nombres).

Exemple :

```
//On simule n=1000 réalisations d'une loi Normale centrée réduite
-->n=1000; X=grand(1,n,'nor',0,1);
//On construit le diagramme empirique
-->clf(); histplot(20,X,style=16)
//On lui superpose la courbe théorique de la densité
-->m=min(X); M=max(X); x=linspace(m,M,1000);
-->y=exp(-x.*x/2)/sqrt(2*%pi);
-->plot(x,y,'r')
-->legend('Loi empirique','Densité d''une loi N(0,1)')
```



c) Fonction de répartition empirique

Supposons que l'on dispose de réalisations indépendantes d'une variable aléatoire réelle X, stockées dans un vecteur Scilab `X`. On utilise la commande `T=tabul(X,'i'); T(:,2)=T(:,2)/sum(T(:,2))` pour obtenir un tableau des fréquences des valeurs observées (ces valeurs étant rangées par ordre croissant). La commande `T(:,2)=cumsum(T(:,2))` remplace les fréquences par les fréquences cumulées.

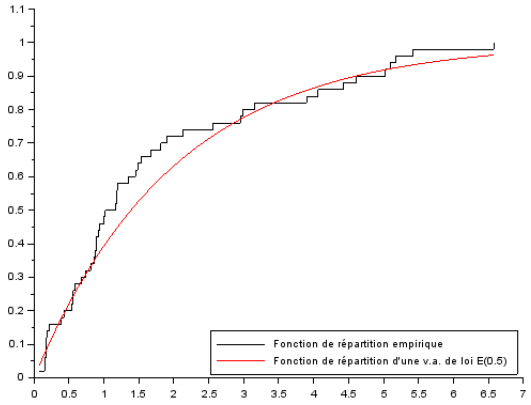
On utilise enfin `plot2d2(T(:,1),T(:,2))` qui trace alors la fonction de répartition empirique des valeurs observées.

Si le nombre de réalisations est élevé, alors on peut superposer la fonction de répartition empirique avec celle de loi théorique et observer une forte similitude (c'est une conséquence de la loi des grands nombres).

Exemples :

- Pour un échantillon de taille 50 :

```
-->a=0.5; X=grand(1,50,'exp',1/a);
-->T=tabul(X,'i'); T(:,2)=T(:,2)/sum(T(:,2));
-->T(:,2)=cumsum(T(:,2));
-->clf(); plot2d2(T(:,1),T(:,2))
-->x=linspace(min(T(:,1)),max(T(:,1)),1000);
//On lui superpose la courbe de densité
théorique
-->y=1-exp(-x/2);
-->plot(x,y,'r')
-->legend('Fonction de répartition
empirique', 'Fonction de répartition
d''une v.a. de loi E('+string(a)+')',4)
```



- Pour un échantillon de taille 500 :

```
-->a=0.5; X=grand(1,500,'exp',1/a);
-->T=tabul(X,'i'); T(:,2)=T(:,2)/sum(T(:,2));
-->T(:,2)=cumsum(T(:,2));
-->clf(); plot2d2(T(:,1),T(:,2))
-->x=linspace(min(T(:,1)),max(T(:,1)),1000);
//On lui superpose la courbe de densité
théorique
-->y=1-exp(-x/2);
-->plot(x,y,'r')
-->legend('Fonction de répartition
empirique', 'Fonction de répartition
d''une v.a. de loi E('+string(a)+')',4)
```