Illustration de convergences de suites de variables aléatoires

Charles Suquet
Laboratoire P. Painlevé (UMR 8524 CNRS)
Bât. M2 U.F.R. de Mathématiques
Université des Sciences et Technologies de Lille
F-59655 Villeneuve d'Ascq Cedex France

Charles.Suquet@univ-lille1.fr

1 Introduction

Ce document reprend la présentation faite sur vidéo projecteur lors de l'atelier que j'ai animé le 16 avril 2004 aux Journées Académiques sur le Hasard à Lille. Il s'agissait de montrer quelques illustrations graphiques des deux théorèmes limites les plus importants du calcul des probabilités, la loi forte des grands nombres (LFGN) et le théorème limite central (TLC). Le document a été complété et précisé de façon à pouvoir être lu et compris par des personnes n'ayant pas assisté à l'atelier. J'y ai inclus tous les codes Scilab qui avaient été assez peu montrés lors de l'atelier. On peut lire ce document en se contentant de regarder comment les figures illustrent les résultats théoriques rappelés succinctement au début de chaque section. Si l'on s'intéresse à l'expérimentation présentée ici, il est facile de se procurer et d'installer Scilab sur un ordinateur. Scilab est un logiciel libre développé par l'INRIA. Il existe en version Linux et Windows. On peut le télécharger depuis le site de l'INRIA à l'URL :

http://scilabsoft.inria.fr/

On trouvera également sur ce site de la documentation. Tous les scripts et fonctions décrits dans ce document sont téléchargeables à partir de ma page personnelle à l'URL :

http://math.univ-lille1.fr/~suquet/

On y trouvera aussi mes polycopiés cités dans le texte.

2 Autour de la LFGN

Dans toute cette section, les X_k sont des variables aléatoires réelles, définies sur le même espace probabilisé, indépendantes et de même loi (i.i.d.). On pose

$$S_n := \sum_{k=1}^{n} X_k.$$

Théorème 1 (Kolmogorov-Khintchine)**.** *Si (X_k) est une suite de variables aléatoires réelles i.i.d.,*

$$\frac{S_n}{n} \text{ converge presque sûrement } \Leftrightarrow \mathbf{E}|X_1| < +\infty. \tag{1}$$

Lorsqu'il y a convergence, la limite est $\mathbf{E}X_1$.

2.1 Convergence des fréquences

Une application importante de ce théorème est la convergence des fréquences de succès dans une suite d'épreuves répétées de Bernoulli indépendantes. Ce résultat explique *a posteriori* l'approche fréquentiste dans la définition d'une probabilité.

Voici une fonction Scilab (écrite dans un fichier `lfgn.sci`) illustrant ceci.

```
function [y]=lfgn(n,p)
// calcule S_n/n pour un n échantillon de Bern(p)
// affiche le graphe de la ligne polygonale de sommets (k,S_k/k)
// et son asymptote
U=rand(1:n,'uniform');
E=p.*ones(1:n);
X=bool2s(U<p);
S=cumsum(X);
M=S./(1:n);
y=M(n);
xbasc(); plot2d((1:n),[E' M'],[5,2])
endfunction
```

Essayer en tapant [1] dans l'interpréteur de commandes Scilab :

```
-->;getf("/home/suquet/Enseignement/Scilab/LFGN/lfgn.sci");

-->lfgn(1000,0.72)

-->xbasc();lfgn(50000,0.72)
```

La fenêtre graphique de Scilab affiche alors les figures [2] 1 et 2 .

1. Pour charger la fonction, utiliser le menu `File->File operations` et cliquer sur `Getf` après avoir sélectionné le nom du fichier, ici `lfgn.sci`. Scilab répond en affichant la ligne `getf(...` avec le nom de chemin adapté à votre machine. L'instruction `xbasc()` efface le contenu de la fenêtre graphique.

2. Les figures donnent une trajectoire particulière, dépendant de l'état du générateur de nombres aléatoires au moment où on a lancé le calcul de la fonction `lfgn`.

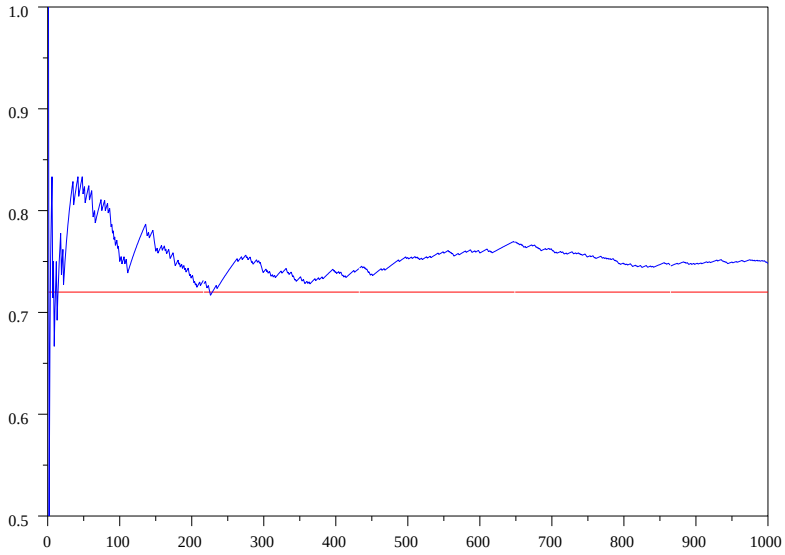

FIG. 1 – Ligne polygonale des fréquences pour $n = 1\,000$ et $X_1 \sim \text{Bern}(0.72)$

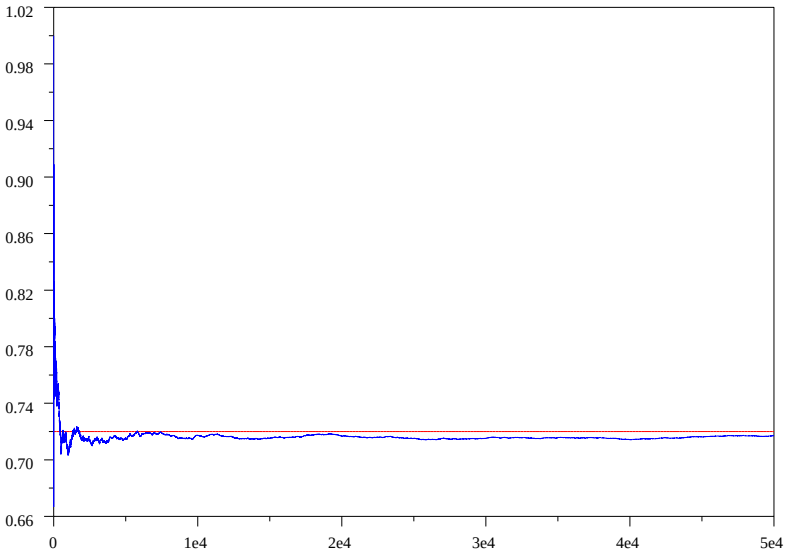

FIG. 2 – Ligne polygonale des fréquences pour $n = 50\,000$ et $X_1 \sim \text{Bern}(0.72)$

Le théorème 1 a une traduction statistique fondamentale : il permet de justifier

la convergence de la fonction de répartition empirique. Considérons une suite (Y_k) de variables aléatoires indépendantes et de même loi de fonction de répartition F. On définit la *fonction de répartition empirique* F_n construite sur l' *échantillon* $Y_1, \ldots, Y_n$ par

$$F_n(x) := \frac{1}{n} \sum_{k=1}^{n} \mathbf{1}_{\{Y_k \leq x\}}, \quad x \in \mathbb{R}. \tag{2}$$

Le théorème 1 appliqué aux variables aléatoires bornées $X_k = \mathbf{1}_{\{Y_k \leq x\}}$ nous donne immédiatement pour tout $x \in \mathbb{R}$ la convergence presque sûre de $F_n(x)$ vers $F(x)$ en remarquant que $\mathbf{E}X_1 = \mathbf{P}(Y_1 \leq x) = F(x)$. Ainsi une loi inconnue peut être reconstituée approximativement à partir de l'observation d'un échantillon de grande taille. En fait, on peut obtenir mieux que la convergence simple presque sûre de F_n vers F.

Théorème 2 (Glivenko-Cantelli)**.** *Soit (Y_k) une suite de variables aléatoires indépendantes, de même loi et (F_n) la suite de fonctions de répartition empiriques associées. Alors*

$$\|F_n - F\|_\infty := \sup_{x \in \mathbb{R}} |F_n(x) - F(x)| \xrightarrow[n \to +\infty]{\text{p.s.}} 0. \tag{3}$$

Voici une fonction Scilab illustrant ce théorème. Elle fait tracer en bleu la f.d.r. empirique F_n d'un n échantillon de la loi gaussienne $\mathfrak{N}(3,2)$ et en rouge la f.d.r. F de la loi limite $\mathfrak{N}(3,2)$. Pour construire F_n, on utilise le fait que $F_n(X_{n:k}) = k/n$, où $X_{n:k}$ désigne la k-ième statistique d'ordre de l'échantillon, *i.e.* la k-ième valeur obtenue dans le réarrangement croissant de l'échantillon.

```
function [y]=GlvkCan(n)
// illustration du théorème de Glivenko-Cantelli pour un n-échantillon
// de N(3,2)
X=3+2.*rand(1:n,'normal');
Xs=gsort(X,'g','i'); // tri croissant (='i') de X
Y=cumsum(ones(Xs)./n);
xbasc();
a=Xs(1)-1;b=Xs(n)+1;t=a:.01:b;
z=cdfnor("PQ",t,3.*ones(t),2.*ones(t));
plot2d(t,z,5)
plot2d2(Xs',Y',2)
y=Y(n);
endfunction
```

En essayant

```
-->;getf("/home/suquet/Enseignement/Scilab/LFGN/GlvkCan.sci");

-->GlvkCan(200)

-->xbasc();GlvkCan(800)
```

on obtient dans la fenêtre graphique de Scilab les figures 3 et 4.

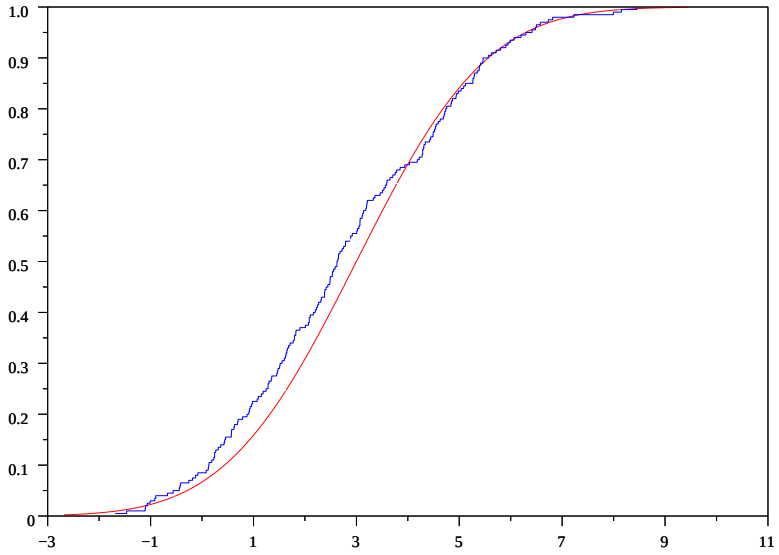

FIG. 3 – F.d.r. empirique pour un échantillon gaussien $\mathfrak{N}(3,2)$ de taille $n = 200$

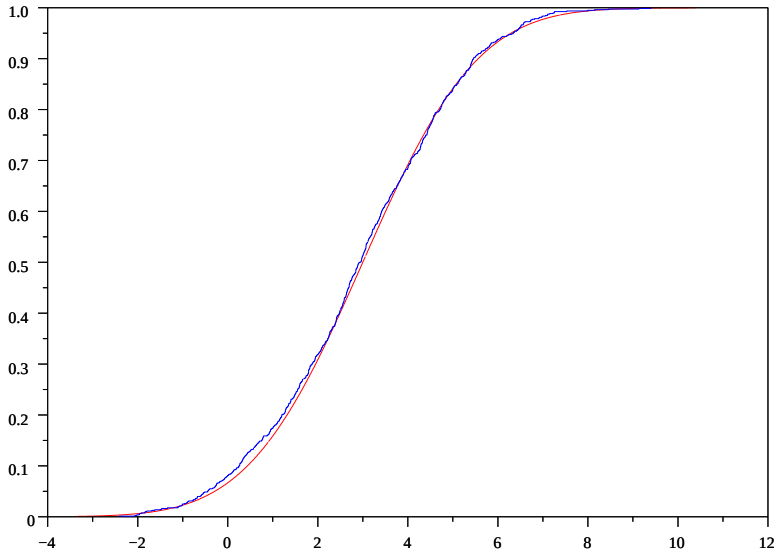

FIG. 4 – F.d.r. empirique pour un échantillon gaussien $\mathfrak{N}(3,2)$ de taille $n = 800$

2.2 Entonnoirs déterministes pour la courbe des S_n/n

Si les X_k sont indépendantes, identiquement distribuées et s'il existe des constantes a et b telles que $\mathbf{P}(a \leq X_1 \leq b) = 1$, alors

$$\forall \varepsilon > 0, \quad \mathbf{P}\Big(\Big|\frac{S_n - \mathbf{E}S_n}{n}\Big| \geq \varepsilon\Big) \leq 2\exp\Big(-n\frac{2\varepsilon^2}{(b-a)^2}\Big). \tag{4}$$

On peut utiliser ce type d'inégalité pour étudier quantitativement les fluctuations asymptotiques de S_n/n autour de $\mathbf{E}X_1$. Pour simplifier, on suppose désormais $a = 0$ et $b = 1$.

De (4) on déduit en exercice, que pour tout entier $N \geq 2$ et tout $\alpha > 1/2$,

$$\mathbf{P}\left\{\forall k > N,\ \Big|\frac{S_k}{k} - \mathbf{E}X_1\Big| \leq \sqrt{\frac{\alpha \ln k}{k}}\right\} \geq 1 - \frac{2}{2\alpha - 1}N^{1-2\alpha}. \tag{5}$$

Par exemple avec $\alpha = 1$,

$$\mathbf{P}\left\{\forall k > 200,\ \mathbf{E}X_1 - \sqrt{\frac{\ln k}{k}} \leq \frac{S_k}{k} \leq \mathbf{E}X_1 - \sqrt{\frac{\ln k}{k}}\right\} \geq 0,99.$$

On obtient ainsi un « entonnoir » déterministe qui avec une probabilité d'au moins $0,99$ encadre *jusqu'à l'infini* la ligne polygonale de sommets $(k, S_k/k)$. On illustre graphiquement cette inégalité en faisant tracer la ligne polygonale d'interpolation des S_k/k et les deux courbes encadrantes (l'entonnoir) correspondantes. On prend pour X_i une variable de Bernoulli. Le script `entonnoir.sce` présenté ci-dessous page 225 réalise ceci. Accessoirement, il donne un exemple de saisie interactive des données et d'initialisation du générateur de nombres aléatoires. Voici un exemple d'utilisation [3], la sortie graphique étant donnée figure 5.

```
-->;exec("/home/suquet/Enseignement/Scilab/LFGN/entonnoir.sce");
Initialisation generateur : taper un entier (9 chiffres max)-->568222
Rentrer la taille n de l'echantillon (n>10) : -->25000
Rentrer le parametre p de la loi de Bernoulli : -->0.34
Rentrer le parametre alpha de l'entonnoir (alpha>1/2): -->0.6
```

Remarque 3. La loi du logarithme itéré (voir [5] et [12]) donne des entonnoirs de la forme $\mathbf{E}X_1 \pm c(\ln\ln k)^{1/2}k^{-1/2}$, donc asymptotiquement meilleurs que ceux présentés ici. De plus S_k/k reste définitivement dans l'entonnoir avec probabilité 1. Néanmoins le rang N à partir duquel ceci se produit est *aléatoire* et on ne sait pas le contrôler.

3. Pour lancer ce script, utiliser le menu `File->File operations` et cliquer sur `Exec` après avoir sélectionné le nom du fichier, ici `entonnoir.sce`. Scilab répond en affichant la ligne `exec(...` avec le nom de chemin adapté à votre machine. Pour une deuxième exécution, il suffit d'utiliser le rappel des commandes avec les touches ↑ ou ↓. Une méthode plus fine de rappel de commande est de taper ! suivi des premières lettres du nom de la commande et d'un retour chariot.

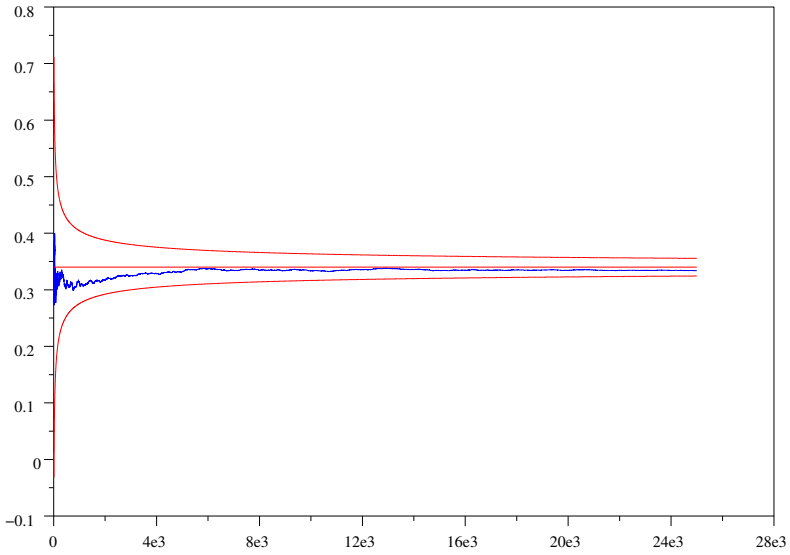

FIG. 5 – Entonnoir

```
//  entonnoir encadrant S_k/k pour un échantillon d'une Bernoulli,
germe=input('Initialisation générateur : taper un entier (9 chiffres max)');
n=input('Rentrer la taille n de l''échantillon (n>10) : ');
// Attention méchant piège avec l'apostrophe devant échantillon!!
p=input('Rentrer le paramètre p de la loi de Bernoulli : ');
alpha=input('Rentrer le paramètre alpha de l''entonnoir (alpha>1/2): ');
rand('seed',germe);
// On commence l'affichage à 10
// Il faut des vecteurs colonnes pour plot2d
t=(10:n)';
X=bool2s(rand(t,'uniform')<p);
S=cumsum(X);
// correction liée au debut à 10
S9=sum(bool2s(rand(1:9,'uniform')<p)); S=S9+S;
f=S./t; //suite des fréquences
enton=sqrt(alpha.*log(t)./t);
y=p.*ones(t)+enton;
z=p.*ones(t)-enton;
// affichage  f en bleu (2), branches entonnoir y et z en rouge (5)
xbasc(); plot2d([t t t], [f y z], [2 5 5])//
xsegs([0 n],[p p],5); // tracé en rouge de l'asymptote
```

2.3 Jeu de pile ou face et lois singulières

Tout $x \in [0,1]$ admet un développement en base 2 de la forme $x = \sum_{k\geq 1} c_k 2^{-k}$, les $c_k \in \{0,1\}$ étant les chiffres binaires (ou bits) de x. Ce développement est unique, sauf si x est un nombre dyadique, *i.e.* de la forme $x = m2^{-j}$, avec m et j entiers. Dans ce cas il y a deux développements, l'un d'eux ayant tous ses chiffres binaires nuls à partir d'un certain rang (développement propre) et l'autre tous ses chiffres binaires égaux à 1 à partir d'un certain rang (développement impropre). Par exemple les deux développements du dyadique 3/8 sont $(0,1,1,0,0,0,0\dots)$ et $(0,1,0,1,1,1,1,\dots)$. On note D l'ensemble des nombres dyadiques de $[0,1]$. Pour $x \in [0,1]$, définissons $s_n(x) := \sum_{k=1}^n c_k$, où les c_k sont les chiffres de l'unique développement binaire de x si $x \in [0,1]\setminus \mathrm{D}$, de son développement propre si $x \in \mathrm{D}$. Autrement dit, $s_n(x)$ est le nombre de chiffres 1 parmi les n premiers bits de x. Posons

$$A_p := \Big\{ x \in [0,1];\ \frac{s_n(x)}{n} \xrightarrow[n\to+\infty]{} p \Big\}.$$

Remarquons au passage que si $x \in \mathrm{D}$, la fréquence de chiffres 1 tend vers 0 pour son développement propre (et vers 1 pour son développement impropre). Comme $0 < p < 1$, ceci montre que A_p et D sont disjoints. Par unicité de la limite, il est clair également que si p et p' sont deux éléments distincts de $]0,1[$, $A_p \cap A_{p'} = \emptyset$.

Soit maintenant $(X_k)_{k\geq 1}$ une suite de variables aléatoires de Bernoulli indépendantes de même paramètre $p \in]0,1[$. La série $\sum_{k\geq 1} 2^{-k} X_k$ convergeant p.s., on note U sa somme. La loi de cette variable aléatoire U est donc une mesure de probabilité sur $[0,1]$ que nous noterons μ_p. Par la loi forte des grands nombres,

$$\mu_p(A_p) = \mathbf{P}(U \in A_p) = \mathbf{P}\left(\frac{1}{n}\sum_{k=1}^n X_k \xrightarrow[n\to+\infty]{} p \right) = 1.$$

Comme μ_p est une probabilité et $A_{p'}$ est disjoint de A_p, nécessairement $\mu_p(A_{p'}) = 0$. Ceci implique que $\mu_{p'}$ ne peut avoir de densité par rapport à μ_p. En effet s'il existait une telle densité f on aurait :

$$1 = \mu_{p'}(A_{p'}) = \int_{A_{p'}} f \,\mathrm{d}\mu_p = 0,$$

car $\mu_p(A_{p'})$ étant nul, l'intégrale par rapport à μ_p de toute fonction mesurable positive (même infinie) sur $A_{p'}$ est nulle. En échangeant les rôles de p et p', on voit que μ_p n'a pas de densité par rapport à $\mu_{p'}$. On dit que les deux mesures sont *étrangères*.

D'autre part on peut montrer par un calcul de fonction caractéristique que $\mu_{1/2}$ est la loi uniforme sur $[0,1]$, c'est à dire la trace de la mesure de Lebesgue sur $[0,1]$. Par conséquent pour $p \neq 1/2$, la loi μ_p *n'a pas de densité par rapport à la mesure de Lebesgue.*

Notons F_p la fonction de répartition de μ_p définie par

$$\forall x \in \mathbb{R}, \quad F_p(x) = \mathbf{P}(U \leq x) = \mu_p(]-\infty, x]).$$

On vérifie que F_p est continue sur $\mathbb{R}$ en montrant qu'elle n'a pas de saut, ce qui revient à prouver que $\mu_p(\{x\}) = 0$ pour tout $x \in \mathbb{R}$. C'est évident pour $x \notin A_p$. Pour $x \in A_p$, le

développement binaire $x = \sum_{k\geq 1} c_k 2^{-k}$ est unique, ce qui légitime l'équivalence :

$$U = x \Leftrightarrow \forall k \in \mathbb{N}^*,\ X_k = c_k.$$

On en déduit l'inclusion d'évènements : $\{U = x\} \subset \{\forall k = 1, \ldots, n,\ X_k = c_k\}$ pour tout $n \geq 1$. En posant $q = 1 - p$ et $r = \max(p, q)$ on obtient par indépendance des X_k :

$$\forall n \geq 1, \quad \mu_p(\{x\}) = \mathbf{P}(U = x) \leq \mathbf{P}(\forall k = 1, \ldots, n,\ X_k = c_k) = p^{s_n(x)} q^{n - s_n(x)} \leq r^n.$$

Comme $p \in]0, 1[$, il en est de même pour r, donc r^n tend vers 0 quand n tend vers $+\infty$. L'inégalité ci-dessus étant vraie pour tout $n \geq 1$, on en déduit que $\mu_p(\{x\}) = 0$.

Nous avons ainsi construit une famille infinie de *lois à fonction de répartition continue et sans densité* (au sens usuel, *i.e.* relativement à la mesure de Lebesgue). De telles lois sont appelées *lois singulières*.

La représentation graphique de F_p fait l'objet de l'exercice de T.P. suivant. On pourra en trouver un corrigé détaillé dans [14].

Le but de cet exercice est de tracer une représentation graphique de F_p. On note $F_{p,n}$ la fonction de répartition de

$$U_n = \sum_{k=1}^{n} 2^{-k} X_k.$$

1. *Expliciter et représenter graphiquement $F_{p,1}$ et $F_{p,2}$. On utilisera* `plot2d2`.
2. *Trouver un moyen simple pour obtenir de manière récursive la liste des masses ponctuelles de la loi de U_n, autrement dit les hauteurs de sauts de $F_{p,n}$.*
3. *Utiliser ceci pour tracer la représentation graphique de $F_{p,n}$. On écrira la suite des instructions dans un fichier* `.sce`. *Quelle valeur de n vous paraît raisonnable pour représenter F_p par $F_{p,n}$ à l'écran ?*

Voici le script ayant servi à réaliser la figure 7, voir [14] pour les explications.

```
// Tracé pour plusieurs valeurs du paramètre p de la fdr F_n de
//   U_n=\sum_{i=1}^n 2^{-i}X_i,  les X_i etant iid de loi Bern(p).
// Penser à adapter le vecteur de couleurs si on modifie la taille de p
// éviter les couleurs 8 et 34 qui sont celles du fond d'écran
n=13;
p=0.1:0.1:0.9; q=1-p;
x=0:2^{-n}:1;
F=zeros(length(x),length(p));
for k=1:length(p),
   M=[q(k) p(k)];
   for i=2:n,
      M=[M.*q(k) M.*p(k)];
   end
   F(:,k)=[cumsum(M) 1]';
end
xbasc()
plot2d2("onn",x',F, [2 3 4 5 6 5 4 3 2])
```

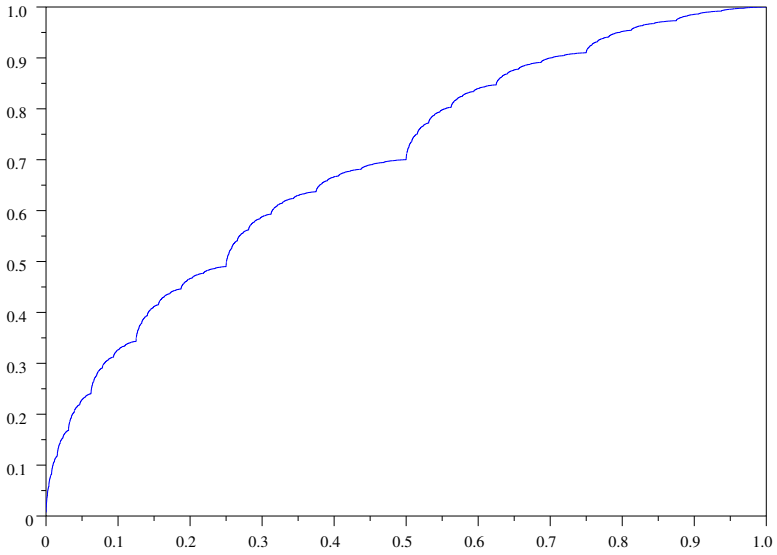

FIG. 6 – Représentation de $F_{p,n}$ avec $p = 0{,}3$ et $n = 14$

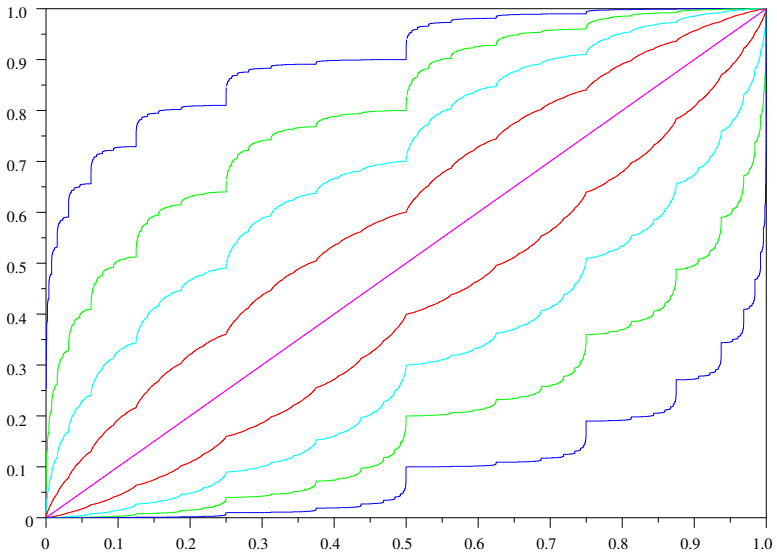

FIG. 7 – Représentation des $F_{p,n}$ avec $p = k/10$, $k = 1, \ldots, 9$ et $n = 13$

2.4 Le cas de la loi de Cauchy

D'après le théorème 1, si X_1 n'a pas d'espérance, alors S_n/n ne peut converger presque sûrement. On illustre cette remarque en examinant le cas où les X_k suivent la loi de Cauchy donnée par sa densité

$$f : \mathbb{R} \to \mathbb{R}, \quad t \longmapsto \frac{1}{\pi(1+t^2)}.$$

Pour cette loi, $\mathbf{E}|X_1| = +\infty$. Pour simuler une loi de Cauchy, on exploite le fait que si U est de loi uniforme sur $[0,1]$, $\tan(\pi(U-1/2))$ suit la loi de Cauchy. On observe de très grandes fluctuations de S_n/n, même pour les grandes valeurs de n. On peut d'ailleurs vérifier que S_n/n a même loi que X_1. Voici un petit script.

```
// Une v.a. de Cauchy ne vérifie pas la loi des grands nombres
// Illustration graphique
n=input('Rentrer la taille n de l''echantillon : ');
N=1:n; U=rand(N,'uniform');
X=tan(%pi.*(U-.5)); // échantillon de la loi de Cauchy
S=cumsum(X); // vecteur  des sommes partielles
Y=S./N; //vecteur des S_k/k
xbasc();plot2d(N',Y',2); // ligne polygonale de sommets (k,S_k/k) en bleu
xsegs([0 n],[0 0],5); // tracé en rouge de l'axe des abscisses
```

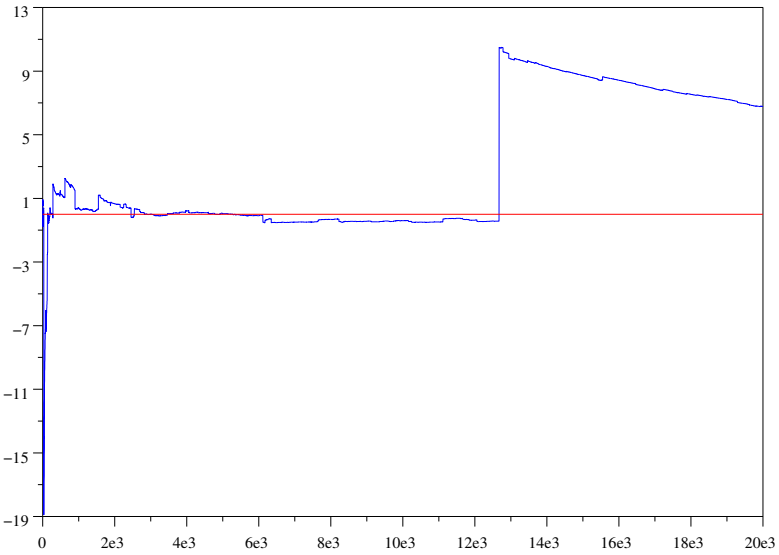

FIG. 8 – Ligne polygonale des S_k/k typique pour un échantillon de Cauchy ($n = 20\,000$)

3 Autour du TLC

Les X_k étant des variables aléatoires réelles définies sur le même espace probabilisé et indépendantes, on note encore $S_n := X_1 + \cdots + X_n$ et on étudie la convergence *en loi* de

$$S_n^* := \frac{S_n - \mathbf{E}S_n}{\sqrt{\operatorname{Var} S_n}},$$

lorsque cette quantité est définie. On envisagera aussi le cas où les X_k sont des vecteurs aléatoires [4] de $\mathbb{R}^d$.

Définition 4. *Soit $(Y_n)_{n\geq 1}$ une suite de vecteurs aléatoires de $\mathbb{R}^d$. On dit qu'elle converge en loi vers le vecteur aléatoire Y de $\mathbb{R}^d$ si pour toute fonction continue bornée $h : \mathbb{R}^d \to \mathbb{R}$, la suite de réels $\mathbf{E}h(Y_n)$ converge vers $\mathbf{E}h(Y)$.*

Remarque 5. Il n'y a pas unicité de la limite en loi. Si Z est un vecteur aléatoire de même loi que Y alors $\mathbf{E}h(Y) = \mathbf{E}h(Z)$ et Y_n converge aussi en loi vers Z. La convergence en loi n'est pas une vraie convergence de suites de vecteurs aléatoires, c'est une convergence de leurs lois, c'est-à-dire des mesures de probabilité $P_{Y_n} = \mathbf{P} \circ Y_n^{-1}$ sur $\mathbb{R}^d$. Puisque la limite en loi de Y_n peut être n'importe quel vecteur aléatoire ayant même loi que Y, ceci légitime l'abus de langage utilisé dans l'enoncé du théorème 10 où l'on dit que S_n^* converge en loi vers la loi gaussienne $\mathfrak{N}(0,1)$.

Remarque 6. Il convient d'être particulièrement prudent avec les opérations sur la convergence en loi. Par exemple *il n'est pas vrai* que si X_n converge en loi vers X et Y_n converge en loi vers Y, alors $X_n + Y_n$ converge en loi vers $X + Y$. Pour le voir, il suffit de noter que si on remplace X par X' de même loi que X et Y par Y' de même loi que Y, $X' + Y'$ n'a pas forcément même loi que $X + Y$.

Remarque 7. Un des avantages de la définition 4 est de rendre immédiate la propriété suivante fort utile en statistique. Si Y_n converge en loi vers Y, alors pour toute fonction continue g définie sur $\mathbb{R}^d$, à valeurs dans $\mathbb{R}$ ou $\mathbb{R}^k$, $g(Y_n)$ converge en loi vers $g(Y)$.

Remarque 8. L'inconvénient de la définition 4 est de masquer la signification intuitive de la convergence en loi qui est la convergence des $\mathbf{P}(Y_n \in B)$ vers $\mathbf{P}(Y \in B)$ pour tout « bon ensemble » B. De manière moins intuitive, mais rigoureuse, « bon ensemble » signifie précisément ensemble borélien de $\mathbb{R}^d$ dont la frontière est de mesure nulle pour P_Y. Si $d = 1$ et B est un intervalle $]-\infty, x]$, sa frontière se réduit au singleton $\{x\}$. Elle est de mesure nulle pour P_Y si et seulement si la fonction de répartition F de Y n'a pas de saut au point x. Ainsi pour $d = 1$, en notant F_n la f.d.r. de Y_n, la convergence en loi de Y_n vers Y implique la convergence de $F_n(x)$ vers $F(x)$ en tout point x où F est continue. La réciproque est d'ailleurs vraie [5].

4. La définition de la convergence en loi est donnée ci-après directement en dimension d pour des raisons d'économie. C'est évidemment anti-pédagogique.

5. Ceci permet d'énoncer la conclusion du théorème limite central sous la forme élémentaire : $\forall x \in \mathbb{R}$, $\mathbf{P}(S_n^* \leq x) \to \Phi(x)$, où Φ est la f.d.r. de la loi $\mathfrak{N}(0,1)$, évitant ainsi complètement de parler de convergence en loi.

Définition 9. *Soit $m \in \mathbb{R}$ et $\sigma \in]0,+\infty[$. On appelle densité gaussienne ou normale $f_{m,\sigma}$ sur $\mathbb{R}$ la fonction :*

$$f_{m,\sigma} : \mathbb{R} \longrightarrow \mathbb{R}^+, \quad t \longmapsto \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(t-m)^2}{2\sigma^2}\right).$$

La loi de densité $f_{m,\sigma}$ est appelée loi gaussienne de paramètres m et σ et notée $\mathfrak{N}(m,\sigma)$. Dans le cas particulier $\sigma = 0$, on appelle encore gaussienne la loi de la v.a. constante égale à m (il n'y a évidemment pas de densité dans ce cas).

Théorème 10. *Soit $(X_i)_{i\geq 1}$ une suite de variables aléatoires indépendantes, de même loi et de carré intégrable (et non constantes). Notons $\mu = \mathbf{E}X_1$, $\sigma^2 := \operatorname{Var} X_1$ avec $\sigma > 0$. Alors*

$$S_n^* := \frac{S_n - \mathbf{E}S_n}{\sqrt{\operatorname{Var} S_n}} = \frac{S_n - n\mu}{\sigma\sqrt{n}} \xrightarrow[n\to+\infty]{\text{loi}} \mathfrak{N}(0,1).$$

La vitesse de convergence dans le théorème 10 est en $O(n^{-1/2})$ dans les bons cas. On la mesure par le supremum sur $\mathbb{R}$ de $|\mathbf{P}(S_n^* \leq x) - \Phi(x)|$, où Φ est la f.d.r. de $\mathfrak{N}(0,1)$.

Théorème 11 (Berry-Esséen, 1941–42, Katz 1963, Petrov 1965)**.** *Soit $(X_i)_{i\geq 1}$ une suite de variables aléatoires i.i.d. telle que $\mathbf{E}X_1 = 0$, $\mathbf{E}|X_i|^3 < +\infty$. On note $\sigma^2 := \mathbf{E}X_1^2$ ($\sigma > 0$). Il existe alors une constante universelle $C > 0$ telle que pour tout $n \geq 1$,*

$$\Delta_n := \sup_{x\in\mathbb{R}} \left|\mathbf{P}\Big(\frac{S_n}{\sigma\sqrt{n}} \leq x\Big) - \Phi(x)\right| \leq C\frac{\mathbf{E}|X_1|^3}{\sigma^3}\frac{1}{\sqrt{n}}.$$

Si $\mathbf{E}|X_1|^{2+\delta} < +\infty$ pour un $\delta \in]0,1]$, on a avec une constante universelle $A > 0$,

$$\forall n \geq 1, \quad \Delta_n \leq A\frac{\mathbf{E}|X_1|^{2+\delta}}{\sigma^{2+\delta}}\frac{1}{n^{\delta/2}}.$$

La meilleure valeur de C connue à jour est $C = 0{,}797\,5$ (Van Beek, 1972).

On dit que le vecteur aléatoire $X = (X_1, \ldots, X_d)$ dans $\mathbb{R}^d$ est gaussien si toute combinaison linéaire de ses composantes est une variable aléatoire réelle gaussienne. La loi de X est alors caractérisée par le vecteur des espérances $(\mathbf{E}X_1, \ldots, \mathbf{E}X_d)$ et la matrice de covariance K de terme général $K_{i,j} = \operatorname{Cov}(X_i, X_j)$. Si cette matrice est définie positive, la loi de X admet une densité.

Théorème 12. *Soit (X_k) une suite de vecteurs aléatoires de $\mathbb{R}^d$, indépendants, de même loi et de carré intégrable (i.e. $\mathbf{E}\|X_1\|^2 < +\infty$) et $S_n := \sum_{k=1}^n X_k$. Alors*

$$\frac{S_n - \mathbf{E}S_n}{\sqrt{n}} \xrightarrow[n\to+\infty]{\text{loi}} \mathfrak{N}(0,K), \tag{6}$$

où K est la matrice de covariance de X_1.

Malgré la ressemblance formelle de cet énoncé avec celui du théorème 10, il n'est pas possible ici de modifier la normalisation $\sqrt{n}$ par $\sigma\sqrt{n}$ pour avoir toujours la même loi limite $\mathfrak{N}(0,I)$, où I est la matrice identité sur $\mathbb{R}^d$.

3.1 Approximation de la loi binomiale par une loi de Poisson

Avant de regarder l'application du TLC à l'approximation d'une loi binomiale par une gaussienne, rappelons l'approximation par une loi de Poisson.

Théorème 13. *Si $(p_n)_{n\geq 1}$ est une suite de réels de $[0,1]$ vérifiant*

$$np_n \to \lambda \in]0,+\infty[, \quad \textit{quand} \quad n \to +\infty, \tag{7}$$

alors :

$$\forall k \in \mathbb{N}, \quad C_n^k p_n^k (1-p_n)^{n-k} \longrightarrow \mathrm{e}^{-\lambda}\frac{\lambda^k}{k!}, \quad \textit{quand} \quad n \to +\infty.$$

On illustre cette approximation par la comparaison des diagrammes en bâtons de la loi $\mathrm{Bin}(n,p)$ et de la loi $\mathrm{Pois}(\lambda)$ avec $\lambda = np$. On ne peut évidemment pas afficher tous les bâtons de la loi de Poisson (il y en a une infinité). On se limite aux valeurs de k entre 0 et $2\lambda+4$. Ce choix n'est pas complètement arbitraire, il repose sur l'inégalité suivante. Si S_n suit la loi $\mathrm{Bin}(n,p)$ et Y_n la loi $\mathrm{Pois}(\lambda)$ avec $\lambda = np$, on a pour tout $k \geq 2\lambda+1$, $\mathbf{P}(S_n > k) \leq \mathbf{P}(Y_n = k)$, cf. exercice A.4 dans [13].

```
// Comparaison graphique de la loi Bin(n,p) avec Pois(np)
// On affiche les probas poissonnienes pour k=0 jusqu'a 2*lambda+4.
n=input('Rentrer le parametre n de la loi binomiale : ');
p=input('Rentrer le parametre p de la loi binomiale : ');
lambda=n.*p;
write(%io(2),'approximation par la loi Pois('+string(lambda)+')')
m=ceil(2.*lambda+4);
z=cdfpoi("PQ",0:m,lambda.*ones(0:m));
pois=[z(1), z(2:(m+1))-z(1:m)];
pp=p.*ones(0:m);qq=1-pp;
y=cdfbin("PQ",0:m,n.*ones(0:m),pp,qq);
bin=[y(1), y(2:(m+1))-y(1:m)];
xpois=(0:m)-.04; // decalage des abscissses pour affichage
xbin=(0:m)+.04;
xbasc();
plot2d3("gnn",[xpois; xbin]',[pois; bin]',[11 5],"011"," ",..
    [-1,0,m+1,1.25.*max(bin)],[0,m+2,0,5])
// les .. dans plot2d3 permettent de passer à la ligne pour une longue
// instruction.
// Le 4eme argument fixe les couleurs: pois bleu (11), bin rouge (5).
// Le 7eme argument  fixe les dimensions du cadre de facon adaptative.
// Le 8eme controle les graduations. Ainsi l'avant derniere
//  graduation verticale (en montant) donne le maximum des
//  probas binomiales et la hauteur du cadre est de 1.25 fois ce max.
```

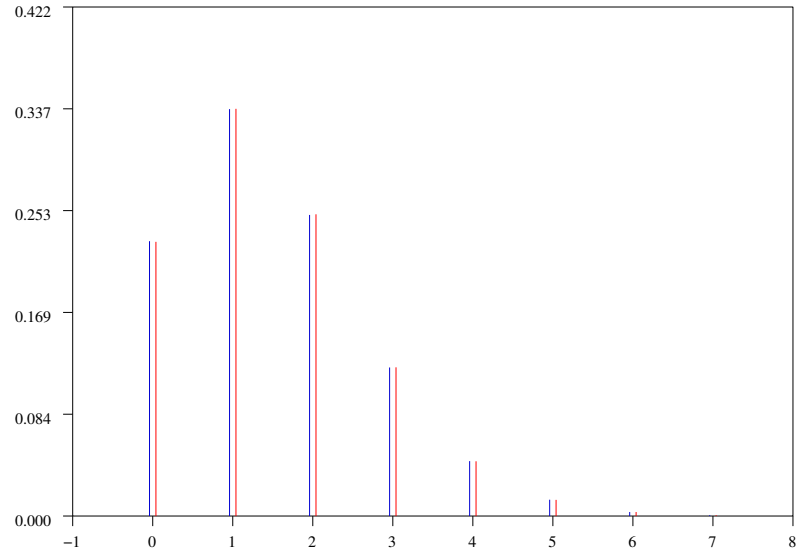

FIG. 9 – Approximation de la loi Bin(400; 0,003 7) par la loi Pois(1,48)

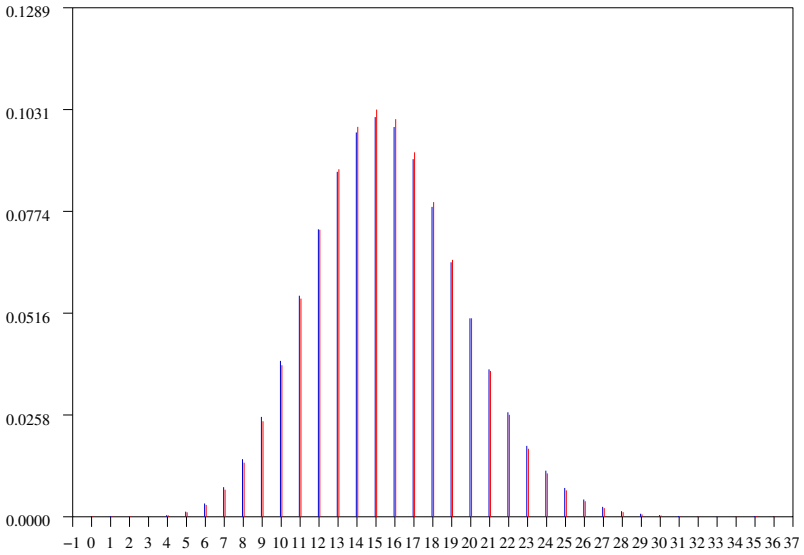

FIG. 10 – Approximation de la loi Bin(400; 0,039) par la loi Pois(15,6)

3.2 Approximation gaussienne de la binomiale

Théorème 14 (De Moivre-Laplace). *Si les X_k sont indépendantes et de même loi de Bernoulli de paramètre $p \in]0,1[$, avec $q := 1-p$, on a*

$$S_n^* := \frac{S_n - np}{\sqrt{npq}} = \sqrt{\frac{n}{pq}}\left(\frac{S_n}{n} - p\right) \xrightarrow[n\to+\infty]{loi} \mathfrak{N}(0,1).$$

Comme la fonction de répartition Φ de $\mathfrak{N}(0,1)$ est continue sur $\mathbb{R}$, ceci équivaut à

$$\forall x \in \mathbb{R}, \quad \mathbf{P}(S_n^* \leq x) \xrightarrow[n\to+\infty]{} \Phi(x) = \int_{-\infty}^{x} \exp\left(-\frac{t^2}{2}\right)\frac{\mathrm{d}t}{\sqrt{2\pi}}.$$

Le théorème de de Moivre-Laplace est le premier exemple historique de théorème limite central. Il est possible d'en donner une démonstration élémentaire[6] basée essentiellement sur des estimations fines des coefficients binomiaux à l'aide de la formule de Stirling (voir par exemple [15] ou l'atelier [7]).

En fait on peut préciser la convergence de la loi binomiale vers la gaussienne par un théorème de *limite locale*[7] qui nous donne ici en notant S_n une variable de loi $\mathrm{Bin}(n,p)$:

$$\mathbf{P}(S_n = k) = \frac{1}{\sqrt{2\pi npq}} \exp\left(\frac{-(k-np)^2}{2npq}\right) + \frac{\varepsilon_{n,k}}{\sqrt{n}},$$

avec

$$\max_{0\leq k\leq n} |\varepsilon_{n,k}| \xrightarrow[n\to+\infty]{} 0.$$

Pour illustrer ce résultat, on trace le diagramme en bâtons de la loi binomiale et le graphe de la densité gaussienne approximante. La figure 11 a été obtenue en tapant

```
-->limlocbin(200,0.34)
```

La fonction `limloc` est définie ci-dessous comme un élément d'une « bibliothèque » de fonctions `Binomiales.sci` (on n'a gardé ici que les trois indispensables) que l'on charge une fois en mémoire avec un `getf`.

```
function [y]=coefbinA(n)
// "A" comme additive, fournit le vecteur ligne des C_n^k, k=0:n
A=[1]; // initialisation n=0
for i=1:n, A=[A 0]+[0 A];end
y=A
endfunction
//
// Limitations: on peut aller jusqu'a n=1029. Pour le voir, taper :
// y=coefbinA(1029);max(y)
```

6. Élémentaire n'est pas synomyme de facile. Dans ce contexte, cela signifie que les techniques mathématiques utilisées relèvent du cours d'analyse d'une deuxième année de DEUG.

7. Une référence accessible pour ce théorème est Revuz [10, IV 5,5].

```
// puis
// y=coefbinA(1030);max(y)

function [y]=loibin(n,p)
// n entier, p probabilité
// retourne le vecteur ligne des masses ponctuelles de la loi Bin(n,p)
// p(k+1)=C_n^k p^k q^(n-k)
//
C=coefbinA(n);
q=1-p;
y=C.*p.^(0:n).*q.^(n:-1:0);
endfunction

function []=limlocbin(n,p)
// affiche le diagramme en bâtons de loibin(n,p) en bleu
// et la densité gaussienne approximante en rouge
x=0:n;y=loibin(n,p);
xx=0:0.1:n; moy=n.*p; var=n.*p.*(1-p);
yy=(2.*%pi.*var)^(-0.5).*exp(-(xx-moy).^2./(2.*var));
xbasc(); plot2d3(x',y',2); plot2d(xx',yy',5)
endfunction
```

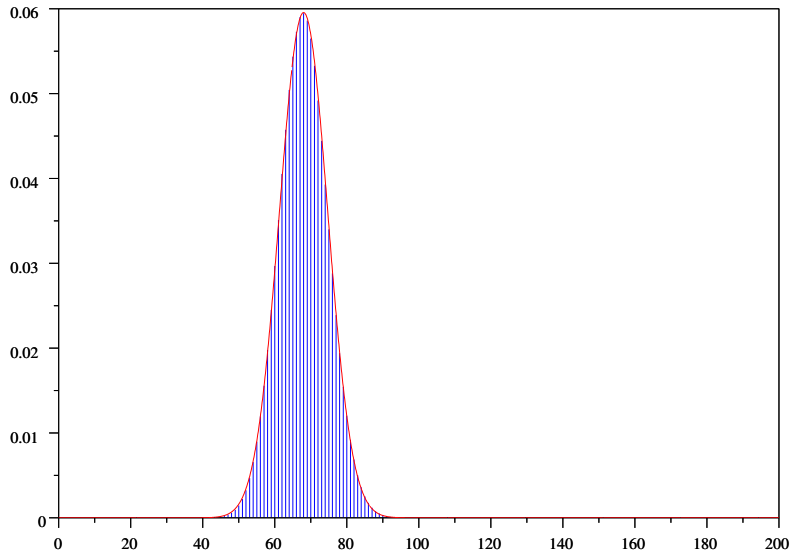

FIG. 11 – Approximation de la loi Bin(200; 0,34) par la loi $\mathfrak{N}(68;\ 6{,}699)$

3.3 Intervalles de confiance

Une application directe du théorème de de Moivre-Laplace est la construction d'*intervalles de confiance* pour l'estimation d'une probabilité inconnue p à partir de l'observation d'un *échantillon* de n variables de Bernoulli indépendantes de paramètre p. Considérons pour $t > 0$ l'événement

$$A_{n,t} := \Big\{\omega \in \Omega;\ -t \leq \sqrt{\frac{n}{pq}}\Big(\frac{S_n(\omega)}{n} - p\Big) \leq t\Big\}.$$

Le théorème de de Moivre-Laplace nous dit que pour n assez grand, on peut utiliser l'approximation :

$$\mathbf{P}\big(A_{n,t}\big) \simeq \Phi(t) - \Phi(-t) = 2\Phi(t) - 1.$$

Ceci peut se réécrire

$$\mathbf{P}\bigg(\frac{S_n}{n} - t\sqrt{\frac{pq}{n}} \leq p \leq \frac{S_n}{n} + t\sqrt{\frac{pq}{n}}\bigg) = 2\Phi(t) - 1 + \varepsilon_n.$$

On ignore la valeur de p, donc a fortiori celle de $\sqrt{pq}$. Heureusement, il est possible de la majorer car $p(1-p)$ est maximal pour $p = 1/2$. D'où

$$\sqrt{pq} \leq \sqrt{\frac{1}{4}} = \frac{1}{2}, \tag{8}$$

de sorte qu'en notant

$$B_{n,t} := \Big\{\omega \in \Omega;\ \frac{S_n(\omega)}{n} - \frac{t}{2\sqrt{n}} \leq p \leq \frac{S_n(\omega)}{n} + \frac{t}{2\sqrt{n}}\Big\},$$

l'inclusion $A_{n,t} \subset B_{n,t}$ nous donne :

$$\mathbf{P}\big(B_{n,t}\big) \geq 2\Phi(t) - 1 + \varepsilon_n. \tag{9}$$

En pratique, n est fixé et on a observé des valeurs numériques explicites $x_1, \ldots, x_n$ que l'on interprète comme les valeurs de $X_1(\omega), \ldots, X_n(\omega)$ pour *un même* ω tiré au sort (suivant $\mathbf{P}$). On est donc en présence d'une valeur numérique explicite, $S_n(\omega)/n = (x_1 + \cdots + x_n)/n$, disons pour fixer les idées $S_n(\omega)/n = 0{,}53$. Proposer pour le paramètre inconnu p l'intervalle de confiance

$$I_{n,t} = \bigg[0{,}53 - \frac{t}{2\sqrt{n}};\ 0{,}53 + \frac{t}{2\sqrt{n}}\bigg],$$

c'est faire le *pari* que le ω observé est bien dans $B_{n,t}$. La probabilité de gagner ce pari est minorée par $2\Phi(t) - 1 + \varepsilon_n$. On dit que $I_{n,t}$ est un intervalle de confiance pour p avec un *niveau* d'au moins $2\Phi(t) - 1 + \varepsilon_n$. En pratique, on laisse tomber le ε_n et on détermine t de façon approchée grâce à la tabulation de Φ. Par exemple pour un niveau de confiance

de 95%, on est ramené à la résolution de l'équation $\Phi(t) = 1{,}95/2 = 0{,}975$ d'où $t \simeq 1{,}96$, ce qui nous donne l'intervalle

$$I_n = \left[\frac{S_n(\omega)}{n} - \frac{1{,}96}{2\sqrt{n}};\ \frac{S_n(\omega)}{n} + \frac{1{,}96}{2\sqrt{n}}\right], \quad \text{au niveau de confiance 95\%.}$$

En fait les statisticiens préfèrent une variante de cette méthode pour obtenir des intervalles de confiance plus étroits, notamment quand p n'est pas trop proche de $1/2$. L'idée est de remplacer la variance inconnue pq de X_1 par un *estimateur*, au lieu de la majorer de façon certaine par (8). Ainsi en estimant pq par $M_n(1-M_n)$ où $M_n := S_n/n$, on obtient au niveau de confiance 95% l'intervalle

$$J_n = \left[M_n(\omega) - 1{,}96\sqrt{\frac{M_n(\omega)(1-M_n(\omega))}{n}};\ M_n(\omega) + 1{,}96\sqrt{\frac{M_n(\omega)(1-M_n(\omega))}{n}}\right].$$

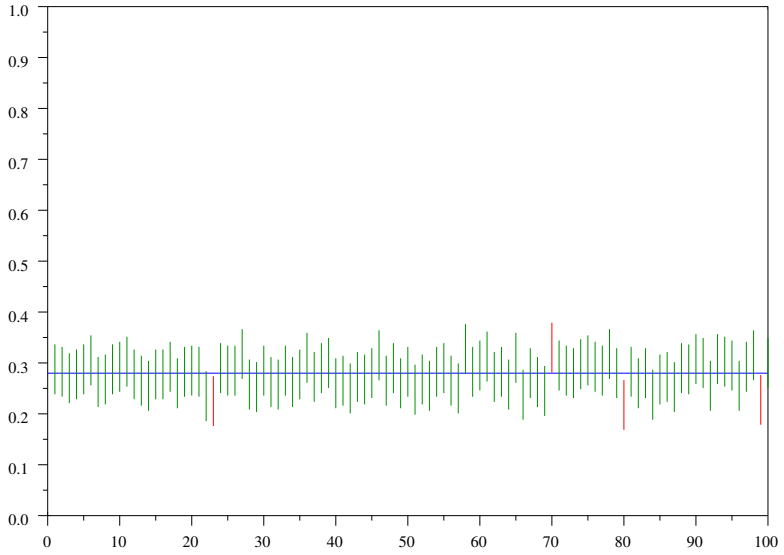

FIG. 12 – Peigne de 100 intervalles de confiance ($n = 400$, $p = 0{,}28$)

La figure 12 représente 100 intervalles de confiance du type I, *i.e.* avec majoration de $p(1-p)$ par $1/4$. Pour l'obtenir on a simulé 100 échantillons de taille 400 pour chacun desquels on a calculé les bornes de l'intervalle de confiance correspondant. Les intervalles contenant effectivement la vraie valeur de p sont représentés verticalement en vert, les autres (ici il y en a 4) en rouge. Cette figure a été obtenue avec le script `Peigne.sce` qui outre sa sortie graphique, retourne les numéros d'échantillons pour lesquels $p \notin I$. Voici une copie de l'écran de l'interpréteur de commandes :

```
-->;exec("/home/suquet/Enseignement/Scilab/CLT/Peigne.sce");
Taille d'echantillon ?-->400
Rentrer la probabilité p à estimer:-->0.28

 p hors intervalle de confiance pour les échantillons No :
   23
   70
   80
   99
```

Le code du script `Peigne.sce` est le suivant.

```
// Le peigne des intervalles de confiance pour
// l'estimation d'une probabilité inconnue
//
// Saisie des parametres
n=input('Taille d''echantillon ?');
p=input('Rentrer la probabilité p à estimer:');
Mn=zeros(1:100); // initialisation des moyennes arithmétiques
a=zeros(1:100);b=zeros(1:100);couleur=ones(1:100);
xbasc();
plot2d([0 100],[0 1],[-1 -1],"021");// astuce pour fixer le cadre
xsegs([0 100],[p p],[2]); // horizontale p
for i=1:100,
    X=bool2s(rand(1:n,"uniform")<p);Mn(i)=sum(X)./n;
    a(i)=Mn(i)-1.96.*(4.*n).^(-0.5);
    b(i)=Mn(i)+1.96.*(4.*n).^(-0.5);
    if (p<a(i))|(p>b(i)) then
         couleur(i)=5; // rouge si p hors intervalle
    else couleur(i)=13; //  vert si p dans intervalle
    end,
    xsegs([i i],[a(i) b(i)],couleur(i))
end
rep=find(couleur == 5); // numeros des intervalles rouges
if rep == [] then
  printf('%s\n', 'Tous les intervalles de confiance contiennent p');
else
  printf('\n%s\n',...
         ' p hors intervalle de confiance pour les échantillons No : ');
  printf('%5i\n',rep');
end
```

La figure 13 obtenue avec le script `Peigne2.sce` compare le peigne des intervalles de type I avec celui des intervalles du type J (variance estimée). Les intervalles du type J

sont plus courts. Il n'est donc pas étonnant d'en trouver plus qui « manquent » p (ici 8 contre 4).

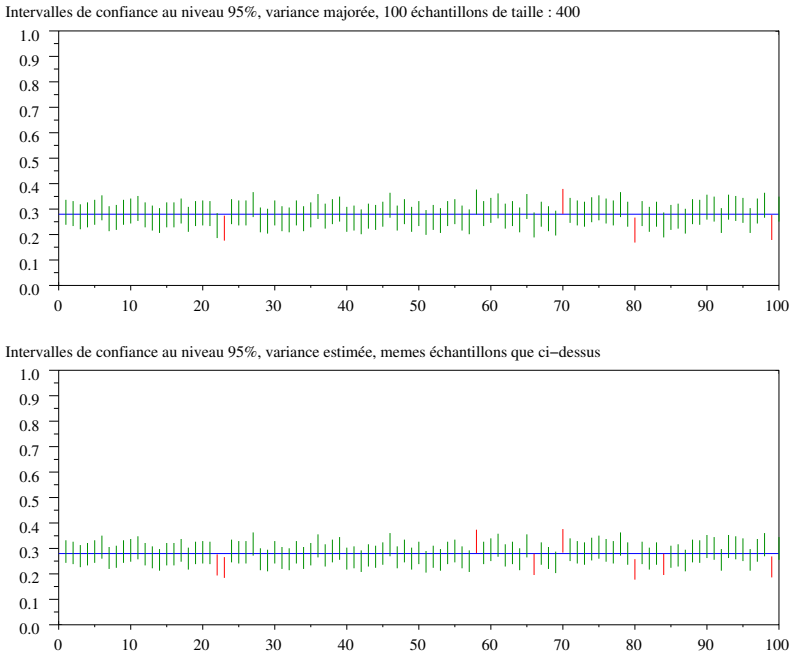

FIG. 13 – Peignes de 100 intervalles de confiance I et J ($n = 400$, $p = 0{,}28$)

Voici le code du script `Peigne2.sce`. Noter l'utilisation de `xsetech` pour gérer le partage de la fenêtre graphique.

```
// Le peigne des intervalles de confiance pour
// l'estimation d'une probabilité inconnue (niveau 95%)
// On affiche les deux types d'intervalles de confiance:
// avec variance majorée par 1/4 et avec varaince estimée
// Saisie des parametres
n=input('Taille d''echantillon ?');
p=input('Rentrer la probabilité p à estimer:');
Mn=zeros(1:100); // initialisation des moyennes arithmétiques
a=zeros(1:100);b=zeros(1:100);
A=zeros(1:100);B=zeros(1:100);;
couleur=ones(2,100);
for i=1:100,
    X=bool2s(rand(1:n,"uniform")<p);Mn(i)=sum(X)./n;
    a(i)=Mn(i)-1.96.*(4.*n).^(-0.5);
    b(i)=Mn(i)+1.96.*(4.*n).^(-0.5);
    A(i)=Mn(i)-1.96.*sqrt(Mn(i).*(1-Mn(i))./n);
    B(i)=Mn(i)+1.96.*sqrt(Mn(i).*(1-Mn(i))./n);
```

```
    if (p<a(i))|(p>b(i)) then
         couleur(1,i)=5; // rouge si p hors intervalle
    else couleur(1,i)=13; //  vert si p dans intervalle
    end,
    if (p<A(i))|(p>B(i)) then
         couleur(2,i)=5; // rouge si p hors intervalle
    else couleur(2,i)=13; //  vert si p dans intervalle
    end,
end
xbasc();
xsetech([0,0,1,0.5]);
plot2d([0 100],[0 1],[-1 -1],"021");// astuce pour fixer le cadre
titregauche="Intervalles de confiance au niveau 95%, variance majorée, ";
titredroit="100 échantillons de taille : "+string(n);
xtitle(titregauche + titredroit);
xsegs([0 100],[p p],[2]); // horizontale p
xsegs([1:100; 1:100],[a; b],couleur(1,:));
//
xsetech([0,0.5,1,0.5]);
plot2d([0 100],[0 1],[-1 -1],"021");
Titregauche="Intervalles de confiance au niveau 95%, variance estimée, ";
Titredroit="memes échantillons que ci-dessus";
xtitle(Titregauche + Titredroit);
xsegs([0 100],[p p],[2]);
xsegs([1:100; 1:100],[A; B],couleur(2,:));
```

3.4 Lois de Pareto et vitesse de convergence dans le TLC

Pour une illustration expérimentale de la vitesse de convergence dans le TLC (voir théorème 11), on propose l'exercice de T.P. suivant.

On dit que X suit la loi de Paréto de paramètres p et 1, si

$$\mathbf{P}(X > x) = \begin{cases} 1 & si\ x \leq 1, \\ x^{-p} & si\ x > 1. \end{cases}$$

L'espérance existe pour $p > 1$, la variance pour $p > 2$. Elles valent respectivement :

$$\mathbf{E}X = \frac{p}{p-1}, \qquad \operatorname{Var} X = \frac{p}{(p-1)^2(p-2)}.$$

1. *Comment générer un échantillon de la loi de Paréto à partir d'un échantillon uniforme ?*
2. *Étudier expérimentalement l'influence du paramètre p sur la valeur maximale de l'échantillon.*
3. *Générer N échantillons de taille n de la loi de Paréto de paramètre p (données à faire saisir interactivement) et calculer pour chacun la somme centrée réduite du TLC (S_n^*). Faire tracer l'histogramme de ces N sommes centrées réduites (classes de longueur $1/2$ entre -4 et $+4$) ainsi que la courbe de la densité gaussienne standard. Étudier l'influence du paramètre p sur la qualité de l'approximation gaussienne.*

L'intérêt de la loi de Paréto dans ce contexte est que $\mathbf{E}|X|^r$ est fini pour tout réel r tel que $0 \leq r < p$ et infini pour tout $r \geq p$. En guise de solution, voici le code du script `ParetoCLT.sce` qui a servi à générer les figures 14 et 15 (noter la différence d'échelle verticale entre les deux figures).

```
// Le TLC pour des lois de Pareto
//
// Saisie des parametres
n=input('Taille d''echantillon ?');
N=input('Nombre d''echantillons?');
p=input('Rentrer le parametre p de la loi Par(p,1):');
EX=p./(p-1);VarX=p./((p-1).^2.*(p-2));
// generation des echantillons par  X=U.^{-1/p}
// et des sommes normalisees TLC
S=zeros(1:N)';// une seule allocation memoire pour S
for i=1:N,
    X=rand(1:n,"uniform").^(-1./p); // on ne memorise pas X(i)!
    S(i)=(sum(X)-n.*EX).*(n.*VarX).^(-0.5);
end
xbasc();histplot(-4:.5:4,S,[2]); // histogramme en bleu
t=linspace(-4,4,201)';f=(2.*%pi).^(-.5).*exp(-0.5.*t.^2);
plot2d(t,f,[21],"000") // courbe gaussienne standard en rouge
```

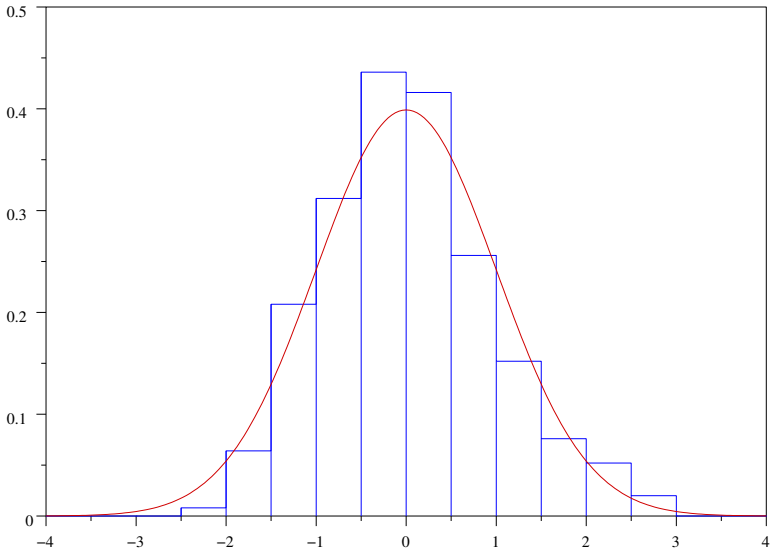

FIG. 14 – Histogramme des valeurs de S^*_{1000} pour $N = 500$ et $p = 3$

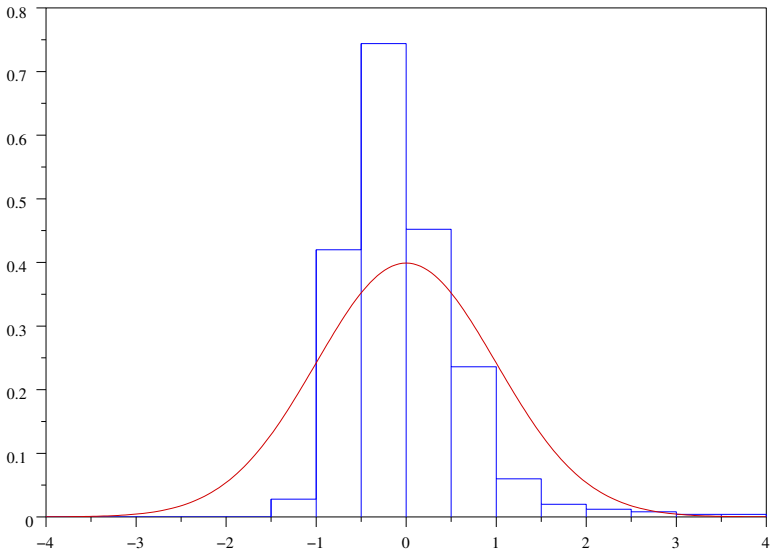

FIG. 15 – Histogramme des valeurs de S^*_{1000} pour $N = 500$ et $p = 2{,}1$)

3.5 Loi uniforme sur un triangle et TLC dans $\mathbb{R}^2$

Voyons maintenant une illustration du théorème limite central dans $\mathbb{R}^2$. Pour disposer d'une suite de vecteurs aléatoires X_n indépendants et de même loi, on commence par simuler un n-échantillon de vecteurs aléatoires de loi uniforme sur un triangle (voir [16] pour l'algorithme).

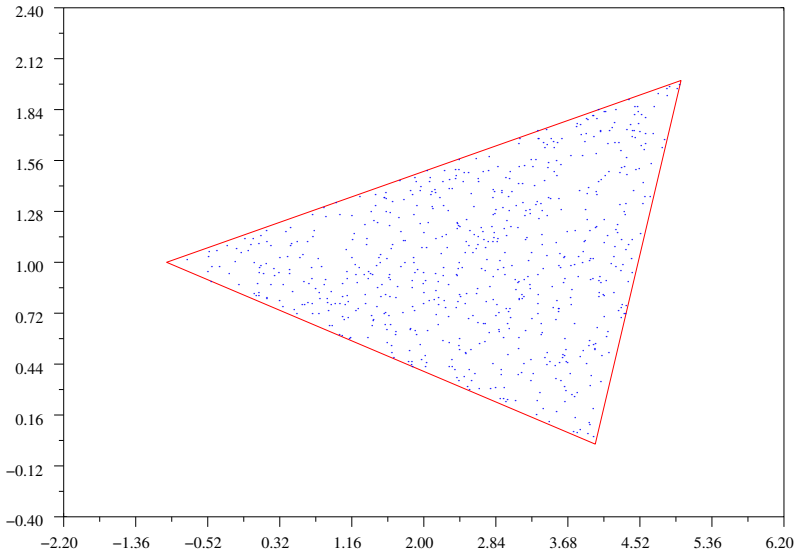

FIG. 16 – Échantillon de taille 700 de la loi uniforme sur le triangle

La figure 16 a été obtenue en lançant comme suit le script `Triangunif.sce` qui commence par une saisie interactive des coordonnées des sommets du triangle vues comme les colonnes de la matrice `M`.

```
-->;exec("/home/suquet/Enseignement/Scilab/CLT/Triangunif.sce");
Coordonnées des sommets du triangle (matrice 2x3)?-->[4 5 -1; 0 2 1]
 M  =

!   4.    5.   - 1. !
!   0.    2.     1. !
Taille d'echantillon ?-->700
```

Pour cet échantillon on calcule

$$S_n^* = \frac{S_n - \mathbf{E}S_n}{\sqrt{n}}$$

qui est donc un vecteur aléatoire de $\mathbb{R}^2$ dont la loi est approximativement $\mathfrak{N}(0, K)$, où K est la matrice de covariance de la loi uniforme sur le triangle. Pour visualiser ceci,

on répète l'opération un grand nombre N de fois, obtenant un nuage de points $S^*_{n,i}$, $i = 1, \ldots, N$, qui « ressemble » à celui d'un N-échantillon de la loi $\mathfrak{N}(0, K)$, dont on trace les lignes de niveau de la densité. Ces lignes de niveau sont des ellipses centrées en 0. La figure 17 qui illustre tout ceci a été obtenue en lançant le script `TriUnCLT.sce` comme suit [8].

```
-->;getf("/home/suquet/Enseignement/Scilab/CLT/fq2d.sci");

-->;exec("/home/suquet/Enseignement/Scilab/CLT/TriUnCLT.sce");
Coordonnées des sommets du triangle (matrice 2x3)?-->[3 5 6; 0 2 1]
 M  =

!   3.    5.    6. !
!   0.    2.    1. !
Taille d'échantillon ?-->700
nombre d'échantillons-->600
```

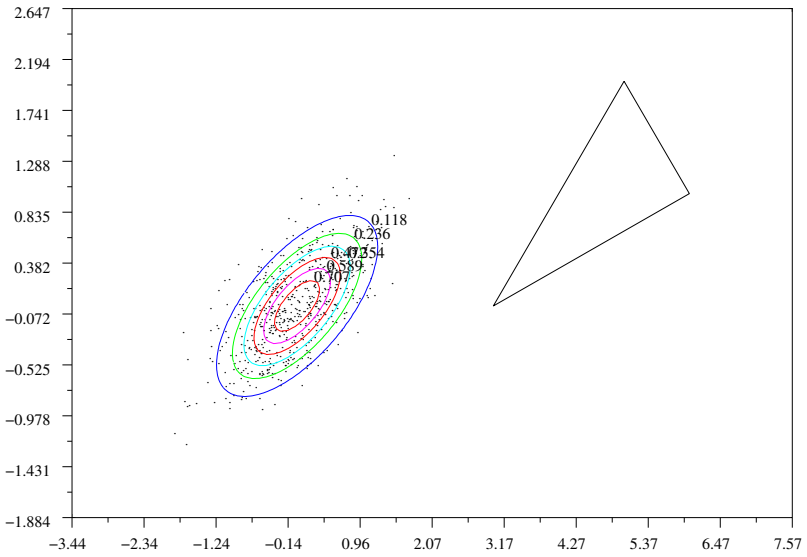

FIG. 17 – Nuage de point approximativement gaussien des $S^*_{700,i}$, $i = 1, \ldots, 600$

Il est recommandé d'essayer plusieurs formes de triangles pour voir comment évolue la forme des ellipses lignes de niveau gaussiennes associées.

Voici maintenant le code Scilab des scripts utilisés.

8. La fonction `fqd2.sci` est utilisée par le script. Il faut donc la charger une fois avant la première exécution de `TriUnCLT.sce`.

```
// Script Triangunif.sce
// Dessine un triangle, génère un échantillon de la loi uniforme
// sur ce triangle et affiche le nuage de points.
// Chaque point du nuage est une combinaison convexe des sommets avec
// pour coefficients les espacements d'un 2-échantillon uniforme sur [0,1]
//
M=input('Coordonnées des sommets du triangle (matrice 2x3)?');
M
n=input('Taille d''echantillon ?');
xsom=M(1,:);ysom=M(2,:);
U=rand(2,n); // n colonnes de 2-échantillons de U[0,1]
so=gsort(U,'r','i');// tri ascendant par colonne pour statistiques d'ordre
esp=[so; ones(1:n)]-[zeros(1:n); so];// n colonnes d'espacements
X=xsom*esp; //abscisses du nuage
Y=ysom*esp; //ordonnées du nuage
//
// Affichages
//
xbasc();
xmin=min(xsom)-0.2*(max(xsom)-min(xsom));
xmax=max(xsom)+0.2*(max(xsom)-min(xsom));
ymin=min(ysom)-0.2*(max(ysom)-min(ysom));
ymax=max(ysom)+0.2*(max(ysom)-min(ysom));
plot2d([xmin xmax],[ymin ymax],[-1 -1],"021");// cadre vide
// L'utilisation de la couleur doit être gérée par le contexte graphique
// à l'extérieur de 'xpoly'
xset("use color",1);
xset('color',5);
xpoly(xsom,ysom,"lines",1);// tracé du triangle en rouge
xset('color',2);
xpoly(X,Y,'marks');// tracé du nuage de points en bleu
xset("default")
// fin du script

function [z]=fq2d(x,y,R)
// Calcule les valeurs  de la forme quadratique de matrice R (2x2)
//  aux points d'abscisse x_i et d'ordonnée y_j, x et y  vecteurs.
//
m=length(x);n=length(y);
z=zeros(m,n);
for i=1:m,
  for j=1:n,
    z(i,j)=[x(i), y(j)]*R*[x(i);y(j)];
  end
```

```
end
endfunction

// Script TriUnCLT.sce
// TLC pour des vecteurs uniformes sur un triangle.
// Charger au préalable la fonction fq2d.sci
//
// 1)Dessine un triangle, génère N échantillons de taille n de la
//   loi uniforme sur ce triangle.
//
// 2)Affiche le nuage de points des N sommes normalisées des échantillons.
//
// 3)Affiche les lignes de niveau de la densité gaussienne limite
//
// suggestion d'essai M=4.*rand(2,3), n=500, N=800
M=input('Coordonnées des sommets du triangle (matrice 2x3)?')
n=input('Taille d''échantillon ?');
N=input('nombre d''échantillons');
xsom=M(1,:);ysom=M(2,:);
SX=zeros(1,N);SY=zeros(1,N);
for i=1:N
U=rand(2,n); // n colonnes de 2-échantillons de U[0,1]
so=gsort(U,'r','i');// tri ascendant par colonne pour statistiques d'ordre
esp=[so; ones(1:n)]-[zeros(1:n); so];// n colonnes d'espacements
X=xsom*esp-sum(xsom)./3; //abscisses de l'échantillon centré
Y=ysom*esp-sum(ysom)./3; //ordonnées de l'échantillon centré
SX(i)=n.^{-0.5}.*sum(X); // somme normalisée de l'échantillon no i
SY(i)=n.^{-0.5}.*sum(Y);
end
//
// Affichages triangle et nuage
//
xbasc();
// On veut que tout tienne dans le cadre: triangle et nuage
a=min([SX xsom]);b=max([SX xsom]);
c=min([SY ysom]);d=max([SY ysom]);
xmin=a-0.2*(b-a);
xmax=b+0.2*(b-a);
ymin=c-0.2*(d-c);
ymax=d+0.2*(d-c);
plot2d([xmin xmax],[ymin ymax],[-1 -1],"021");// cadre vide
xpoly(xsom,ysom,"lines",1);// tracé du triangle
xpoly(SX,SY,'marks');// tracé du nuage de points
//
```

```
// Calcul de la covariance par P*cov(V)*P'
// où V est un vecteur de loi uniforme sur le triangle
// associe a la base orthonormee canonique
// et P la matrice de transformation de cette base en (AB,AC)
//
P=[M(:,2)-M(:,1), M(:,3)-M(:,1)];
CV=P*[1/18, -1/36; -1/36, 1/18]*P';
// Lignes de niveau de la densité gaussienne de la loi limite
x=linspace(xmin,xmax,101);y=linspace(ymin,ymax,101);
z=1/(2*%pi*sqrt(det(CV))).*exp(-0.5.*fq2d(x,y,inv(CV)));
contour2d(x,y,z,6,2:7,"000");
// fin du script
```

4 Sondages et deuxième tour

En guise de contribution au thème *Statistique et Citoyenneté*, terminons par une petite étude empirique de la difficulté de pronostiquer les noms des deux candidats restant au second tour d'une élection présidentielle lorsqu'il y a beaucoup de candidats. Regardons d'abord deux sorties du script `presidentielle1.sce` qui simule le sondage d'un échantillon de 500 personnes avant le premier tour d'un élection présidentielle où concourent 12 candidats désignés par une lettre A, B,..., L, dans l'ordre alphabétique correspondant à leur pourcentage de voix dans la population totale.

```
-->;exec("/home/suquet/Enseignement/Scilab/Divers/presidentielle1.sce");

Candidat % Pop  VoixEc  % Ech

A        19.00      97  19.40
B        17.00      87  17.40
C        16.00      87  17.40
D        10.00      50  10.00
E         8.00      40   8.00
F         7.00      32   6.40
G         6.00      32   6.40
H         5.50      27   5.40
I         4.00      18   3.60
J         3.00      10   2.00
K         2.50      12   2.40
L         2.00       8   1.60

Pronostic du sondeur pour l'ordre d'arrivée au 1er tour :

A B C D E F G H I K J L
```

```
-->;exec("/home/suquet/Enseignement/Scilab/Divers/presidentielle1.sce");

Candidat % Pop  VoixEc  % Ech

A        19.00      95  19.00
B        17.00      85  17.00
C        16.00      90  18.00
D        10.00      44   8.80
E         8.00      35   7.00
F         7.00      34   6.80
G         6.00      31   6.20
H         5.50      29   5.80
I         4.00      19   3.80
J         3.00      14   2.80
K         2.50       8   1.60
L         2.00      16   3.20

Pronostic du sondeur pour l'ordre d'arrivée au 1er tour :

A C B D E F G H I L J K
```

On voit sur la deuxième sortie une erreur de pronostic du deuxième tour qui prévoit un duel A-C au lieu de A-B. Quand on exécute une douzaine de fois le script, on s'apercoit que ce type d'erreur semble assez fréquent. Pour en avoir le coeur net, on réalise 1 000 simulations d'un tel sondage de premier tour et on regarde la fréquence d'erreur de pronostic du second tour (*i.e.* prévision de présence au second tour d'au moins un candidat autre que A ou B).

```
-->;exec("/home/suquet/Enseignement/Scilab/Divers/presidentielle2.sce");

Nombre de sondages simulés :  1000
Pourcentage d'erreurs de pronostic sur les finalistes :  37.50
```

Voici pour finir le code des deux scripts. Basiquement on simule un vecteur de loi multinomiale en partant d'un échantillon de taille 500 de la loi uniforme sur $[0,1]$, en découpant $[0,1]$ en 12 intervalles de longueurs égales aux proportions de voix dans la population totale des candidats et en comptant le nombre d'observations dans chaque intervalle. Du point de vue technique, ces scripts illustrent l'utilisation des fonction des tri, de formatage et d'affichage des nombres et des chaînes de caractères. Ils sont paramétrables via la modification de la taille n d'échantillon et du vecteur p des proportions de voix, la dénomination des candidats s'adaptant automatiquement à la longueur de p.

```
// Script presidentielle1.sce
// Sondage pour le 1er tour d'une élection présidentielle.
// Simulation d'un vecteur aléatoire de loi multinomiale de paramètres
```

```
//  n (nombre d'épreuves ou de sondés) et
//  p vecteur des probabilités de résultats élémentaires pour une épreuve,
//  ici proportions de voix de chaque candidat dans la population totale.
//  Affichage du pronostic premier tour en ligne avec adaptation
//  automatique au nombre de candidats donc à la dimension du vecteur p
//
n=500;
p=[0.19 0.17 0.16 0.10 0.08 0.07 0.06 0.055 0.04 0.03 0.025 0.02];
d=length(p);
t=[0 cumsum(p)];// graduation de [0,1] en  d intervalles de longueur p(i)
//
for i=1:d, Cand(i)=code2str(-9-i); end
// cette boucle nomme les candidats sous la forme A B C D etc.
rand("uniform");U=rand(1:n);
X=zeros(d,1); // initialisation
for i=1:d,
X(i,1)=sum(bool2s((t(i)<U)&(U<=t(i+1))));
end
//
// Calcul du rang de chaque candidat dans l'échantillon sondé
[Xtrie, rang]=sort(X);
// Affichage valeurs population et échantillon
printf('\n%-9s%-5s%8s%-7s\n\n','Candidat','% Pop','VoixEc','  % Ech')
printf('%-9s%5.2f%8i%7.2f\n',Cand,100.*p',X,100.*X./n)
// Affichage pronostic
// on veut écrire le pronostic horizontalement comme une seule
// chaîne de caractères alternant lettres et espaces
s=Cand(rang)'; // on repasse en vecteur ligne
blanc=' ';
u=blanc(ones(1:2*d));
u(1:2:2*d)=s;
prono=strcat(u);
printf('\n%s\n',...
       'Pronostic du sondeur pour l''ordre d''arrivée au 1er tour :')
printf('\n%s',prono)
// fin du script

// Script presidentielle2.sce
// Calcul de la probabilité empirique d'erreur de pronostic
// sur les noms des candidats accédant au deuxième tour de l'élection
// présidentielle. Voir presidentielle1.sce pour plus de détails.
//
// On simule N=1000 échantillons de taille n=500
p=[0.19 0.17 0.16 0.10 0.08 0.07 0.06 0.055 0.04 0.03 0.025 0.02];
```

```
d=length(p);
t=[0 cumsum(p)];// graduation de [0,1] en  d intervalles de longueur p(i)
n=500; // taille  d'échantillon
N=1000; // nombre de sondages simulés
R=zeros(1:N); //initialisation vecteur des erreurs de pronostic
rand("uniform");
X=zeros(d,1); // initialisation
for k=1:N,
   U=rand(1:n);
   for i=1:d,
      X(i,1)=sum(bool2s((t(i)<U)&(U<=t(i+1))));
   end
   // Calcul du rang de chaque candidat dans l'échantillon sondé
   [Xtrie, rang]=sort(X);
   R(k)=bool2s((rang(1)<3)&(rang(2)<3));
end
err=100.*(N-sum(R))./N;
printf('\n%s%i\n','Nombre de sondages simulés :  ',N)
printf('%s%5.2f',...
       'Pourcentage d''erreurs de pronostic sur les finalistes :  ',err)
// fin du script
```

Références

[1] P. Barbe et M. Ledoux. *Probabilité*. Espaces 34, Belin, 1998.

[2] P. Billingsley. *Probability and measure*. Wiley, third edition 1995.

[3] E. Borel. *Probabilité et certitude*. Que sais-je ? N° 445 P.U.F.

[4] W. Feller. *An Introduction to Probability Theory and its Applications*, Vol. I. Wiley.

[5] D. Foata et A. Fuchs. *Calcul des Probabilités*. Dunod, 1998.

[6] J. Jacod et P. Protter. *L'essentiel en théorie des probabilités*. Cassini, 2003.

[7] J.-P. Lubet. *Pierre-Simon Laplace (1749–1827) : la* théorie des hasards *et la* langue de l'Analyse. Atelier aux Journées Académiques sur le Hasard, Lille 15-16 avril 2004.

[8] J.-Y. Ouvrard. *Probabilités tome 2, Maîtrise–Agrégation*. Cassini, 2000.

[9] B. Pinçon. *Une introduction à Scilab*. Polycopié, version 0.999, 2003.
http://www.iecn.u-nancy.fr/~pincon/scilab/docA4.pdf

[10] D. Revuz. *Probabilités*.

[11] W. F. Stout. *Almost sure convergence*. Academic Press, 1974.

[12] Ch. Suquet. *Lois des Grands Nombres*. Polycopié, préparation à l'Agrégation de Mathématiques à Lille 1, 2003.
http://math.univ-lille1.fr/~suquet/ens/Agr/indexAgr.html

[13] Ch. Suquet. *Théorème limite central*. Polycopié, préparation à l'Agrégation de Mathématiques à Lille 1, 2003.
http://math.univ-lille1.fr/~suquet/ens/Agr/indexAgr.html

[14] Ch. Suquet *Corrigé multidigressif du T.P. Initiation à Scilab et LFGN*. Polycopié, préparation à l'Agrégation de Mathématiques à Lille 1, 2003.
http://math.univ-lille1.fr/~suquet/ens/Agr/indexAgr.html

[15] Ch. Suquet. *Introduction au Calcul des Probabilités*. Polycopié, cours de DEUG MIAS 2 et MASS 2, Lille 1, 2003.
http://math.univ-lille1.fr/~suquet/index.html

[16] Ch. Suquet. *Simulation*. Polycopié, préparation à l'Agrégation de Mathématiques à Lille 1, 2004.
http://math.univ-lille1.fr/~suquet/ens/Agr/indexAgr.html

[17] P. S. Toulouse. *Thèmes de Probabilités et Statistique*. Dunod, 1999.

[18] B. Ycart. *Démarrer en Scilab et statistiques en Scilab*. Polycopié, Université Paris V, 2001.
http://scilabsoft.inria.fr/books.html