Simulations numériques

Compte-rendu

Réalisé par **M. Jove et M. Ulas** Encadré par **M. Ginovart**

Année **2017-2018**

Introduction

Le module probabilités et statiques s'inscrit dans la seconde année de la formation IMR (Informatique Multimédia et Réseaux) à l'ENSSAT. Ce module permet de nous faire découvrir, via une mise en pratique, le chapitre nommé « simulations numériques ».

Ce TP-Projet doit permettre d'appréhender et de comprendre le théorème central limite pour établir la convergence en loi de la somme d'une suite de variables aléatoires (uniforme, exponentielle) vers la loi normale. Ce TP doit également permettre d'étudier la convergence en loi de la binomiale vers la loi de Poisson.

Par la suite, il va s'agir de choisir N variables aléatoires de même loi, d'effectuer les calculs (moyenne empirique, espérance, écart-type) liés à cette loi et de mettre en évidence la loi des grands nombres.

Finalement, au travers de ce TP, nous aborderons le sujet de la marche aléatoire pour déterminer la position d'un marcheur selon une vitesse donnée à l'instant nT. On pourra alors générer différentes trajectoires en fonction du nombre de pas d'un marcheur (-s et s).

I- Théorème central limite

Soit la variable aléatoire suivante :

$$Y_N = \frac{(\sum_{1}^{N} X_i - NE(X_i))}{\sqrt{(N)}\sigma}$$

Nous commencerons par faire varier N ainsi que les paramètres de la loi afin d'obtenir plusieurs histogrammes. Sur chaque histogramme, nous superposerons la densité de probabilité N(0,1).

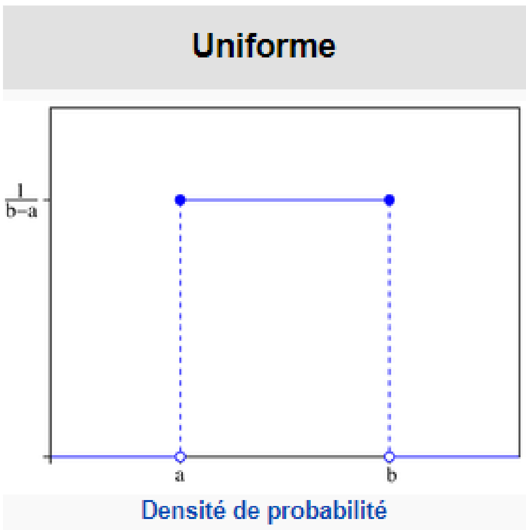
A) Partie 1

1/ Loi uniforme

La loi uniforme est définie sur un intervalle entre a et b (où a < b). C'est une loi continue où la densité de probabilité f est la fonction constante sur l'intervalle [a;b] :

$$f(x) = \begin{cases} \frac{1}{(b-a)} si\, x \in [a;b] \\ 0\, ailleurs \end{cases}$$

On s'attend à obtenir avec cette loi la densité de probabilité suivante :



On génère donc 1000 variables aléatoires de loi uniforme sur un intervalle donné.

```
function Xi=genererRandUniforme(borneA, borneB, iter)
    Xi = grand(iter,1,'unf',borneA,borneB);
endfunction
```

On affiche ensuite le cumul de ces v.a sur 50 avec la fonction histplot :

```
histplot(50,Xi,rect=[-3,0,3,1])
```

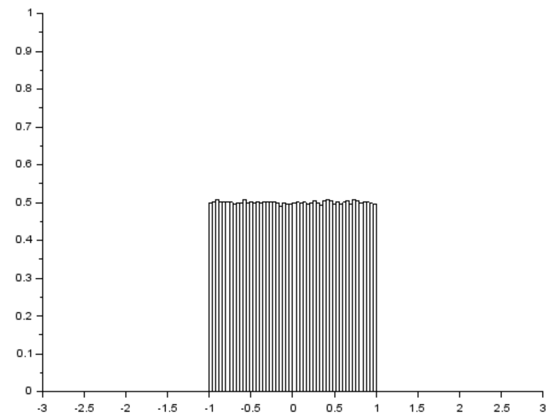


Illustration 1: Répartition loi uniforme sur l'intervalle [-1,1]

D'après notre résultat, on sait que chaque Xi est donc bien généré en suivant la loi uniforme continue. On cherche maintenant à visualiser la densité de probabilité de la loi normale N(0,1).

```
function tracerDensiteNormale(borneA, borneB)
    C=[borneA:1/1000:borneB];
    plot2d(C,exp(-C.^2/2)/sqrt(2*%pi),
endfunction
```

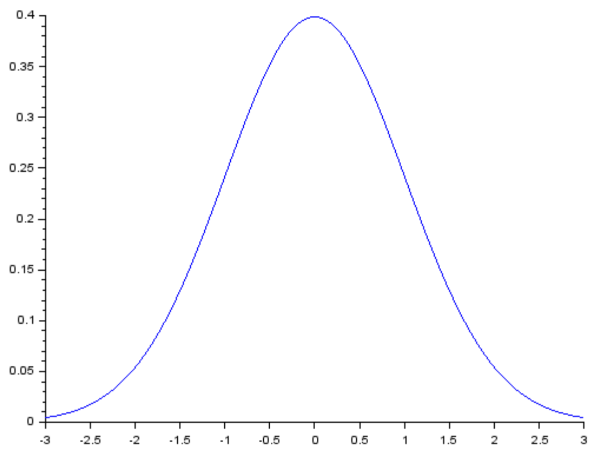


Illustration 2: Densité de probabilité de la loi normale

Enfin, nous faisons varier le nombre d'itérations sur la v.a Y_N et nous comparons les différents résultats que nous obtenons. On réalise alors un histogramme pour différentes valeurs de N (N= 100 et N=1000) où l'on superpose la densité de probabilité de la loi normale N(0,1).

```
function [Y]=centrerReduireUnif(N)

    borneA = -1;
    borneB = 1;

    esperance = (borneA + borneB)/2;
    var = ((borneB - borneA)^2)/12;

    for i=1:N
        Xi = genererRandUniforme(borneA, borneB, i);
        Y(i)= (sum(Xi) - i*esperance)/(sqrt(i)*sqrt(var));
    end
endfunction

Yn = centrerReduireUnif(100);
Yn = centrerReduireUnif(1000);
```

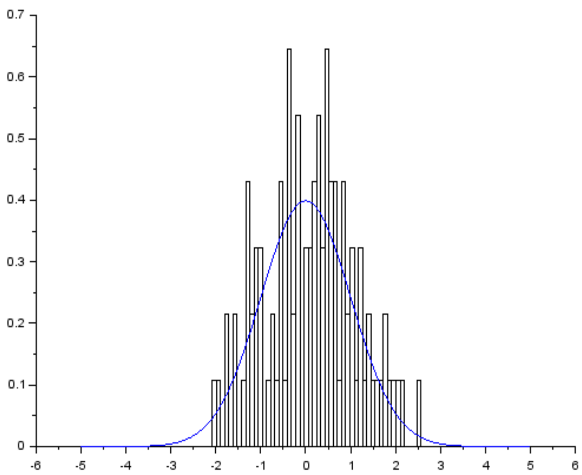


Illustration 3: Représentation de la loi Y_N avec Xi générés par la loi uniforme continue comparé à la densité de probabilité de la loi normale (N=100)

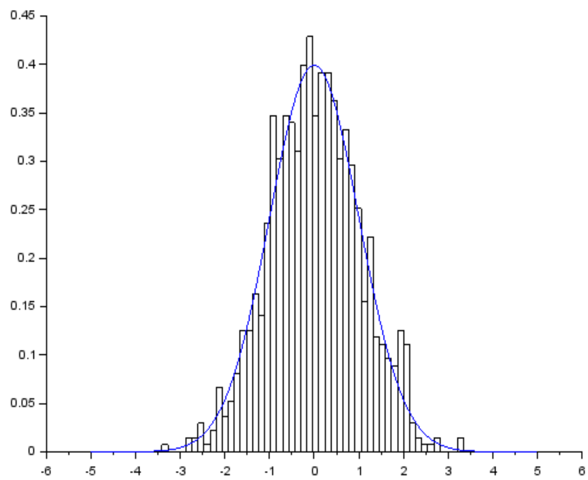


Illustration 4: Représentation de la loi Y_N avec Xi générés par la loi uniforme continue comparé à la densité de probabilité de la loi normale N=1000

On constate que lorsqu'on augmente N, la loi uniforme approxime de manière plus précise la loi densité de probabilité de la loi normale.

2/ Loi continue (exponentielle)

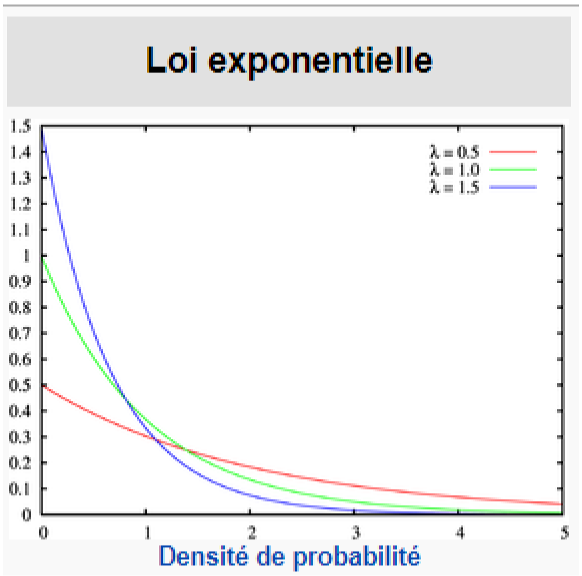
Réalisons à présent, la même expérience mais cette fois-ci avec une loi différente. Nous avons utilisé une loi continue et plus précisément la loi exponentielle pour générer les nombres aléatoires.

La loi exponentielle est une loi continue où la densité de probabilité f est la fonction constante sur l'intervalle [a;b] :

$$f(t) = 0 \text{ si } t < 0\,;$$

$$f(t) = \frac{1}{\mathbb{E}(X)} e^{-\frac{t}{\mathbb{E}(X)}} \text{ pour tout } t \geq 0.$$

En utilisant la loi exponentielle, on s'attend à obtenir une courbe ayant la forme suivante :



On génère donc 1000 variables aléatoires de loi exponentielle avec comme paramètre d'entré « lambda ».

```
function Xi=genererRandExpo(lambda, iter)
    Xi = grand(iter, 1,'exp',1/lambda);
endfunction
```

On affiche ensuite le cumul de ces v.a sur 50 valeurs :

```
lambda=1; puis lambda=2 ;

Xi= genererRandExpo(lambda, 1000);
histplot(50,Xi)

function Xi=genererRandExpo(lambda, iter)

    // On génère i lignes sur une colonne de v.a suivant la loi expo
    Xi = grand(iter,1,'exp',1/lambda);
endfunction
```

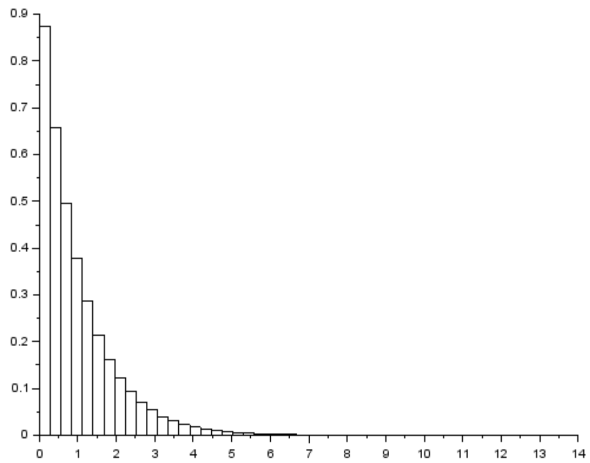


Illustration 5: Loi exponentielle avec lambda =1

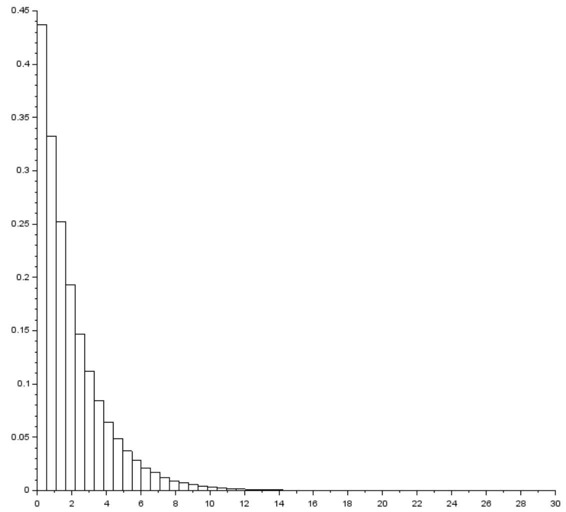


Illustration 6: Loi exponentielle avec lambda=2

Maintenant que nos Xi sont générés avec la loi exponentielle, on applique les résultats à Y_N et faisons varier N. Après ça, nous superposons à nouveau le résultat de ces différentes itérations par rapport à la densité de probabilité de la loi normale.

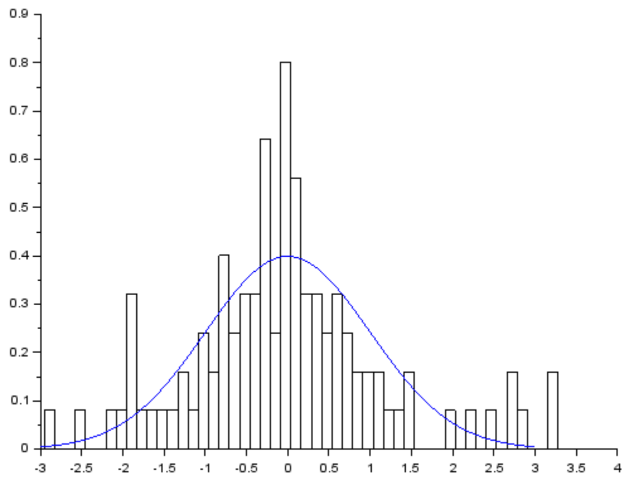


Illustration 7: Loi exponentielle sur densité de probabilité de la loi normale avec N=100

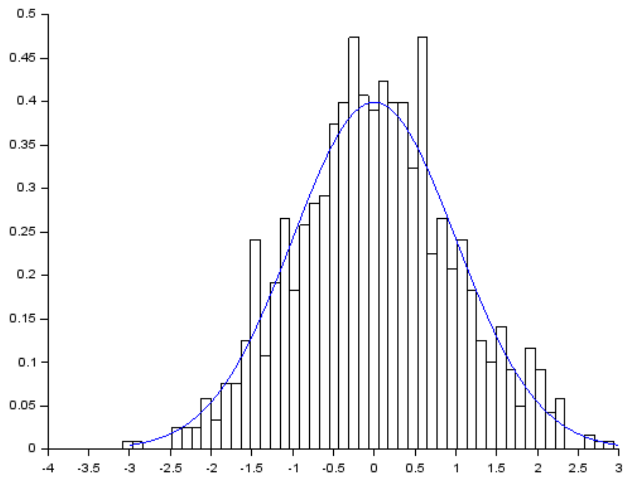


Illustration 8: Loi exponentielle sur densité de probabilité de la loi normale avec N=1000

Le constat est identique en comparant le résultat obtenu pour la loi uniforme et la loi exponentielle. Plus N augmente, plus la loi Y_N va se superposer à la densité de probabilité de la loi normale.

En définitive, lorsqu'on augmente N pour la loi Y_N avec des Xi suivants une loi exponentielle ou uniforme et que l'on superpose la densité de probabilité de la loi normale, on va obtiendra toujours une approximation de la loi normale centrée réduite N(0,1).

B) Partie 2

Dans cette deuxième, on cherche à approximer la loi binomiale vers la loi de poisson. On superposera donc l'histogramme d'une loi binomiale à la densité de probabilité d'une loi de poisson. On testera en faisant varier les paramètres d'entrés n et p.

On commence par créer la fonction pour générer les variables aléatoires suivant la loi binomiale.

```
function Xi=genererRandBinomiale(n, p, iter)
    // On génère "iter" lignes sur une colonne de v.a suivant la loi
Binomiale
    Xi = grand(iter,1,'bin',n,p);
endfunction
```

1/ Variation des paramètres n et p

Traçons tout d'abord la densité de probabilité de la loi de poisson définie par la formule ci-après :

$$p(k) = P(X = k) = \frac{\lambda^k}{k!}e^{-\lambda}$$

Pour cela voici la fonction associer afin de l'afficher sur Scilab :

```
function tracerDensitePoisson(borneA, borneB, lambda)
    x= borneA:borneB;
    y = exp(-lambda) * lambda^x ./ factorial(x);
    plot2d(x, y, rect=[0,0,10,0.3],style=2);
    xtitle("Distribution théorique", "Valeurs de k", "P(X=k)");
endfunction
```

Maintenant, générons des v.a qui suivent la loi binomiale de paramètres p=0,03 et n=100. Pour ces valeurs, on obtient λ=3.

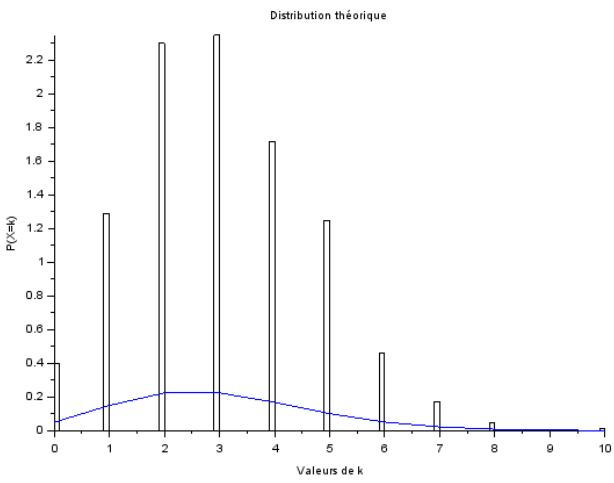


Illustration 9: En prenant en paramètres p=0,03 et n=100

Maintenant on fait varier les paramètres, p=0,5 et n=1000. On obtient pour ces valeurs λ=500.

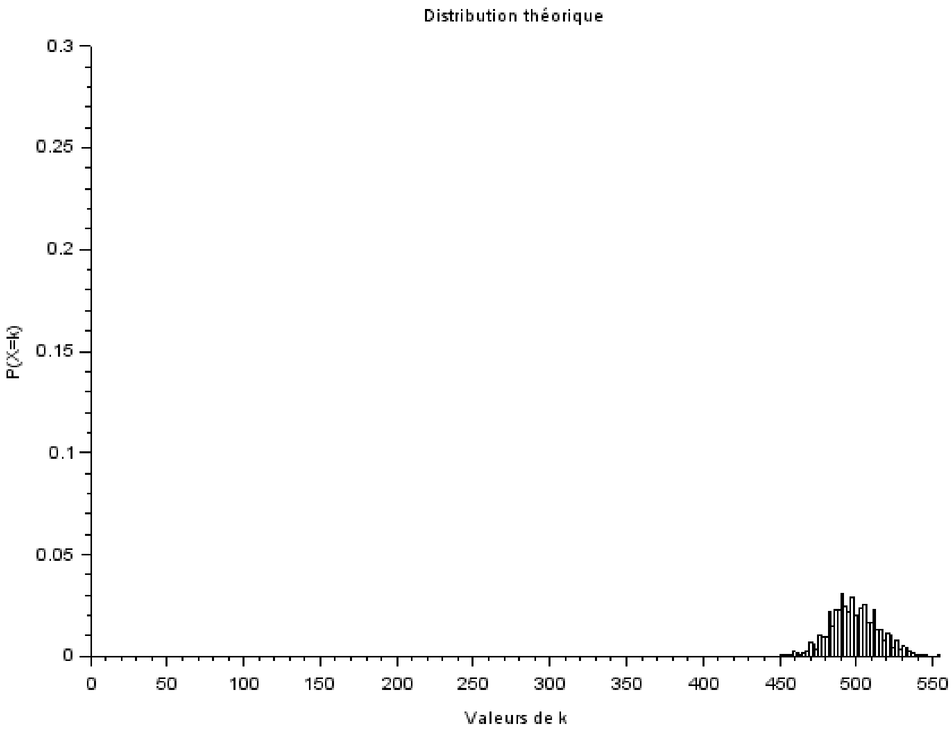


Illustration 10: En prenant en paramètres p=0,5 et n=1000

On constate d'après la figure que la courbe représentant la loi de poisson n'est plus visible. Cela s'explique par le fait que la valeur de l'exponentielle va tendre vers 0 car e(-500) ~= 7,12e-218.

2/ Conclusion

Nous avons ensuite fait varier n et p avec différentes valeurs pour comprendre le critère de convergence en loi de la loi binomiale vers la loi de Poisson. Finalement, on peut dire que si l'on fait varier les paramètres n et p, on constate que pour un n « assez grand » (n>30) et pour p voisin de 0 (p<0,1) tels que np(1-p)<= 10, on peut approcher la loi binomiale par la loi de Poisson de paramètre lambda = n*p.

II- Loi des grands nombres

Nous avons choisi de générer N variables aléatoires Xi indépendantes suivant la loi normale de paramètres (μ=0 et σ=0,2). Donc, l'espérance est équivalent à μ et son écart-type (σ) vaut la raciné carré de σ^2 soit σ. La moyenne empirique d'une loi est donnée par la formule ci-après :

$$\overline{X_N}=\frac{1}{N}*\sum_{i=1}^{N} X_i$$

Nous allons donc réaliser une moyenne empirique de notre loi précédemment citée, nous obtenons avec le code suivant ceci :

```
function [Xn]= moyenneEmpirique(lambda, N)

    for i=1:N
        Xi = genererRandNormale(0,0.2, i);
        Xn(i) = (1/i)*sum(Xi);
    end
endfunction
```

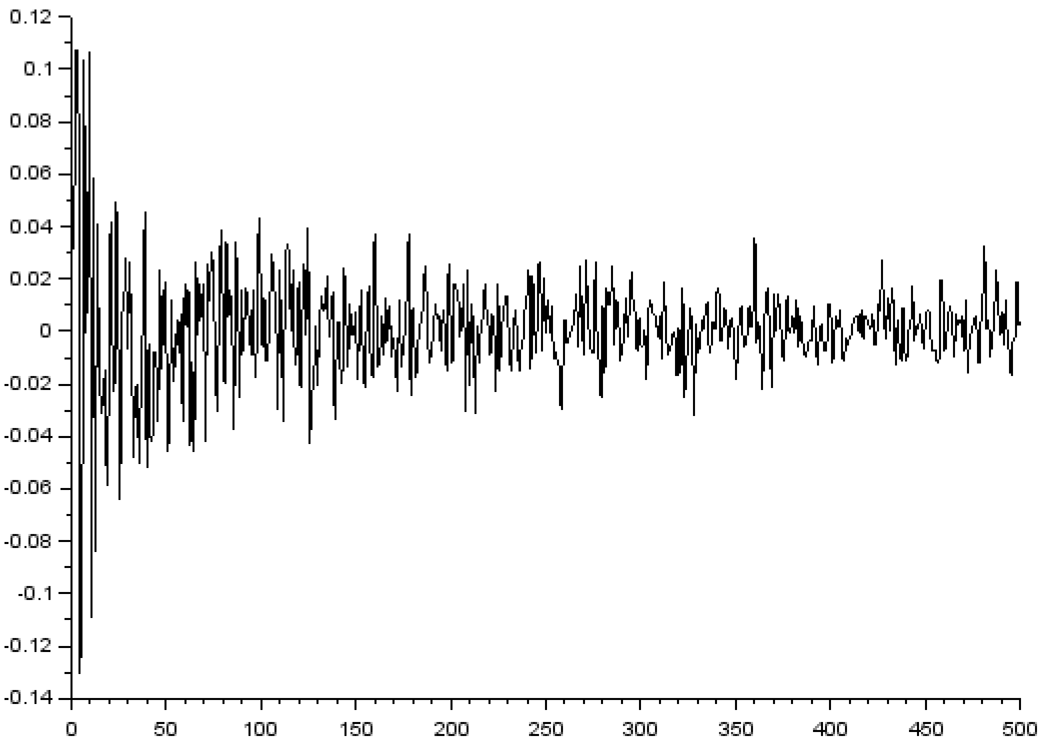


Illustration 11: Réalisation d'une moyenne empirique en fonction de N avec N=500

On constate avec une réalisation de la moyenne empirique que lorsque l'on prend un N<50, l'erreur est beaucoup plus importante. On s'attend à obtenir pour notre loi choisie une moyenne empirique avoisinant μ soit 0. or ici on voit bien que les valeurs oscillent entre -0,1 et 0,1 et sont par conséquent loin de notre 0. Aussi, l'écart-type est équivalent à σ soit 0,2 or si l'on prend deux valeurs avec un N>50, on peut se rendre compte qu'ici notre écart-type est équivalent à 0,08. On constate que plus N augmente, plus notre écart-type diminue et plus on se rapproche de l'espérance de notre loi d'origine.

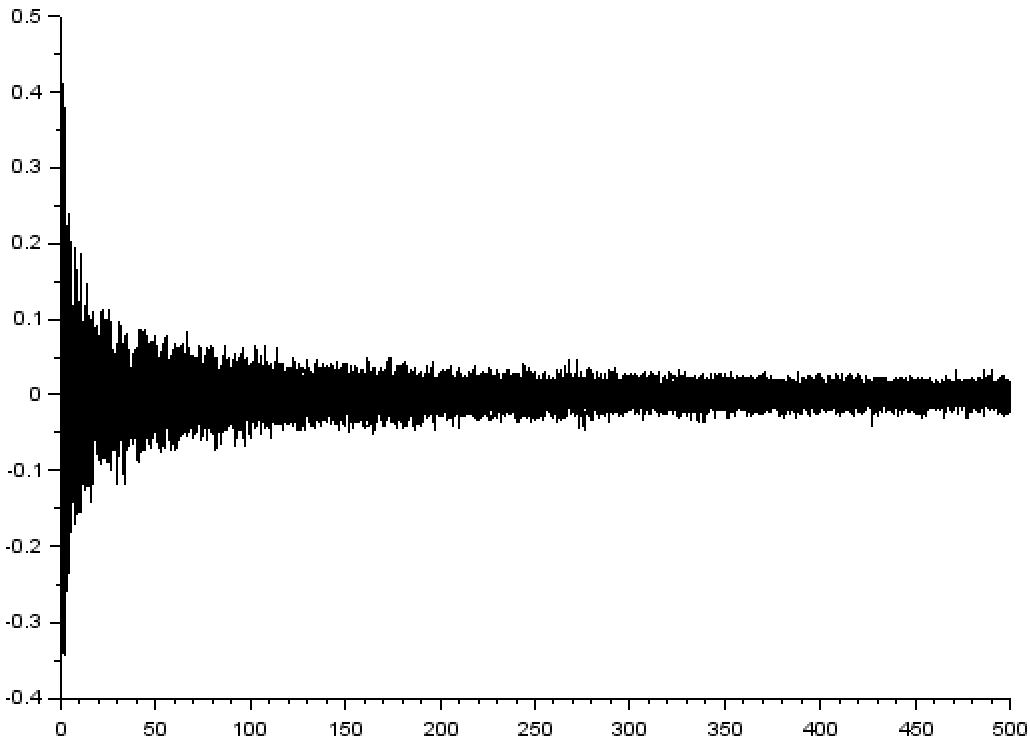


Illustration 12: Réalisation de 50 moyennes empiriques en fonction de N avec N=500

Ici, on constate bien qu'avec un N faible (<10) notre écart-type est trop élevé par rapport à nos attentes (0,8). En revanche, on voit nettement l'espérance à 0 apparaître et l'écart-type diminuer au début puis sembler se stabiliser quand N>300 avec une valeur d'environ 0,1.

La loi des grands nombres est donc bien vérifiée dans notre exercice avec la loi normale de paramètres (μ=0 et σ=0,2).

III- Marche aléatoire

Dans ce troisième exercice, on tente de modéliser une marche aléatoire. On jette une pièce bien équilibrée toutes les T secondes et selon le résultat obtenu, on effectue un pas de longueur s à droite sinon un pas de longueur -s à gauche. Chaque lancé/jet pour choisir le mouvement constitue une épreuve une de Bernoulli ayant une issue équiprobable. Après n pas, le nombre X obtenu en tirant « pile » suit une loi binomiale B(n,1/2).

$$P(X=k) = \frac{1}{2^n}\binom{n}{k}$$

où $\binom{n}{k}$ est le nombre de combinaisons de *k* éléments pris parmi *n*.

L'objectif de cette représentation est d'obtenir la position du marcheur à l'instant nT.

A) Partie 1

1) Soit Xi la v.a « déplacement n°i ». Sachant que le déplacement du marcheur est lié au lancé d'une pièce équilibrée, il se déplace à droite si face apparaît sinon à gauche. On peut donc dire que la loi suivie par le déplacement n°i est une loi de bernoulli car à chaque expérience est indépendante et identiquement distribuées de paramètre p=1/2.

2) Si un déplacement n°i constitue une loi de bernoulli et que nous avons N pas à faire durant la marche. Nous pouvons en déduire que notre marcheur suit une loi de Binomiale équivalente à B(nombre_pas, 1/2).

On peut aussi l'écrire comme cela :

$$X(nT,\omega)=X1+X2+\ldots+Xn$$

3) Représentons maintenant les trajectoires de trois marches aléatoire de différents pas. La première fera 20 pas, la seconde 100 et la dernière 1000.

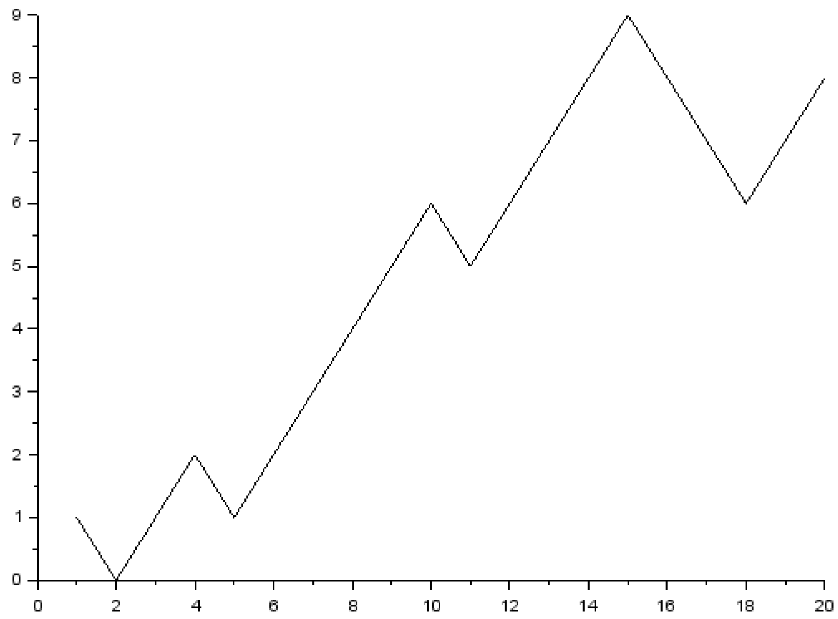


Illustration 13: Marche aléatoire de 20 pas

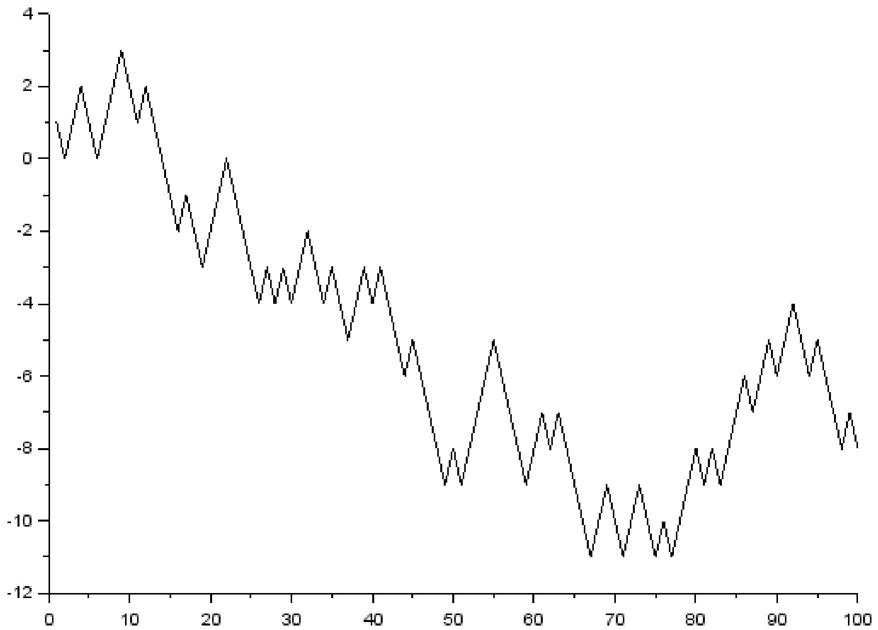


Illustration 14: Marche aléatoire de 100 pas

Illustration 15: Marche aléatoire de 1000 pas

4)

Calculons l'espérance de X(nT) :

$$E[X(nT,\omega)]=E[X1+X2+...+Xn]=\sum_{i=1}^{n}E(Xi)=n*(2*s*p-s)=0$$

Ce n'est pas ce qu'on observe sur les trois cas de nos marches. Cependant, nous sommes sans doute tombé sur des cas peu probable, il faudrait sans doute répéter une marche de 1000 pas un million de fois pour se rendre compte que notre espérance est bien 0. Sur un cas précis, il est très difficile de s'en apercevoir. Sur la marche des 1000 pas, on peut voir qu'on est resté quasiment la moitié du temps très proche de l'espérance avant que la marche diverge.

Calculons la variance de X(nT) :

$$V[X(nT,\omega)]=V[X1+X2+...+Xn]=\sum_{i=1}^{n}V(Xi)=n$$

Donc la variance croit en fonction du temps qui passe.

B) Partie 2

5) En accélérant le rythme de la promenade, en prenant T de plus en plus petit et un pas également de plus en plus petit. Plus précisément on pose :

$$s^2 = \alpha T$$

On a donc un pas de s² et -s², calculons l'espérance de X :

$$\|X(nT)\|^2 = E(X(nT)^2) = ns^2 = \frac{(ts^2)}{T} s \to 0 \; avec \; s = \sqrt{(T)}$$

On s'attend à avoir grâce à la réduction du pas et du temps entre chaque lancé de pièce, à réduire la variance. Cela va selon nous réduire la vitesse d'agrandissement de la variance. Ainsi, on obtiendra après chaque marche un résultat souvent beaucoup plus proche de l'espérance. Suivant cette logique, on imagine assez bien que sur 1000 marches aléatoires, la grande majorité de résultat sera contenu dans la variance calculée et qu'en comparant le tableau de nos résultats de l'ensemble des marches avec une loi normale N(0,sqrt(T)), on trouvera probablement une courbe similaire.

6) Représentation des trajectoires de X avec n fixé et T de plus en plus petit :

```
function marche=marcheAleatoireReduc(nombre_pas, T, s)

    alpha = 1;
    for i=1:T:nombre_pas
        pas = genererRandBinomiale(1,0.5,1);
        s = sqrt(alpha * T);
        if pas == 0 then
            tabPas(i) = s;
        elseif pas == 1 then
            tabPas(i) = -s;
        end
        marche(i) = sum(tabPas);
        alpha = alpha - alpha/1000;
    end
endfunction
```

Notre alpha est à chaque tour mise à jour à la baisse, on obtient le résultat suivant pour une marche de 10000 pas :

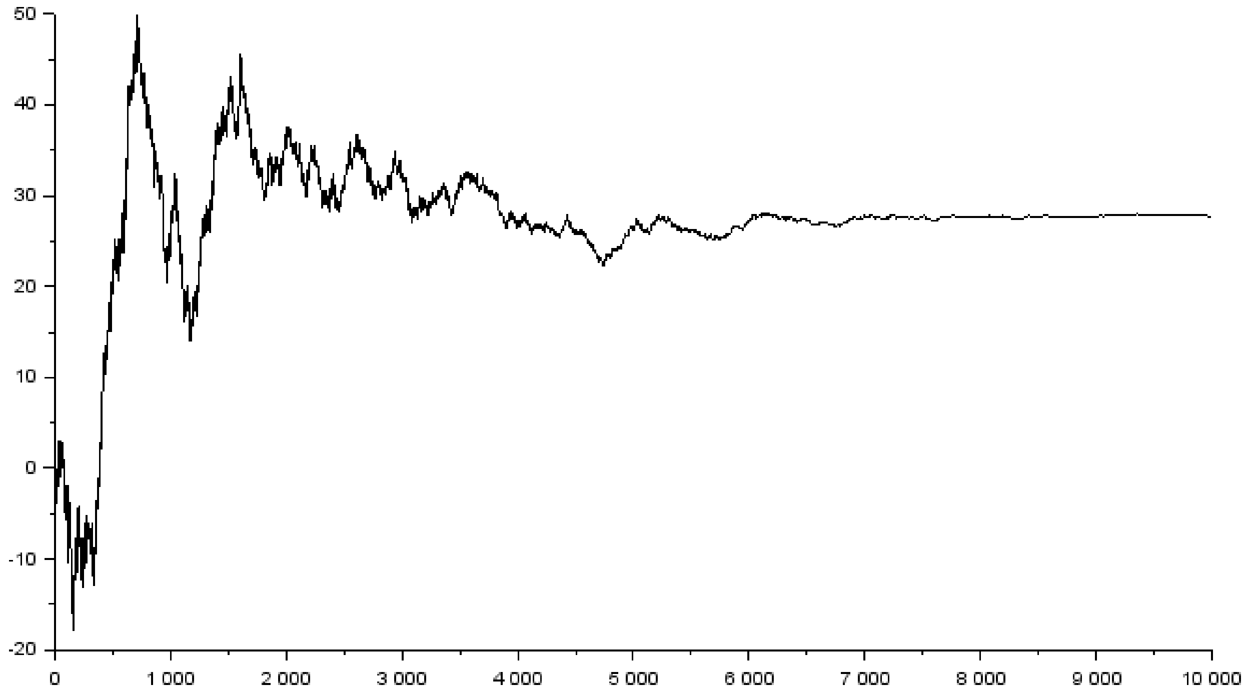


Illustration 16: Marche aléatoire de 10000 pas avec T de plus en plus petit au fil du temps

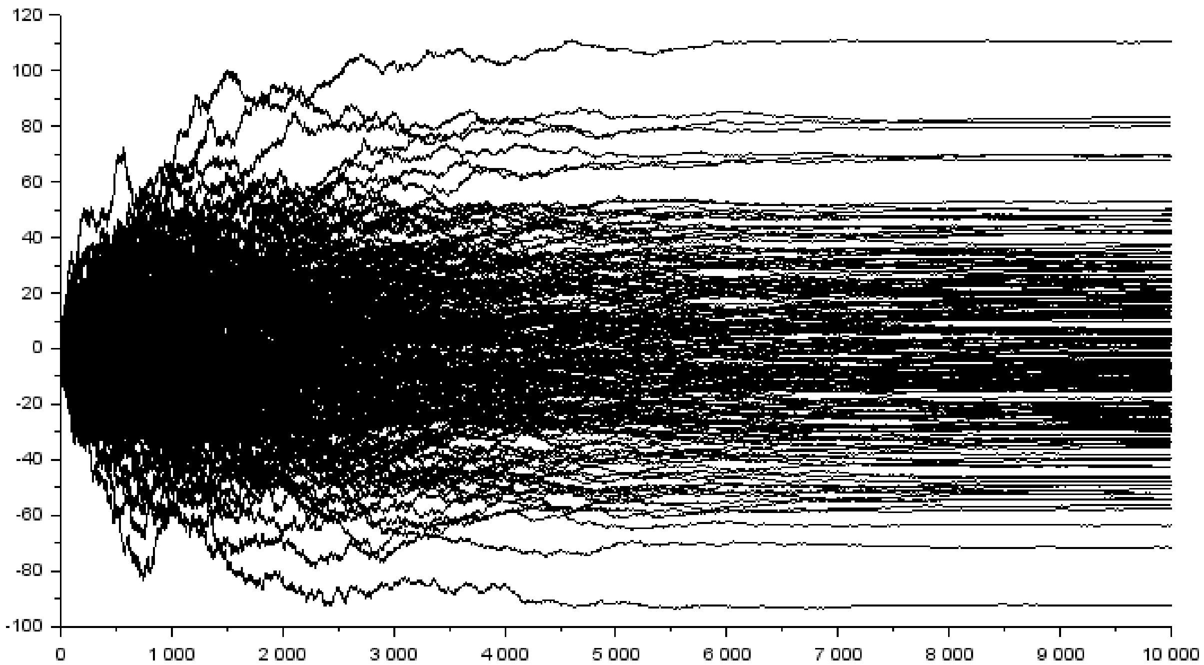


Illustration 17: 200 marches aléatoires de 10 000 pas

7) Comme vu ci-dessus, on remarque qu'a chaque fin de marche de 10 000 pas, une très grande majorité des marches se situent au final entre -25 et 25 par rapport à l'espérance = 0 puisque l'on fait tendre s avec la racine carré de T. On estime donc une variance de 50 même si l'on ne la pas calculée.

Représentons maintenant le résultat de ces 200 marches :

```
for i=1:200
       marche4 = marcheAleatoireReduc(10000,1,1);
       res(i) = marche4($)
end

histplot(50, res)
```

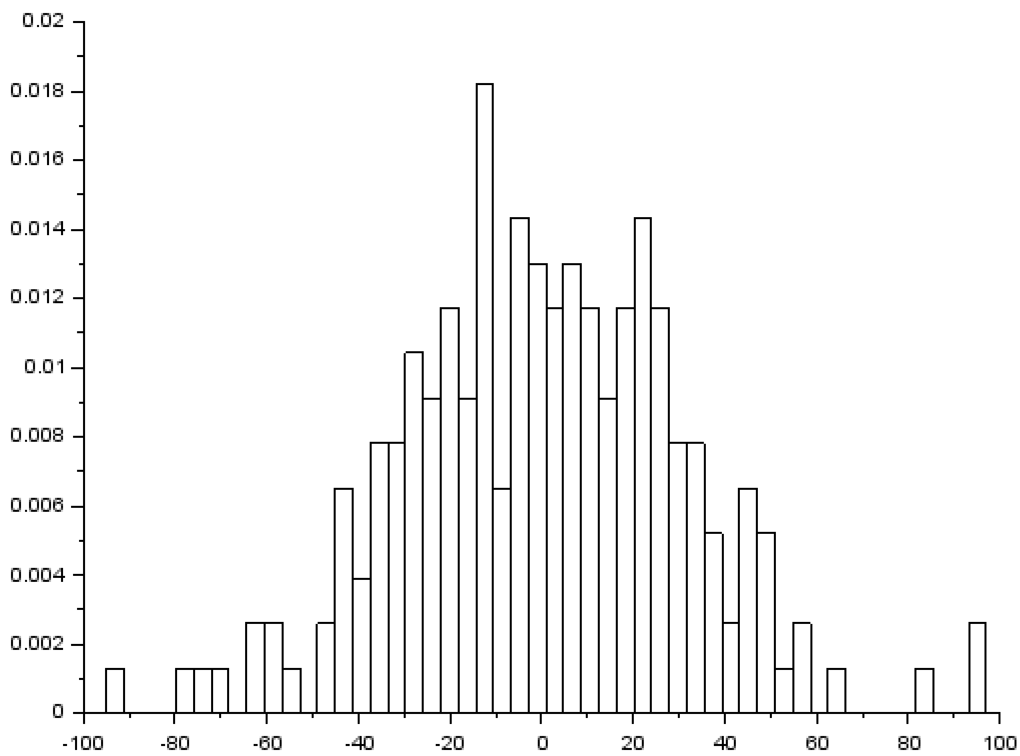


Illustration 18: Histogramme du résultat de 200 marches aléatoire avec s qui tend vers sqrt(T)

On reconnaît alors que le résultat obtenu semble tendre vers une loi normale de paramètres N(0, sqrt(T)) où T vaut 0.0000452 (après 10 000 pas).

UNIVERSITÉ DE RENNES 1

ENSSAT LANNION

Conclusion

En conclusion, ce projet nous a permis d'approfondir nos connaissances sur le thème des simulations numériques. Les différentes étapes constituant le TP nous ont permis de mener une réflexion de bout en bout sur les exercices (théorème central limite, loi des grands nombres et la marche aléatoire).

La première partie portant sur le théorème central limite avait pour objectif de démontrer la convergence en loi de la somme de variables aléatoires vers la loi normale. Pour cela, nous avons augmenté N en constatant une amélioration de la précision qui se caractérisait par la convergence en loi vers la loi normale. Ensuite, il s'agissait de réaliser le même principe mais avec la superposition de la loi binomiale sur la loi de Poisson. Nos analyses nous ont permis d'établir que si l'on faisait varier les paramètres n et p, on constatait que pour un n « assez grand » (n>30) et pour p voisin de 0 (p<0,1) tels que np(1-p)<= 10, on pouvait approcher la loi binomiale par la loi de Poisson de paramètre lambda = n*p.

La deuxième partie avait pour objectif de choisir une loi quelconque et de mettre en évidence la loi des grands nombres. Pour ce faire, nous avons choisi la loi normale et nous avons réalisé graphiquement 50 moyennes empiriques en fonction de N avec N=500. Nous avons constaté que pour un N faible notre écart-type était trop élevé par rapport à nos attentes (0,8). En revanche, pour N>300 nous avons constaté que l'espérance à 0 apparaissait et que l'écart-type diminuait au début puis semblait se stabiliser quand N>300 avec une valeur d'environ 0,1.

La dernière partie portant sur la marche aléatoire avait pour but de déterminer la position d'un marcheur selon une vitesse donnée à l'instant nT. Nous avons donc généré différentes trajectoires en fonction du nombre de pas d'un marcheur (-s et s).