

# Traiter du hasard : Hasardologie

Jean-Marc Duquesnoy  
Professeur de mathématiques au  
Lycée André Malraux de Béthune

IREM de Lille  
`jean-marc.duquesnoy@ac-lille.fr`

Dernière mise à jour : 3 janvier 2020

*Le*

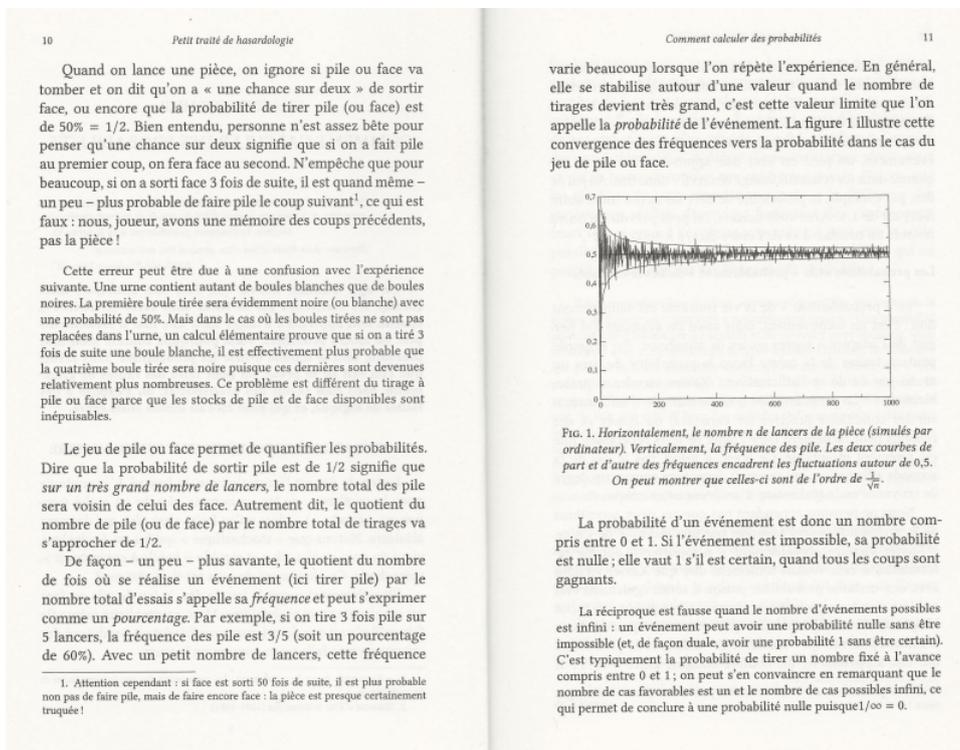
*jamais de quoi on parle, ni si ce qu'on dit e*  
**Bertrand Russel**

# Table des matières

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Première partie : simuler le lancer d'une pièce : jeu du « Pile ou Face »</b>	<b>5</b>
2.1	Premier essai . . . . .	5
2.2	Deuxième essai . . . . .	5
2.3	Troisième essai . . . . .	6
2.4	Quatrième essai . . . . .	6
2.5	Une explication probabiliste . . . . .	7
2.5.1	La fonction <code>alea()</code> . . . . .	7
2.5.2	Loi de la variable $Y$ . . . . .	7
2.5.3	Que se passe-t-il si on utilise la fonction <code>random()</code> ? . . . . .	8
2.6	A vous d'expérimenter le hasard . . . . .	9
2.6.1	Une nouvelle fonction <code>alea()</code> . . . . .	9
2.6.2	Une autre façon encore de définir la fonction <code>alea()</code> . . . . .	9
<b>3</b>	<b>Deuxième partie : Méthode de Monte-Carlo</b>	<b>10</b>
3.1	T'as pas vu Monte-Carlo 😊 . . . . .	10
3.2	Même à Monte-Carlo, ça triche 😞 . . . . .	11
3.3	Et au bac alors? . . . . .	13
<b>4</b>	<b>Conclusions</b>	<b>15</b>
<b>5</b>	<b>Remerciements</b>	<b>15</b>

# 1 Introduction

Ce qui a déclenché en moi la motivation d'écrire ce qui suit est la lecture du livre d'Hubert Krivine : *Traité d'hasardologie*, en particulier ces deux pages



mais aussi un *document ressources* sur l'algorithmie et la programmation<sup>1</sup>, à partir de la page 12 jusqu'à la page 23, que l'on peut trouver en suivant le lien [ici](#) 📖.

Dans les deux cas, les auteurs ne précisent pas les lois qui ont été choisies pour réaliser les différentes simulations exposées, ce qui pourrait laisser penser que, dès lors que l'on réalise une simulation d'une épreuve aléatoire, le choix d'une loi uniforme discrète ou continue est implicite.

Ce document sera l'occasion de mettre en scène la **loi des grands nombres**<sup>2</sup> au travers de différentes simulations d'un jeu de Pile ou Face, mais aussi de la méthode de Monte-Carlo, l'objectif étant de pointer du doigt une mauvaise utilisation de l'expression « tirer au hasard un réel entre 0 et 1 ».

1. Pour être honnête, je dois reconnaître que l'on y indique, page 8, que la fonction `random.randint(1,50)` renvoie un entier choisi au hasard suivant la loi uniforme sur  $[[1; 50]]$ , mais c'est le seul moment dans le document où l'on précise le modèle choisi pour simuler une épreuve aléatoire.

2. **Théorème** (loi forte des grands nombres pour les fréquences) : Soit  $S_n$  le nombre de succès obtenus au cours de  $n$  épreuves répétées indépendantes, avec pour chaque épreuve la probabilité de succès égale à  $p$ . Quand  $n$  tend vers l'infini, la fréquence de succès  $S_n/n$  converge avec probabilité 1 vers  $p$ . On dit aussi que  $S_n/n$  converge presque-sûrement vers  $p$ .

### Avertissements :

- Le langage utilisé pour les simulations sera le langage python <sup>3</sup>.
- J'ai inclus le code python de chacun des programmes de simulation directement dans le source L<sup>A</sup>T<sub>E</sub>X de ce document, le package *pythontex.sty* ayant été inséré dans le préambule. Concrètement cela implique que les scripts python sont interprétés à chaque compilation du source L<sup>A</sup>T<sub>E</sub>X <sup>4</sup> et que les résultats des calculs des fréquences observées et les figures tracées que vous observerez en lisant ce document changent, de fait. Cela donnera plus de crédit à ce document : on ne pourra pas me suspecter de choisir les résultats qui pourraient orienter les conclusions.
- Le lecteur constatera que les graphiques générés par quelques scripts python sont enregistrés suivant un chemin qui m'est personnel. Charge à lui de renseigner la commande *savefig* comme bon lui semble.

---

3. Préconisé par l'**Inspection Générale** depuis l'année scolaire 2017-2018 en classe de seconde.

4. On compile trois fois successivement le source L<sup>A</sup>T<sub>E</sub>X à l'aide de *pdflatex*, puis *pythontex* et de nouveau *pdflatex*.

## 2 Première partie : simuler le lancer d'une pièce : jeu du « Pile ou Face »

Comme dans le document d'Hubert Krivine, je vais simuler le lancer d'une pièce.

**Méthode de simulation choisie :** On tire un nombre réel au hasard entre 0 et 1, si ce nombre est inférieur à 0.5 alors on décide qu'on a obtenu « Pile », sinon « Face »<sup>5</sup>.

On va calculer et afficher la fréquence d'apparition du Pile lorsqu'on simule 100 lancers successifs et indépendants<sup>6</sup> de la pièce.

### 2.1 Premier essai

On lance la pièce 100 fois successivement.

Le script *python* : 

```
1 from math import *
2 from random import *
3 def simu(n):
4     f=0
5     for i in range(0,n):
6         if alea() < 0.5:
7             f+=1
8     print('la fréquence observée de Pile pour ',n, 'lancers est égale à ',f/n)
```

La commande *simu(100)* appelée deux fois successivement affiche alors dans la console

la fréquence observée de Pile pour 100 lancers est égale à 0.31 None

la fréquence observée de Pile pour 100 lancers est égale à 0.27 None

#### Interprétation du résultat affiché

Le résultat vous paraît-il cohérent ? bof!!

### 2.2 Deuxième essai

On lance la pièce 1000 fois successivement.

La commande *simu(1000)* appelée deux fois successivement affiche alors dans la console

la fréquence observée de Pile pour 1000 lancers est égale à 0.296 None

la fréquence observée de Pile pour 1000 lancers est égale à 0.313 None

#### Interprétation du résultat affiché

Décidément 😞 !!

5. Pour un élève, se posera la question de savoir si la condition  $<0.5$  peut être remplacée par  $\leq 0.5$  ?

6. Doit-on s'interroger sur l'indépendance des simulations successives.

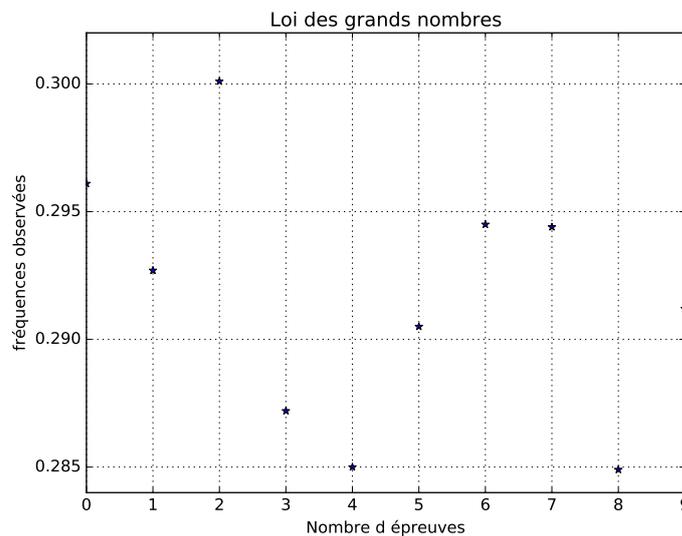
## 2.3 Troisième essai

On simule 10000 lancers 10 fois de suite, puis on observe l'évolution de la fréquence calculée pour chacun des 10000 lancers <sup>7</sup>.

Le script *python* : 

```
1 from math import*
2 from random import*
3 from matplotlib import*
4 from matplotlib.pyplot import*
5 # p représente le nombre de fois où l'on simule n lancers
6 # f représente la fréquence observée de Pile à l'issue de chaque série de n lancers
7 def simu(n,p):
8     L=[]
9     for i in range(0,p):
10        f=0
11        for i in range(0,n):
12            if alea() < 0.5:
13                f+=1
14        L.append(f/n)
15 # axis([0,p,0,1])
16 xlabel('Nombre d'épreuves')
17 ylabel('fréquences observées')
18 title('Loi des grands nombres')
19 grid(True)
20 plot(L,'*')
21 savefig('/media/jm/C5DC-39E4/ANNEE-2018-2019/IREM/PileouFace/Graphe1.pdf')
```

On a lancé la commande *simu(10000,10)* qui donne en sortie le graphique suivant :



### Interprétation du résultat affiché

Réaliser l'épreuve dix fois, est-ce suffisant ?

## 2.4 Quatrième essai

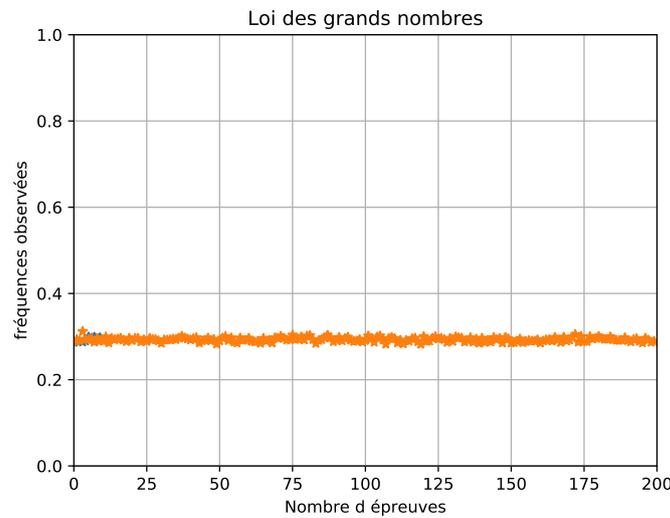


et si on simule l'épreuve 200 fois et observe l'évolution de la fréquence.

---

7.  Je n'affiche pas l'évolution de la fréquence observée en fonction du nombre croissant de simulations.

L'affichage à la compilation <sup>8</sup>, en saisissant la commande `simu(10000,200)` qui donne en sortie le graphique suivant :



### Interprétation du résultat affiché

On ne comprend toujours rien!!!!!! 😞

On pouvait s'attendre à lire une fréquence observée qui se « rapproche » de la probabilité théorique d'obtenir Pile égale à  $\frac{1}{2}$ , comme observé sur le graphique affiché dans le document d'Hubert Krivine.

Que fait la loi des grands nombres ?

Je vous laisse méditer quelque temps, pour revenir vers vous bientôt.....

## 2.5 Une explication probabiliste

### 2.5.1 La fonction `alea()`

Si on fouille dans la librairie `random`, on ne trouve pas trace de la fonction `alea()`.

Ce n'est que normal, elle n'y figure pas.

Je l'ai définie en préambule du script qui simule un lancer de la pièce.

En effet, j'ai exécuté le script suivant, afin de définir la fonction `alea()` qui sera interprétée dès qu'on « l'appellera » dans un script compilé.

Le script `python` : 📄

```

1 from math import *
2 from random import *
3 def alea():
4     t=random()
5     return 2*t-t**2

```

Vous remarquez que la fonction `alea()` renvoie un réel  $2t - t^2$ , où  $t$  est un réel qui sera choisi aléatoirement suivant **la loi uniforme** sur  $[0; 1]$  <sup>9</sup>.

On remarquera que, si  $t \in [0; 1]$ , alors  $2t - t^2 \in [0; 1]$ .

La question qui se pose est de connaître la loi suivie par la variable  $Y = 2X - X^2$  où  $X \sim U_{[0;1]}$ .

8. J'ai ajouté la commande `axis([0,p,0,1])` dans le code pour observer les fréquences entre 0 et 1.

9. La fonction `random()` fait ce travail.

### 2.5.2 Loi de la variable $Y$

Je détermine la fonction de répartition  $F$  de la variable  $Y$ .

Pour tout réel  $t \in [0; 1]$ ,

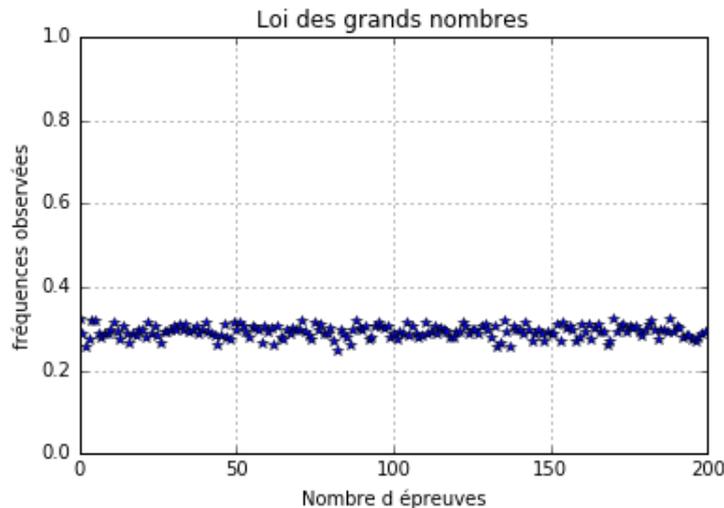
$$\begin{aligned} F(t) &= P(Y \leq t) \\ \Leftrightarrow F(t) &= P(2X - X^2 \leq t) \\ \Leftrightarrow F(t) &= P(X^2 - 2X + t \geq 0) \\ \Leftrightarrow F(t) &= P((X - 1)^2 \geq 1 - t) \\ \Leftrightarrow F(t) &= P([X - 1 \leq -\sqrt{1 - t}] \cup [X - 1 \geq \sqrt{1 - t}]) \\ \Leftrightarrow F(t) &= P([X \leq 1 - \sqrt{1 - t}] \cup [X \geq 1 + \sqrt{1 - t}]) \\ \Leftrightarrow F(t) &= P(X \leq 1 - \sqrt{1 - t}) + P(X \geq 1 + \sqrt{1 - t}) \\ \Leftrightarrow F(t) &= P(X \leq 1 - \sqrt{1 - t}) \\ \Leftrightarrow F(t) &= 1 - \sqrt{1 - t} \end{aligned}$$

Cela nous donne en particulier  $F\left(\frac{1}{2}\right) = 1 - \sqrt{1 - \frac{1}{2}}$

```
1 from math import *
2 from sympy import *
3 from random import *
4 def F(t):
5     return 1-sqrt(1-t)
```

donc  $F\left(\frac{1}{2}\right) = \frac{\sqrt{2} - 1}{\sqrt{2}} \approx 0.292893218813452$

On comprend mieux maintenant pourquoi la commande `simu(10000,200)` affiche en sortie le graphique qui suit :



On a, dès les premiers scripts, simulé le lancer d'une pièce « truquée » !!!!!

### 2.5.3 Que se passe-t-il si on utilise la fonction `random()` ?

Voici un script qui définit la simulation du lancer d'une pièce « non truquée » : le script *python* :



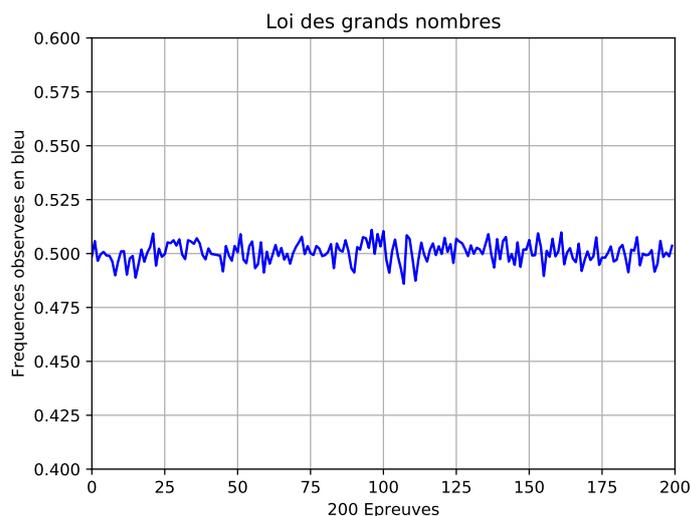
```
1 from math import *
2 from random import *
```

```

3 from matplotlib import*
4 from matplotlib.pyplot import*
5 def ldgn(n,p):
6     L=[]
7     for i in range(n):
8         c=0
9         for j in range(p):
10            if random() <0.5:
11                c=c+1
12            f=float(c)/p
13            L.append(f)
14    axis([0,n,0.4,0.6])
15    xlabel(' 200 Epreuves ')
16    ylabel(' Frequences observees en bleu ')
17    title('Loi des grands nombres')
18    grid(True)
19    plot(L,'b-')
20    savefig('/media/jm/C5DC-39E4/ANNEE-2018-2019/IREM/PileouFace/Graphe4.pdf')

```

On a lancé la commande `ldgn(200,10000)` qui donne en sortie le graphique suivant :



## 2.6 A vous d'expérimenter le hasard

### 2.6.1 Une nouvelle fonction `alea()`

J'ai codé et exécuté le script suivant :

le script *python* : 

```

1 from math import*
2 from random import*
3 from matplotlib import*
4 from matplotlib.pyplot import*
5 def alea2():
6     t=random()
7     return t**2
8
9
10 def ldgn2(n,p):
11     L=[]
12     for i in range(n):

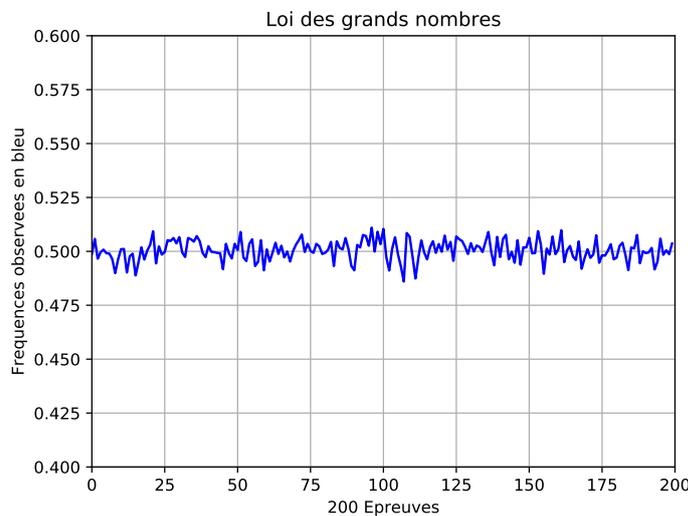
```

```

13     c=0
14     for j in range(p):
15         if alea2() < 0.5:
16             c=c+1
17         f=float(c)/p
18         L.append(f)
19     axis([0,n,0.4,0.6])
20     xlabel(' 200 Epreuves ')
21     ylabel(' Frequences observees en bleu ')
22     title('Loi des grands nombres')
23     grid(True)
24     plot(L, 'b-')
25     savefig('/media/jm/C5DC-39E4/ANNEE-2018-2019/IREM/PileouFace/Graphe5.pdf')

```

et ai obtenu le graphique suivant



**Indication**  : Pour définir la fonction  $alea2()$ , j'ai utilisé la variable aléatoire  $Y = X^2$ , où  $X \sim U_{[0;1]}$ .

**Question** : Vers quelle probabilité théorique les fréquences observées doivent-elles converger si on augmente le nombre de lancers simulés ?

### 2.6.2 Une autre façon encore de définir la fonction $alea()$

J'ai procédé de la même façon, en ayant pris pour  $Y$  la variable aléatoire définie par  $Y = \max(X_1; X_2)$ , où  $X_1 \sim U_{[0;1]}$  et  $X_2 \sim U_{[0;1]}$ ,  $X_1$  et  $X_2$  étant indépendantes.

**Question** : Vers quelle probabilité théorique les fréquences observées doivent-elles converger si on augmente le nombre de lancers simulés ?

## 3 Deuxième partie : Méthode de Monte-Carlo



Pour la loi uniforme sur un carré quelconque  $C$ , la probabilité d'appartenance de  $(X, Y)$  au sous-ensemble  $R$  de  $C$  serait,  $aire(R)/aire(C)$ .

Rappelons qu'une probabilité n'a pas d'unité et que lorsque l'on écrit  $P((X, Y) \in R) = aire(R)$ , c'est un abus de langage.

Par exemple si le carré unité a pour aire  $1m^2$  et si  $aire(R) = 0,37m^2$ , il faut comprendre que  $P((X, Y) \in R) = 0,37$ .

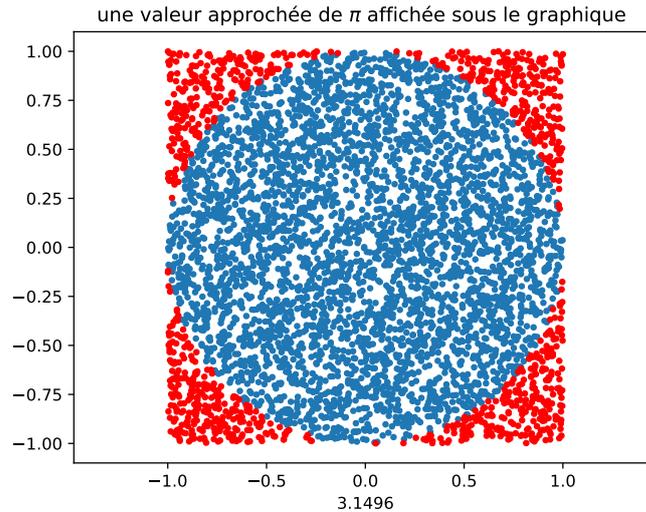
### 3.1 T'as pas vu Monte-Carlo 😊

En m'inspirant d'un des fichiers qu'une collègue m'a transmis à l'issue d'un stage sur la programmation en Python, je vous propose le script suivant, qui met en scène la méthode de Monte-Carlo pour estimer  $\pi$  :

le script *python* : 

```
1 from math import *
2 from random import *
3 from matplotlib import *
4 from matplotlib.pyplot import *
5 clf()
6 N=5000
7 C=0
8 L1=[]
9 L2=[]
10 L3=[]
11 L4=[]
12 for i in range(N) :
13     x=2*random()-1
14     y=2*random()-1
15     if x**2+y**2<=1 :
16         C+=1
17         L1.append(x)
18         L2.append(y)
19     else :
20         L3.append(x)
21         L4.append(y)
22 pi=4*C/N
23 axis( [ -1 , 1 , -1,1 ] )
24 axis( 'equal' )
25 plot(L1,L2, ". ")
26 plot(L3,L4, ". ", color="red")
27 title('une valeur approchée de $\pi$ affichée sous le graphique ')
28 xlabel(round(pi,4))
29
30 savefig( '/media/jm/C5DC-39E4/ANNEE-2018-2019/IREM/PileouFace/Graphe8.pdf' )
31
32
```

exécuté, ce script affiche en sortie le graphique suivant :



La méthode de Monte Carlo donne ici :  $\pi \approx 3.1496$

### 3.2 Même à Monte-Carlo, ça triche 😞

Dans ce qui suit, j'ai simulé des points dans le carré  $[0; 1] \times [0; 1]$  pour, comme dans le script précédent, déterminer une valeur approchée de  $\frac{\pi}{4}$ .

J'utilise comme fonction *alea()* la racine carrée de la fonction *random()*.

Cette fois-ci, je ne cache rien :

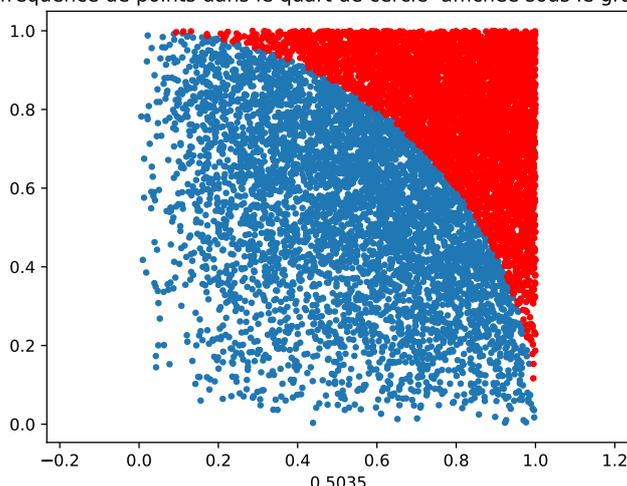
le script *python* : 📄

```

1 import random
2 from math import *
3 import matplotlib.pyplot as plt
4 clf()
5 N=10000
6 C=0
7 L1=[]
8 L2=[]
9 L3=[]
10 L4=[]
11 for i in range(N) :
12     x=sqrt(random.random())
13     y=sqrt(random.random())
14     if x**2+y**2<1 :
15         C+=1
16         L1.append(x)
17         L2.append(y)
18     else :
19         L3.append(x)
20         L4.append(y)
21 f=C/N
22 axis('equal')
23 plot(L1,L2,".")
24 plot(L3,L4,".",color="red")
25 title('La fréquence de points dans le quart de cercle affichée sous le graphique.')
26 xlabel(f)
27 savefig('/media/jm/C5DC-39E4/ANNEE-2018-2019/IREM/PileouFace/Graphe9.pdf')
```

Voici le graphique en sortie :

La fréquence de points dans le quart de cercle affichée sous le graphique.



Il est visible que la distribution des points dans le carré n'est plus uniforme.

De plus,  $\frac{\pi}{4} \approx 0.7854$ , et la fréquence affichée est 0.5035 .

On peut s'interroger sur la méthode....

Il est clair qu'il nous faut déterminer la loi de la variable aléatoire  $Z = X_1^2 + Y_2^2$ , où  $X_1 = \sqrt{X}$  et  $Y_1 = \sqrt{Y}$ , avec  $X \sim U_{[0;1]}$  et  $Y \sim U_{[0;1]}$ ,  $X$  et  $Y$  étant indépendantes.

Le problème se ramène donc à la recherche de la loi de  $Z = X + Y$ .

Notons  $f$  et  $g$  les densités de probabilité des variables  $X$  et  $Y$ .

$f$  et  $g$  valent 1 sur  $[0; 1]$  et sont nulles partout ailleurs.

Quelle est la densité  $f_Z$  de la variables  $Z$  ?

**Rappel** : Pour tout réel  $z$ ,

$$f_Z(z) = (f * g)(z) = \int_0^1 f(z - y)g(y)dy$$

donc

$$f_Z(z) = \int_0^1 f(z - y)dy$$

Or,  $f(z - y) \neq 0$  si  $0 \leq z - y \leq 1$ , donc si  $z - 1 \leq y \leq z$ .

De fait,

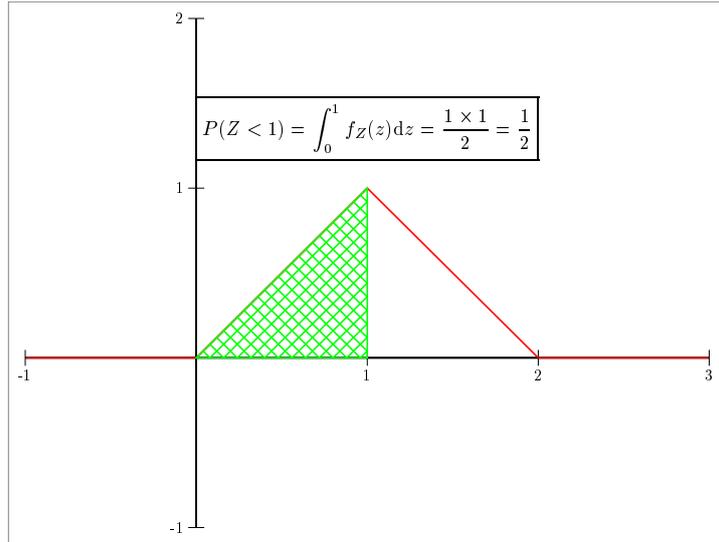
$$f_Z(z) = (f * g)(z) = \int_{z-1}^z f(z - y)dy$$

Si  $z < 0$  ou  $z > 2$ ,  $(f * g)(z) = 0$

Si  $0 \leq z \leq 1$ ,  $(f * g)(z) = \int_0^z dy = z$

Si  $1 \leq z \leq 2$ ,  $(f * g)(z) = \int_{z-1}^1 dy = 2 - z$ .

La densité  $f_Z$  est donc représentée par :



La probabilité qu'un point simulé se trouve dans le quart de cercle est donc égale à

$$P(Z < 1) = \int_0^1 f_Z(z) dz = \frac{1}{2}.$$

Cela permet de valider la fréquence observée de points simulés dans le quart de cercle qui était alors égale à 0.5035, donc proche de la probabilité théorique que l'on a trouvée égale à  $\frac{1}{2}$ .

OUF!!

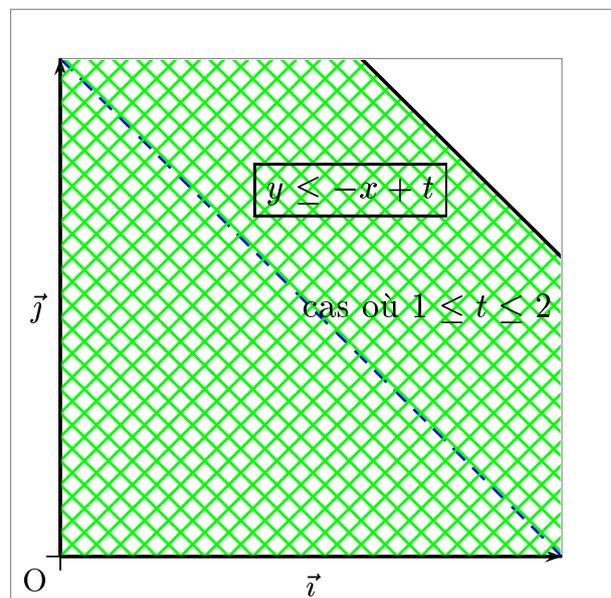
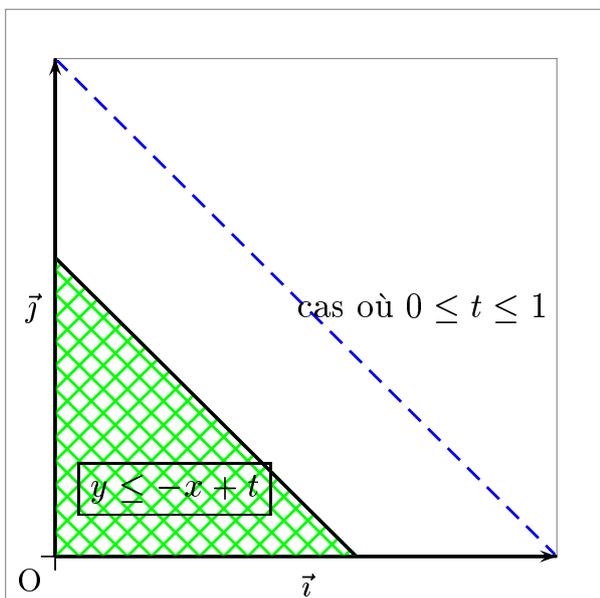
On pouvait aussi avoir une approche géométrique du problème :



En effet  $(X, Y)$  suit la loi uniforme sur le carré unité et la recherche de la probabilité

$P(X + Y \leq t)$  revient au calcul de l'aire de la trace sur le carré unité du demi-plan d'équation  $x + y \leq t$ , soit un triangle rectangle isocèle si  $0 \leq t \leq 1$  et la réunion du triangle rectangle isocèle de côté 1 avec un trapèze isocèle si  $1 \leq t \leq 2$ . On obtient ainsi la fonction de répartition par des calculs d'aire de triangles à la portée d'un collégien (les cas  $t < 0$  et  $t > 1$  étant évidents).

Les deux figures qui suivent illustrent les cas  $0 \leq t \leq 1$  et  $1 \leq t \leq 2$



Ce qui donne :

$$\text{Si } 0 \leq z \leq 1, F(t) = P(X + Y \leq t) = \frac{t^2}{2}$$

$$\text{Si } 1 \leq t \leq 2, F(t) = -1 + 2t - \frac{t^2}{2}$$

On retrouve ainsi la densité de probabilité de la variable  $X + Y$  en dérivant la fonction  $F$  par morceaux.

### 3.3 Et au bac alors ?

Je vous soumetts une partie<sup>10</sup> de l'exercice 1 donné en septembre 2017 en Métropole :

Dîtes-moi ce que vous en pensez maintenant que vous avez lu l'ensemble de ce qui précède 😊.

#### EXERCICE 1

6 points

Commun à tous les candidats

#### Partie A

On considère la suite  $(u_n)$  définie pour tout entier naturel  $n$  par :

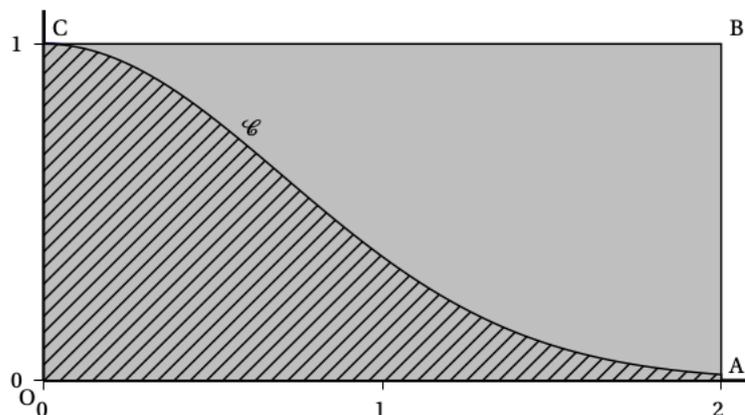
$$u_n = \int_0^n e^{-x^2} dx.$$

On ne cherchera pas à calculer  $u_n$  en fonction de  $n$ .

1. a. Montrer que la suite  $(u_n)$  est croissante.  
b. Démontrer que pour tout réel  $x \geq 0$ , on a :  $-x^2 \leq -2x + 1$ , puis :  
 $e^{-x^2} \leq e^{-2x+1}$ .  
En déduire que pour tout entier naturel  $n$ , on a :  $u_n < \frac{e}{2}$ .  
c. Démontrer que la suite  $(u_n)$  est convergente. On ne cherchera pas à calculer sa limite.
2. Dans cette question, on se propose d'obtenir une valeur approchée de  $u_2$ .

Dans le repère orthonormé  $(O ; \vec{i}, \vec{j})$  ci-dessous, on a tracé la courbe  $\mathcal{C}_f$  représentative de la fonction  $f$  définie sur l'intervalle  $[0; 2]$  par  $f(x) = e^{-x^2}$ , et le rectangle OABC où A(2; 0), B(2; 1) et C(0; 1).

On a hachuré le domaine  $\mathcal{D}$  compris entre la courbe  $\mathcal{C}_f$ , l'axe des abscisses, l'axe des ordonnées et la droite d'équation  $x = 2$ .



10. Le source L<sup>A</sup>T<sub>E</sub>X provient de l'APMEP

On considère l'expérience aléatoire consistant à choisir un point  $M$  au hasard à l'intérieur du rectangle OABC.

On admet que la probabilité  $p$  que ce point appartienne au domaine est :  $p = \frac{\text{aire de } \mathcal{D}}{\text{aire de OABC}}$ .

a. Justifier que  $u_2 = 2p$ .

b. On considère l'algorithme suivant :

L1	<b>Variables</b> : $N, C$ nombres entiers; $X, Y, F$ nombres réels
L2	<b>Entrée</b> : Saisir $N$
L3	<b>Initialisation</b> : $C$ prend la valeur 0
L4	<b>Traitement</b> :
L5	Pour $k$ variant de 1 à $N$
L6	$X$ prend la valeur d'un nombre aléatoire entre 0 et 2
L7	$Y$ prend la valeur d'un nombre aléatoire entre 0 et 1
L8	Si $Y \leq e^{-x^2}$ alors
L9	$C$ prend la valeur $C + 1$
L10	Fin si
L11	Fin pour
L12	Afficher $C$
L13	$F$ prend la valeur $C/N$
L14	Afficher $F$

i. Que permet de tester la condition de la ligne L8 concernant la position du point  $M(X ; Y)$  ?

ii. Interpréter la valeur  $F$  affichée par cet algorithme.

iii. Que peut-on conjecturer sur la valeur de  $F$  lorsque  $N$  devient très grand ?

c. En faisant fonctionner cet algorithme pour  $N = 10^6$ , on obtient  $C = 441\,138$ .

On admet dans ce cas que la valeur  $F$  affichée par l'algorithme est une valeur approchée de la probabilité  $p$  à  $10^{-3}$  près.

En déduire une valeur approchée de  $u_2$  à  $10^{-2}$  près.

## 4 Conclusions

- Premièrement, lorsque l'on tire au hasard un réel entre 0 et 1, le préalable indispensable sera de préciser **selon quelle loi**.
- Deuxièmement, on peut légitimement se poser la question de l'utilisation de la notion de modélisation.

Une modélisation n'a de sens que si l'on précise l'espace probabilisé associé.

Les exemples précédents nous montrent donc le caractère **indispensable** de la précision.

- Un modèle est forcément une description limitée et orientée de la réalité. Il est impossible de décrire de manière **absolue** la réalité à des fins descriptives ou prédictives.

Un modèle n'est pas conçu pour être vrai et représenter parfaitement la réalité, mais pour répondre à un usage.

*Cous le*

**G. Box, statisticien**

## 5 Remerciements

Je tiens à remercier Charles Suquet, professeur de mathématiques à l'université de Lille, dont les conseils et les nombreux ouvrages de cours sur les probabilités qu'il a mis à disposition de la communauté mathématique, enseignants et étudiants, ont été pour moi une source d'enrichissement constant.

Je me suis fortement inspiré en particulier du document suivant que vous pouvez consulter en cliquant sur le livre  .

J'adresse aussi un grand merci à Emmanuel Ostenne, membre du groupe ArCSin de l'IREM de Lille, pour ses différentes relectures.