

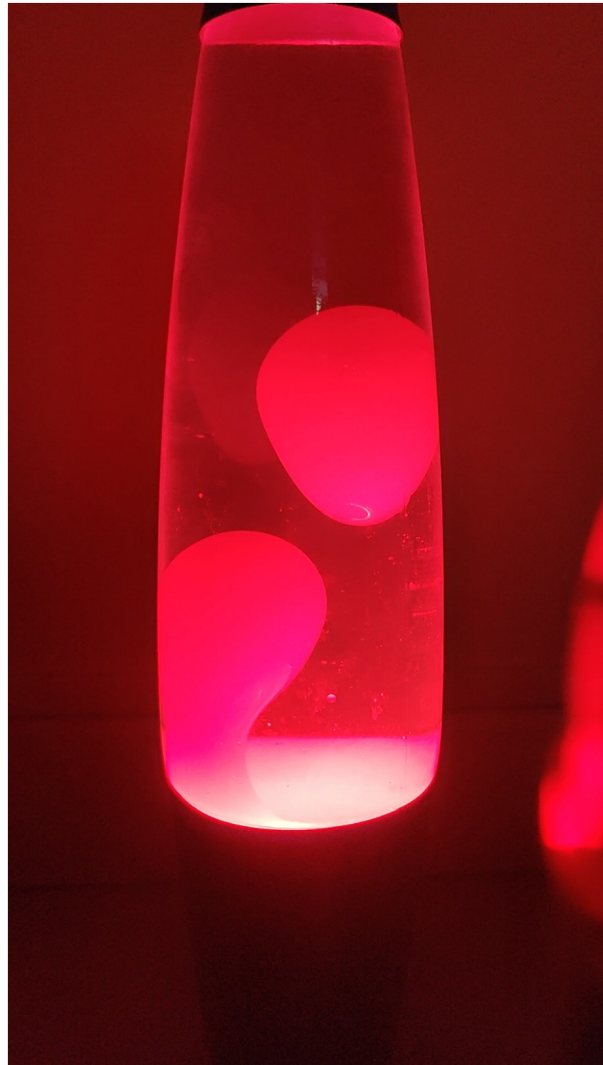
TIPE 2020 – 2021

LA LAMPE À LAVE GÉNÉRATEUR DE NOMBRES PSEUDO- ALÉATOIRES

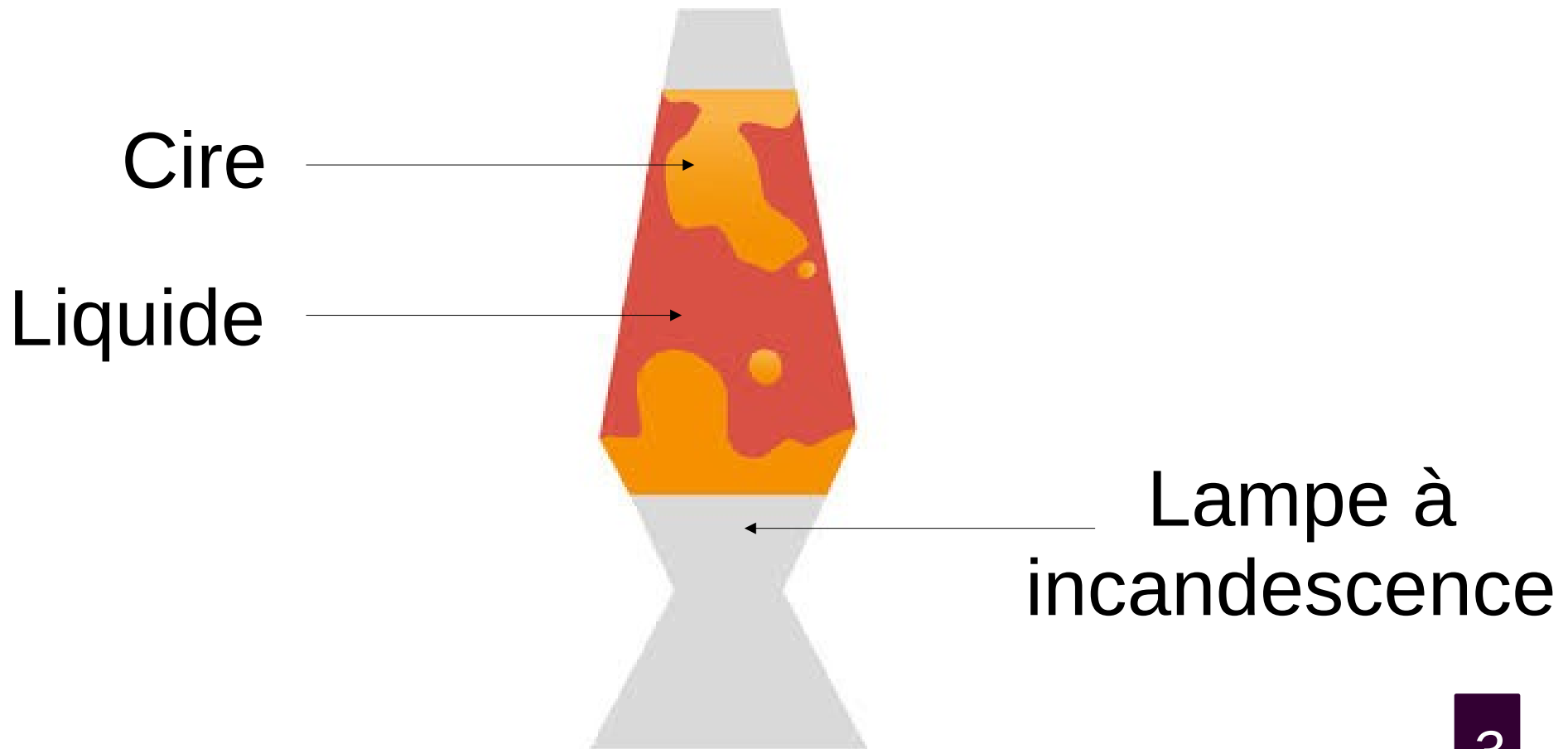
DUBOIS ALESSIA MP

N° D'INSCRIPTION : 29334

La lampe à lave



Fonctionnement de la lampe à lave



Sommaire

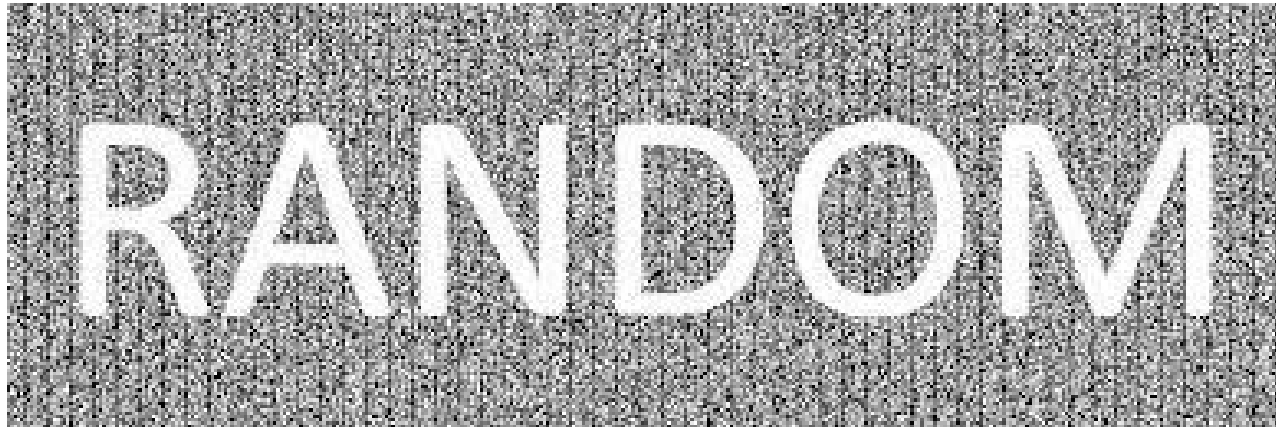
- 1) Modélisation
- 2) Résultats
- 3) Tests complémentaires
- 4) Efficacité

Définir l'aléatoire, une notion complexe




01581	36001	15892	57621	85239
02972	56177	87580	66794	48123
97022	65380	91304	32853	99729
67583	01277	77815	60558	75920
26935	56306	38710	77239	47139
21201	75983	35695	60517	14579
02628	26124	68322	01436	85994
93635	69404	76323	33459	70041
08984	81320	03226	60959	78246
04415	78662	28295	46513	92889
13070	18401	14382	48262	53177
53531	36891	29620	72532	47368
87733	74995	61843	88472	15736
47619	57452	92819	34401	48782
94060	67951	28895	79309	91897

Aléatoire
≠
Pseudo aléatoire



RANDOM



Qu'est-ce qu'un bon générateur de
nombres pseudo aléatoires ?

Objectifs

- Générer des nombres pseudo-aléatoires
- Vérifier le caractère aléatoire de cette suite

Différents générateurs

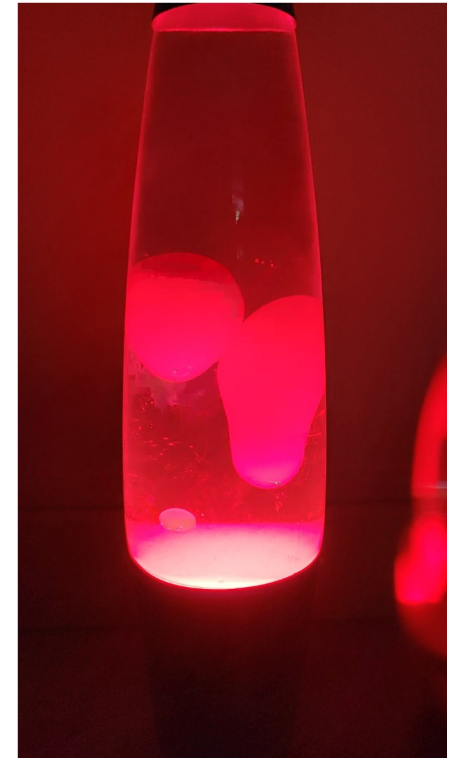
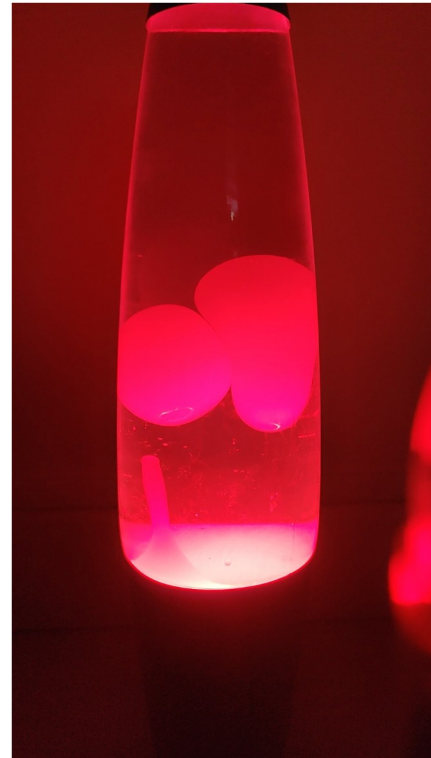
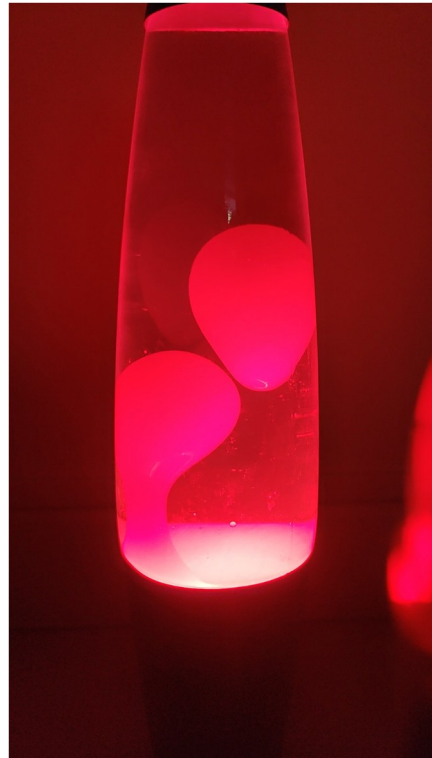
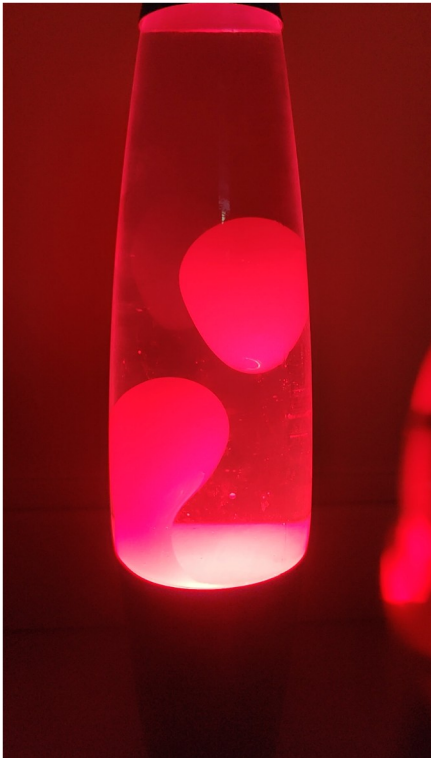
Phénomènes physiques :

- Lampe à lave
- Mécanique quantique
- Souffle direct dans un micro

Algorithmes :

- Générateur congruentiel linéaire
- Méthode de Fibonacci
- Méthode de Von Neumann

Sélection d'images : 1 / 50

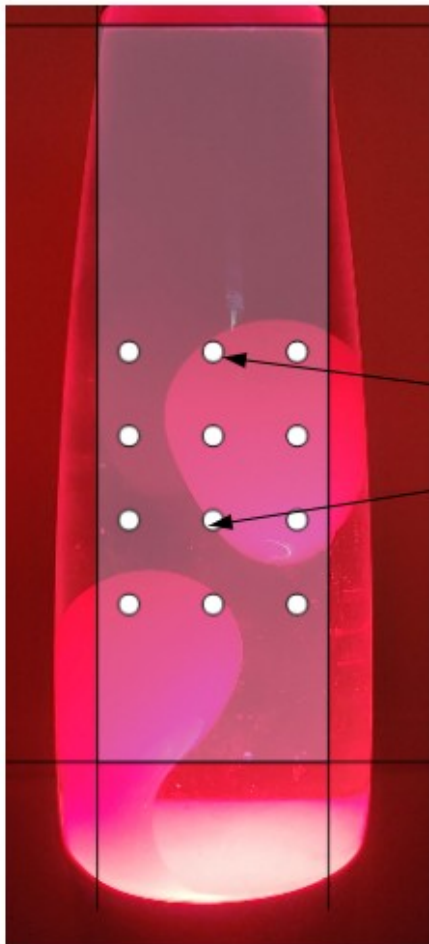


373 images

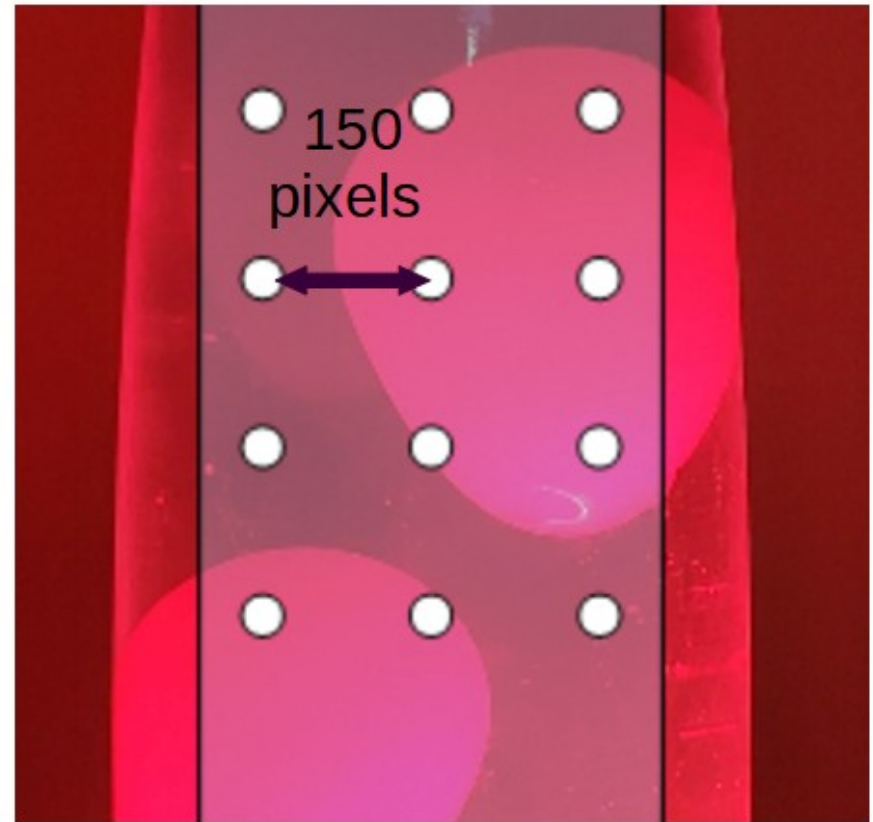
Zone d'étude

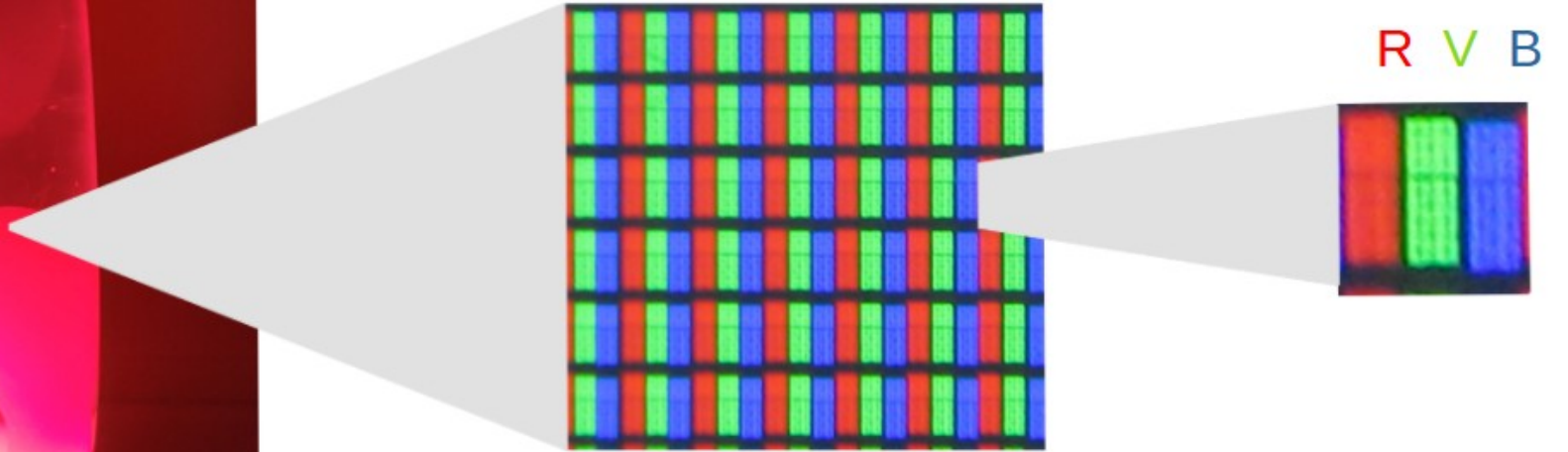
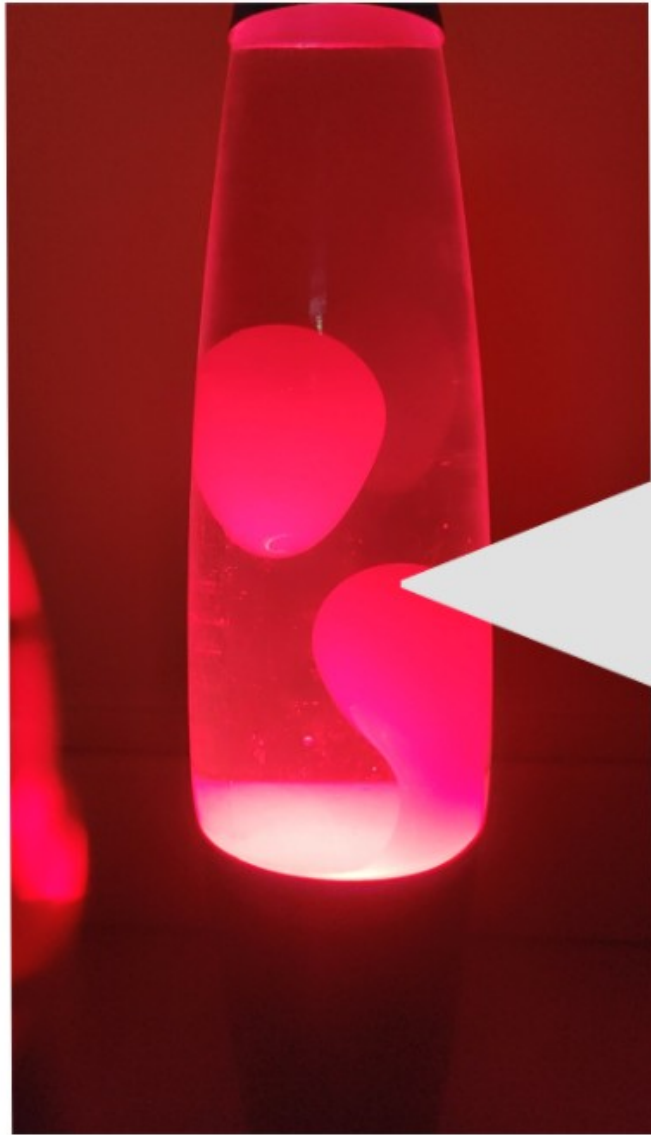


Points étudiés



Exemple
de
points
étudiés





Premiers choix

Choix de la composante rouge

R V B

Valeur de la composante rouge :

0 . 4 4 7 0 5 **8** 8 2 6 6

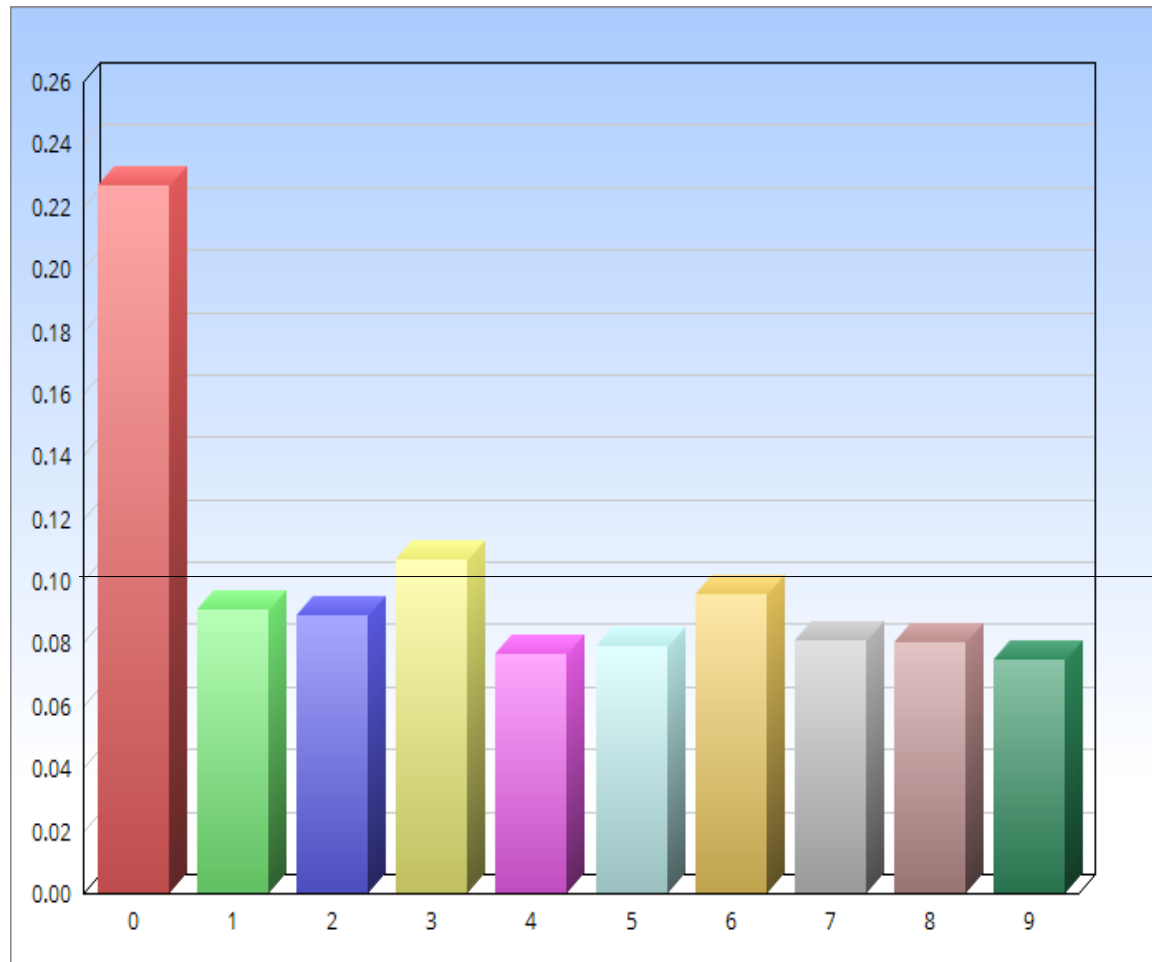
8^{ème} décimale

Premiers résultats

2, 1, 4, 3, 0, 0, 4, 9, 7, 0, 3, 0, 7, 2, 0, 0, 6, 7, 2, 0, 1, 6, 1, 2, 2, 0, 0
, 5, 0, 1, 7, 5, 3, 8, 2, 3, 2, 2, 6, 0, 0, 3, 7, 1, 1, 6, 6, 4, 5, 6, 4, 6, 6,
0, 5, 1, 6, 4, 1, 3, 0, 8, 6, 5, 5, 3, 4, 4, 3, 0, 0, 1, 9, 9, 1, 1, 0, 3, 0, 3,
4, 6, 7, 8, 7, 6, 0, 0, 1, 1, 1, 8, 3, 8, 1, 0, 0, 0, 6, 8, 8, 0, 3, 9, 0, 8, 7
, 4, 2, 6, 9, 9, 1, 6, 0, 0, 1, 2, 2, 6, 9, 6, 1, 3, 0, 0, 9, 6, 5, 6, 0, 0, 6,
5, 3, 5, 9, 8, 6, 4, 4, 5, 0, 9, 4, 1, 8, 6, 0, 7, 5, 0, 9, 4, 4, 2, 1, 0, 8, 9,
0, 5, 1, 5, 0, 9, 2, 5, 1, 3, 9, 3, 5, 1, 5, 0, 5, 3, 9, 0, 2, 7, 1, 2, 4, 2, 4
, 9, 9, 3, 1, 6, 0, 6, 9, 9, 7, 9, 6, 8, 9, 0, 0, 5, 7, 6, 6, 2, 7, 3, 5, 1, 6,
4, 1, 3, 6, 0, 5, 2, 6, 6, 1, 0, 0, 5, 5, 0, 6, 0, 1, 6, 2, 0, 4, 5, 6, 3, 7, 6,
4, 4, 3, 0, 0, 8, 1, 8, 0, 8, 3, 0, 0, 6, 4, 6, 0, 0, 9, 6, 9, 6, 8, 0, 9, 5, 2
, 0, 9, 2, 9, 3, 6, 7, 6, 2, 0, 6, 2, 6, 2, 7, 0, 7, 1, 3, 2, 0, 2, 5, 4, 6, 4,
5, 4, 1, 0, 2, 0, 1, 8, 3, 6, 8, 6, 0, 8, 2, 8, 2, 7, 1, 7, 5, 0, 6, 0, 8, 3, 1,
9, 3, 9, 0, 2, 7, 0, 6, 4, 6, 4, 7, 9, 7, 0, 3, 1, 0, 9, 0, 9, 0, 8, 4, 0, 5, 8
, 7, 0, 2, 0, 6, 1, 5, 9, 0, 3, 8, 7, 2, 5, 0, 8, 0, 7, 6, 4, 6, 0, 9, 8, 3, 0,
8, 7, 0, 2, 0, 1, 2, 7, 2, 3, 9, 1, 4, 0, 6, 0, 1, 0, 7, 3, 5, 9, 0, 2, 1, 9, 0

7 833 valeurs

Répartition



Valeur
théorique

Recherche et résolution du problème

Causes éventuelles du problème :

- Incertitude de mesures
- Choix de composante (R V B) à optimiser
- Choix de décimale à optimiser
- Choix de points sur l'image à optimiser

Correction

Choix de la composante bleue

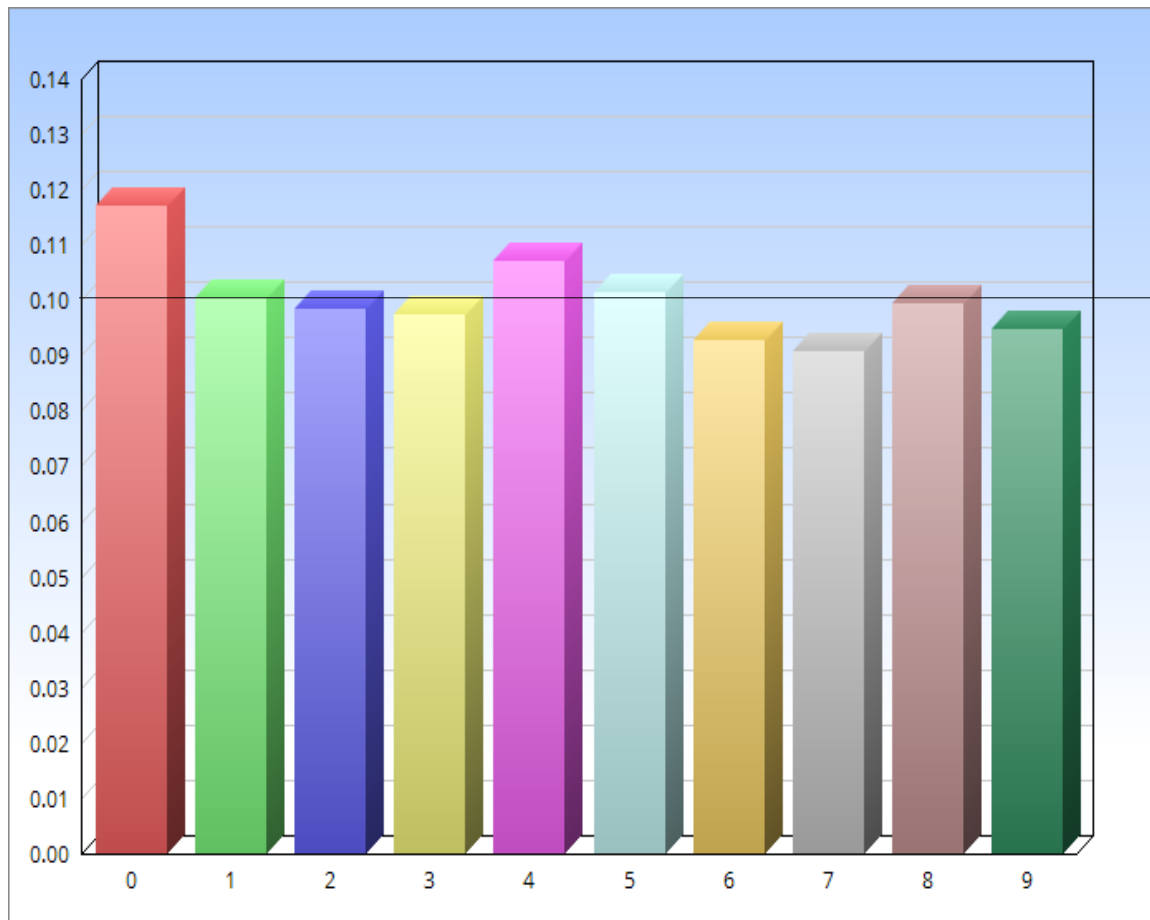
R V **B**

Valeur de la composante bleue :

0 . 0 4 7 0 **5** 8 8 2 4 4

5^{ème} décimale

Résultats / Interprétation



Valeur
théorique

Tests complémentaires

- Moyenne
- Écart type
- Moment d'ordre 3
- Monte Carlo

Moyenne

Liste de **7 833** valeurs

$$\mu = \sum_{k=0}^9 k \frac{N_k}{N}$$

Moyenne théorique : **4.5**

Moyenne expérimentale : **4.36**

Pourcentage d'erreur : **3,1 %**

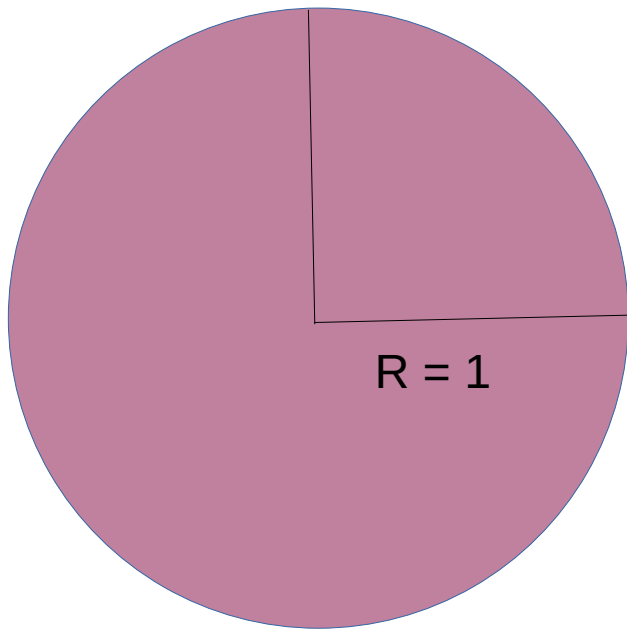
Écart type et moment d'ordre 3

$$\sigma = \sqrt{\mu_2}$$

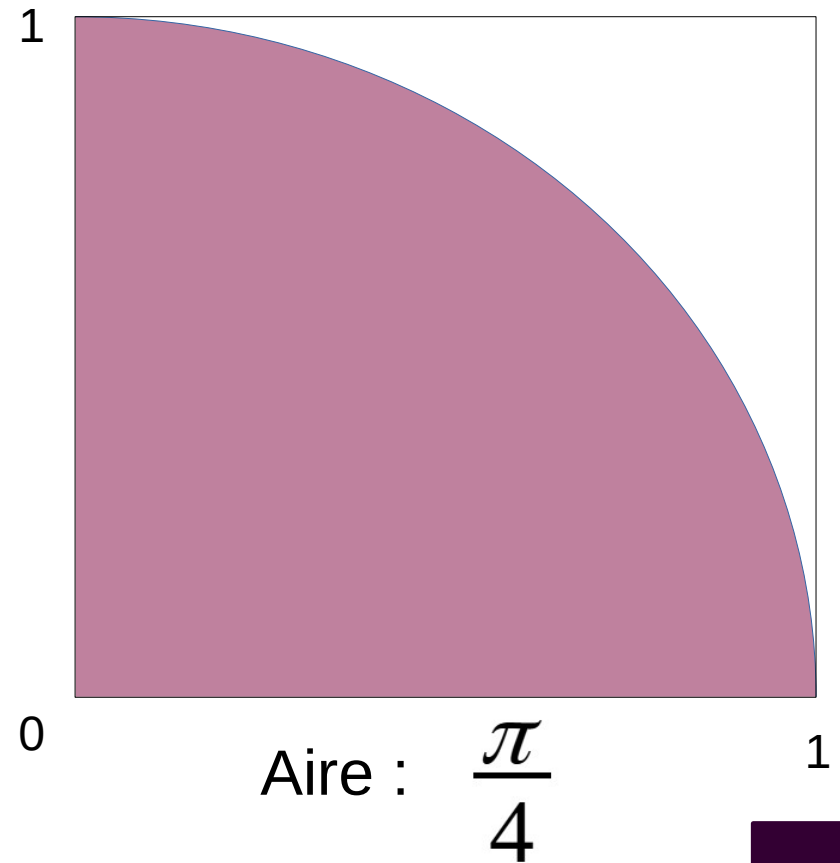
$$\mu_k = \mathbf{E} \left((X - \mathbf{E}(X))^k \right)$$

	Écart type	Moment d'ordre 3
Valeur théorique	2.87	0
Valeur expérimentale	2.89	1.29

Méthode de Monte Carlo



Aire : π



Aire : $\frac{\pi}{4}$

Création de nombres décimaux aléatoires

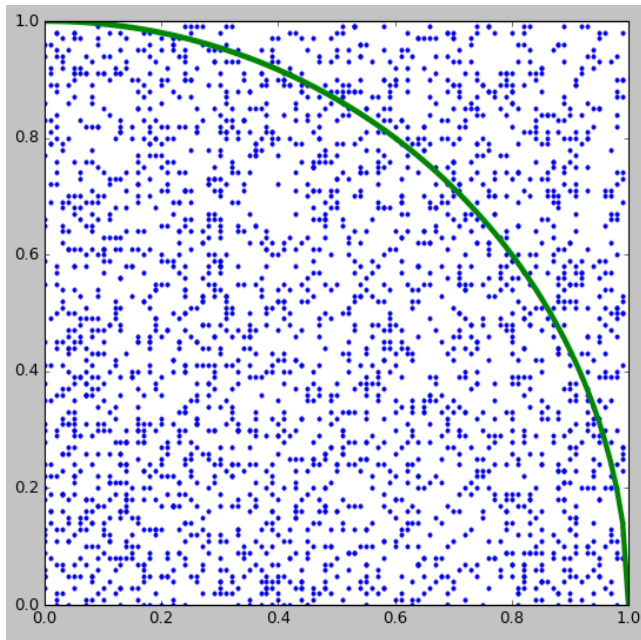
Exemple avec 5 décimales :

[5, 5, 2, 5, 9, 4, 2, 3, 9, 4, 8, 1, 2, 0, 5, 7, 3, 5, 7]

[0.55259 , 0.42394 , 0.81205 ...]

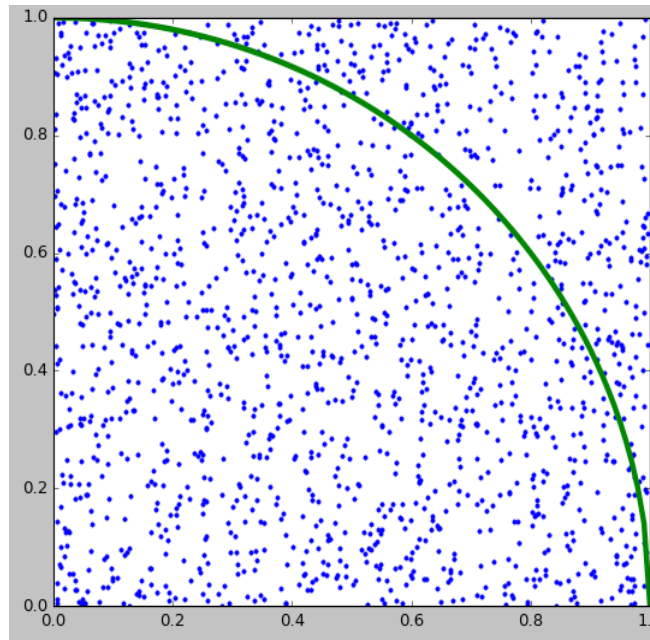
Résultats

2 décimales



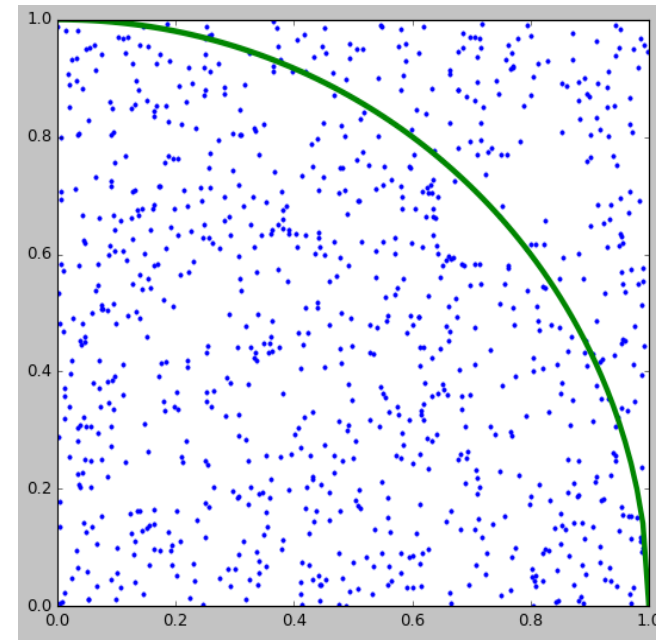
$\pi \sim 3.301$

5 décimales



$\pi \sim 3.145$

10 décimales



$\pi \sim 3.241$

Efficacité



7 833 nombres
en 47,90
secondes

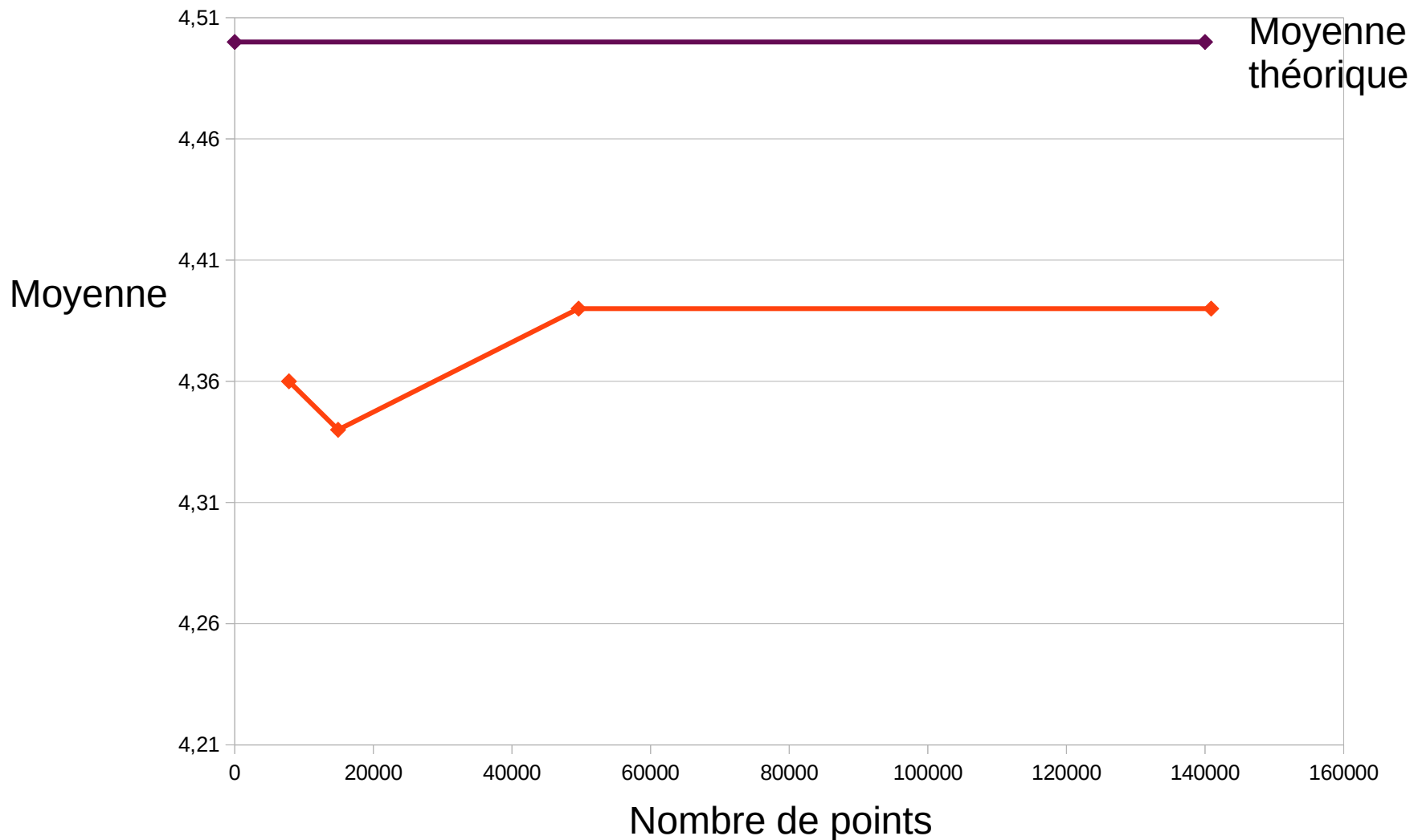


164 nombres
par seconde

Amélioration

	Espacement (en pixels)	Nombre de termes	Moyenne	Écart type	Moment d'ordre 3
Théorie			4.50	2.87	0
	150	7 833	4.36	2.89	1.29
	100	14 920	4.34	2.91	0.88
	50	49 609	4.39	2.91	1.62
	30	140 483	4.39	2.91	0.83

Évolution de la moyenne



Amélioration de l'efficacité

Rapprochement
des points



140 483 valeurs en
48,02 secondes



2 926
nombres par
seconde
(x 18)

Tentative d'amélioration

Réduction
du nombre
d'images



Augmentation
du nombre de
points par
image



Meilleure
efficacité

Maintien du
caractère
aléatoire

Conclusion

- Objectifs globalement atteints
- Caractéristiques d'un bon générateur :
 - Passer les tests
 - Moyenne
 - Écart type
 - Moments d'ordres supérieurs
 - Monte Carlo
 - Générer efficacement une liste de nombres

Améliorations

- Augmenter le nombre d'images
- Augmenter le nombre de points
- Faire d'autres tests

Annexe

- Création de la liste d'images
- Traitement des images
- Répartition
- Moyenne, écart type, moments
- Monte Carlo

Création de la liste d'images

```
def numeros () :  
    liste_num = []  
    for i in range (373) :  
        liste_num.append(1 + i*50)  
    return liste_num  
  
def liste_caractere (liste) :  
    for i in range (373) :  
        liste[i] = str(liste[i])  
    return liste
```

Création de la liste d'images

```
def chemin_images (liste):
    S = 'a\TIPE\a'
    S = S[1:5]
    liste[0] = "\"D:" + S + "E\Images lampe 2/scene0000" + liste[0] + ".png\","
    print(liste[0])
    liste[1] = "\"D:\TIPE\Images lampe 2/scene000" + liste[1] + ".png\","
    print(liste[1])
    for i in range (2,20) :
        liste[i] = "\"D:\TIPE\Images lampe 2/scene00" + liste[i] + ".png\","
        print(liste[i])
    for i in range (20,200) :
        liste[i] = "\"D:\TIPE\Images lampe 2/scene0" + liste[i] + ".png\","
        print(liste[i])
    for i in range (200,373) :
        liste[i] = "\"D:\TIPE\Images lampe 2/scene" + liste[i] + ".png\","

listeentiere = chemin_images (liste_caractere (numeros ()))
```

Liste des images

```
listeimages =["E:\TIPE\Images lampe 2/scene00001.png",  
"E:\TIPE\Images lampe 2/scene00051.png",  
"E:\TIPE\Images lampe 2/scene00101.png",  
"E:\TIPE\Images lampe 2/scene00151.png",  
"E:\TIPE\Images lampe 2/scene00201.png",  
"E:\TIPE\Images lampe 2/scene00251.png",  
"E:\TIPE\Images lampe 2/scene00301.png",  
"E:\TIPE\Images lampe 2/scene00351.png",  
"E:\TIPE\Images lampe 2/scene00401.png",  
"E:\TIPE\Images lampe 2/scene00451.png",  
"E:\TIPE\Images lampe 2/scene00501.png",  
"E:\TIPE\Images lampe 2/scene00551.png",  
"E:\TIPE\Images lampe 2/scene00601.png",  
"E:\TIPE\Images lampe 2/scene00651.png",  
"E:\TIPE\Images lampe 2/scene00701.png",  
"E:\TIPE\Images lampe 2/scene00751.png",  
"E:\TIPE\Images lampe 2/scene00801.png",  
"E:\TIPE\Images lampe 2/scene00851.png",  
"E:\TIPE\Images lampe 2/scene00901.png",  
"E:\TIPE\Images lampe 2/scene00951.png",  
"E:\TIPE\Images lampe 2/scene01001.png",  
"E:\TIPE\Images lampe 2/scene01051.png",  
"E:\TIPE\Images lampe 2/scene01101.png",  
"E:\TIPE\Images lampe 2/scene01151.png",  
"E:\TIPE\Images lampe 2/scene01201.png",  
"E:\TIPE\Images lampe 2/scene01251.png",  
"E:\TIPE\Images lampe 2/scene01301.png",  
"E:\TIPE\Images lampe 2/scene01351.png",
```

Traitement des images

```
def decimale (a,d) :  
    a = a*(10**d)  
    deci = a%10  
    return deci
```

Traitement des images

```
def traitement (image,d) :  
    img = plt.imread(image)  
    nombres = []  
    for i in range (350,700,150) :  
        for j in range (70,1000,150) :  
            nombres.append(m.floor(decimale(img[i,j,:][2],d)))  
    return nombres  
  
def nombresaleatoire (liste,d) :  
    t = time.time()  
    n = len (liste)  
    nombre = []  
    for i in range (n):  
        image = liste[i]  
        nombre = nombre + traitement(image,d)  
    return nombre, time.time()-t
```

Répartition

```
def repartition (liste) :
    s = [0]*10
    p = [0]*10
    n = len(liste)
    for k in range (n) :
        if liste[k]==0 :
            s[0] += 1
        elif liste[k]==1 :
            s[1] += 1
        elif liste[k]==2 :
            s[2] += 1
        elif liste[k]==3 :
            s[3] += 1
        elif liste[k]==4 :
            s[4] += 1
        elif liste[k]==5 :
            s[5] += 1
        elif liste[k]==6 :
            s[6] += 1
        elif liste[k]==7 :
            s[7] += 1
        elif liste[k]==8 :
            s[8] += 1
        else :
            s[9] += 1
    for k in range (10) :
        p[k] = s[k]/n
    return p
```

Moyenne

```
def esperance (liste) :  
    n = len(liste)  
    e = 0  
    for i in range (n) :  
        e = e + i*liste[i]  
    return e
```


Écart type

```
def variance (liste) :  
    n = len(liste)  
    e = 0  
    for i in range (n) :  
        e = e + (i**2)*liste[i]  
    return e - (esperance(liste))**2  
  
def ecart_type (liste) :  
    return (variance (liste))**(1/2)
```

Moment d'ordre 3

```
def moment3(liste) :  
    n = len(liste)  
    e1 = esperance(liste)  
    e2 = 0  
    e3 = 0  
    for i in range (n) :  
        e2 = e2 + (i**2)*liste[i]  
        e3 = e3 + (i**3)*liste[i]  
    return e3 - 3*e1*e2 + 2*(e1**3)
```

Monte Carlo

```
def concat (liste) :
    n = len(liste)
    nombre = ''
    for i in range (n):
        d = str(liste[i])
        nombre = nombre + d
    nombre = float(nombre)
    return nombre /10**n

def conversion (liste , a):
    n = len(liste)
    N = n-n//a
    m = N//a
    C = [0]*m
    for i in range (m) :
        p = i*a
        temp = liste[ p : p+a : 1]
        C[i] = concat(temp)
    return C
```

Monte Carlo

```
def monte_carlo (liste,a) :  
    l = conversion (liste,a)  
    n = len(l)  
    if (n//2 == 0) :  
        abscisse = l[::2]  
        ordonne = l[1::2]  
    else :  
        abscisse = l[:n-1:2]  
        ordonne = l[1::2]  
    total = 0  
    S = 0  
    for i in range (len(abscisse)):  
        if (abscisse[i]**2 + ordonne[i]**2)**(1/2) < 1 :  
            S = S + 1  
        total = total + 1  
    pi = S/total * 4  
    return pi
```