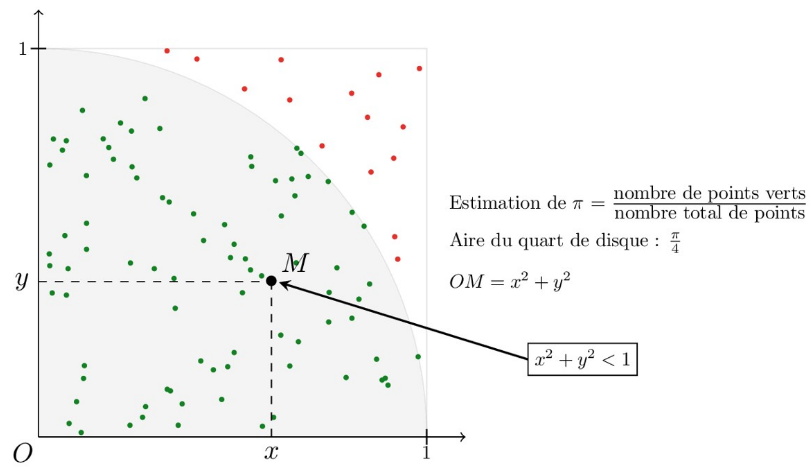
TP3 : SIMULATION DE MONTE CARLO ET INTERVALLE DE CONFIANCE



Koccou MINAGNIKPO

Février 2023

Table des matières

Table des figures

1 Introduction

Dans ce TP, nous aurons pour objectifs de tester l'approximation de la valeur numérique de pi(π)avec la méthode de Monte Carlo. Et pour réaliser cela, nous aurons à utiliser un bon générateur de nombres aléatoires tels que Mersenne Twister proposé par Matsumoto et découvert au TP 2. Ensuite nous terminerons par une examen de l'intervalle de confiance pour quantifier l'incertitude de cette estimation de π.

2 Calcul de PI (π) par la methode de Monte Carlo

2.1 Valeur Pi (π)

Pi (π) est une constante mathématique qui représente le rapport entre la circonférence d'un cercle et son diamètre. C'est une valeur irrationnelle, ce qui signifie qu'elle ne peut pas être exprimée comme une fraction exacte, et qu'elle possède une infinité de décimales. Notons que la valeur de Pi est =3.1415926535..... et La methode Monte Carlo nous aidera a simuler cette valeuur de Pi .

2.2 Methode de Monte Carlo

La méthode de Monte Carlo est une technique de simulation numérique qui peut être utilisée pour estimer la valeur de Pi. L'idée de base est de générer des points aléatoires dans un carré et de compter combien de ces points se trouvent à l'intérieur d'un cercle inscrit dans le carré. En utilisant la proportion de points qui se trouvent à l'intérieur du cercle par rapport au nombre total de points, il est possible d'estimer la valeur de Pi.

2.3 Application de la Methode de Monte Carlo

Pour cette partie, nous avons testé la methode de **Monte Carlo** pour estimer la valeur de Pi en utilisant un nombre donné de points.
Pour ce faire , On a testé notre code ecris avec les différents nombres de points, à savoir 1000, 1000000 et 1000000000.

l'algorithme suivi est le suivant :

```
Variables nbr_points, i, inDisk, xr, yr, Pi en Entier ou Flottant
Début
 inDisk ← 0
Pour i ← 1 à nbr_points
xr ← genrand_real1()
yr ← genrand_real1()
Si xr * xr + yr * yr <= 1 Alors i
nDisk ← inDisk + 1 FinSi
i Suivant
Pi ← (inDisk / nbr_points) * 4
Retourner Pi
Fin
```

- Si on teste pour 1000 points : la valeur de Pi −> 3.124000
- Si on teste pour 1000000 points : la valeur de Pi −> 3.144760
- Si on teste pour 1000000000 points : la valeur de Pi −> 3.141546

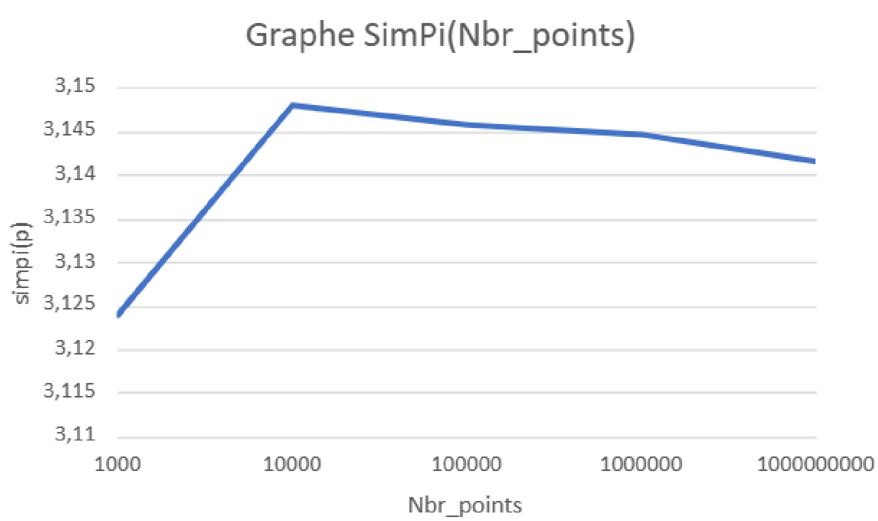


Figure 1 : Graphe montrant l'estimation de Pi

Constat : En augmentant le nombre de points générés, on obtient une estimation plus précise de la proportion de points à l'intérieur du cercle, et donc une estimation plus précise de la valeur de Pi **(avec** $\pi = 3.141592$**)** . Cependant, il faut noter que l'augmentation du nombre de points générés entraîne également une augmentation du temps de calcul nécessaire pour exécuter l'algorithme.

3 Calcul d'expériences indépendantes et obtention de PI moyen

Dans cette partie, il nous est demandé de réaliser plusieurs expériences indépendantes pour estimer la valeur de PI en utilisant la fonction **simPi**. Pour chaque expérience, on utilise le même générateur de nombres pseudo-aléatoires pour générer les nombres aléatoires nécessaires à la simulation de PI. Ensuite, on réalise entre 10 et 40 expériences pour chaque valeur de nbPoints, qui peut prendre les valeurs 1000, 1000000 et 1000000000.
Ainsi pour chaque expérience, on stocke le résultat de l'estimation de PI dans un tableau. Une fois que toutes les expériences sont terminées, on calcule la moyenne des estimations de PI pour chaque valeur de nbPoints. Enfin, on calcule la différence entre la moyenne de nos estimations de PI et la valeur de PI exacte stockée dans la constante M_PI de la bibliothèque mathématique C. Cette différence doit être calculée à la fois en termes d'erreur absolue et d'erreur relative.

3.1 Algorithme et Résultats

L'algorithme utlisé pour realiser cela est le suivant :

```
Variables nbr_points, i, inDisk, xr, yr, Pi en entier ou flottant :
Début
 Pour chaque expérience :
inDisk ← 0
Pour i de 1 à nbr_points
xr ← générer un nombre aléatoire entre 0 et 1
 yr ← générer un nombre aléatoire entre 0 et 1
Si xr * xr + yr * yr <= 1 Alors
 inDisk ← inDisk + 1
FinSi
FinPour Pi ← (inDisk / nbr_points) * 4
Stocker la valeur de Pi dans le tableau Tab_exp à l'indice correspondant à l'expérience en cours
FinPour pi_means ← 0
 Pour chaque expérience stockée dans le tableau Tab_exp :
pi_means ← pi_means + Tab_exp[i]
FinPour
pi_means ← pi_means / nb_exp
erreur_absolue ← valeur absolue de (pi_means - pi)
erreur_rel ← erreur_absolue / pi
Afficher "PI moyen = " + pi_means
Afficher "Erreur absolue = " + erreur_absolue
Afficher "Erreur relative = " + erreur_rel
Fin
```

Voici les resultats obtenues pour chaque expérience realisé :

*** Test Experience_moyenne(Tab_ep, nb_exp) ***

Pour nb_exp=10 :

Nbr_point= 1000

PI moyen= 3.148000

Erreur absolue= 0.006407

Erreur relative= 0.002040

Nbr_point = 1000000

PI moyen= 3.141286

Erreur absolue= 0.000307

Erreur relative 0.000098

Nbr_point= 1000000000

PI moyen= 3.141563

Erreur absolue= 0.000030

Erreur relative= 0.000010

*** Test Experience_moyenne(Tab_ep, nb_exp) ***

Pour nb_exp= 20 :

Nbr_point= 1000

PI moyen= 3.140200

Erreur absolue= 0.001393

Erreur relative= 0.000443

Nbr_point = 1000000

PI moyen= 3.141137

Erreur absolue= 0.000456

Erreur relative 0.000145

Nbr_point= 1000000000

PI moyen= 3.141586

Erreur absolue= 0.000007

Erreur relative= 0.000002

*** Test Experience_moyenne(Tab_ep, nb_exp) ***

Pour nb_exp= 30 :

Nbr_point= 1000

PI moyen= 3.139733

Erreur absolue= 0.001859

Erreur relative= 0.000592

Nbr_point = 1000000

PI moyen= 3.141142

Erreur absolue= 0.000450

Erreur relative = 0.000143

Nbr_point= 1000000000

PI moyen= 3.141584

Erreur absolue= 0.000008

Erreur relative= 0.000003

*** Test Experience_moyenne(Tab_ep, nb_exp) ***

Pour nb_exp= 40 :

Nbr_point= 1000

PI moyen= 3.142400

Erreur absolue= 0.000807

Erreur relative= 0.000257

Nbr_point = 1000000

PI moyen= 3.141186

Erreur absolue= 0.000406

Erreur relative = 0.000129

Nbr_point= 1000000000

PI moyen= 3.141586

Erreur absolue= 0.000006

Erreur relative= 0.000002

4 La Simulation de Monte Carlo et l'intervalles de confiance

Comme nous l'avons fait remarqué au debut, **La méthode de Monte Carlo** est une méthode de simulation numérique qui utilise des échantillons aléatoires pour estimer des quantités inconnues. Dans le cas de l'estimation de PI, la méthode de Monte Carlo consiste à générer des points aléatoires dans un carré unité et à estimer la valeur de PI à partir de la proportion de points qui tombent dans un cercle inscrit dans le carré unité. Lorsqu' on utilise cette méthode pour estimer la valeur de PI, on génère un grand nombre de points aléatoires. Cependant, il est important de comprendre que les résultats de la simulation ne seront pas exacts à 100%. Il y aura toujours une certaine incertitude associée à l'estimation de PI à partir d'un échantillon de points aléatoires. Ainsi en déterminant l'intervalle de confiance, on peut évaluer la précision de l'estimation de PI et avoir une idée de la plage de valeurs probables dans laquelle se situe la vraie valeur de PI avec un niveau de confiance donné.

4.1 Rapel du Principe de l'intervalle de confiance

L'intervalle de confiance est centré sur la moyenne arithmétique, on calcule donc simplement ce qu'on appelle le rayon de confiance (marge d'erreur). Voici l'hypothèse statistique théorique qui est utilisée pour obtenir ce rayon. Si Xi a des distributions gaussiennes indépendantes identiques avec une moyenne théorique μ et une variance de σ^2 , alors la variable aléatoire suivante :

$$T(n) = \frac{\bar{X}(n) - \mu}{\sqrt{\frac{S^2(n)}{n}}}$$

est distribuée selon une loi de Student à n-1 degrés de liberté.

$$S^2(n) = \frac{\sum_{i=1}^{n} [X_i - \overline{X_n}]^2}{n-1}$$

$S^2(n)$ est une estimation sans biais de la variance. Comme nous ne connaissons pas l'écart-type théorique σ, nous estimons la variance avec les résultats que nous avons et utilisons donc cette approximation. On calcule $S^2(n)$ et on l'utlise dans la formule ci-dessous pour obtenir le rayon de confiance (marge d'erreur) au niveau 1-α :

$$R = T_{n-1,1-\frac{\alpha}{2}} \times \sqrt{\frac{S^2(n)}{n}}$$

William Sealy Gosset a introduit Student pour corriger le fait que nous n'utilisons qu'une estimation de σ et non sa vraie valeur.
Le tableau ci-dessous donne la valeur de $T_{n-1,1-\frac{\alpha}{2}}$ pour $\alpha = 0,01$ qui nous sera utile pour un intervalle de confiance à 99% :

α / ddl	0,90	0,50	0,30	0,20	0,10	0,05	0,02	0,01	0,001
1	0,158	1,000	1,963	3,078	6,314	12,706	31,821	63,656	636,578
2	0,142	0,816	1,386	1,886	2,920	4,303	6,965	9,925	31,600
3	0,137	0,765	1,250	1,638	2,353	3,182	4,541	5,841	12,924
4	0,134	0,741	1,190	1,533	2,132	2,776	3,747	4,604	8,610
5	0,132	0,727	1,156	1,476	2,015	2,571	3,365	4,032	6,869
6	0,131	0,718	1,134	1,440	1,943	2,447	3,143	3,707	5,959
7	0,130	0,711	1,119	1,415	1,895	2,365	2,998	3,499	5,408
8	0,130	0,706	1,108	1,397	1,860	2,306	2,896	3,355	5,041
9	0,129	0,703	1,100	1,383	1,833	2,262	2,821	3,250	4,781
10	0,129	0,700	1,093	1,372	1,812	2,228	2,764	3,169	4,587
11	0,129	0,697	1,088	1,363	1,796	2,201	2,718	3,106	4,437
12	0,128	0,695	1,083	1,356	1,782	2,179	2,681	3,055	4,318
13	0,128	0,694	1,079	1,350	1,771	2,160	2,650	3,012	4,221
14	0,128	0,692	1,076	1,345	1,761	2,145	2,624	2,977	4,140
15	0,128	0,691	1,074	1,341	1,753	2,131	2,602	2,947	4,073
16	0,128	0,690	1,071	1,337	1,746	2,120	2,583	2,921	4,015
17	0,128	0,689	1,069	1,333	1,740	2,110	2,567	2,898	3,965
18	0,127	0,688	1,067	1,330	1,734	2,101	2,552	2,878	3,922
19	0,127	0,688	1,066	1,328	1,729	2,093	2,539	2,861	3,883
20	0,127	0,687	1,064	1,325	1,725	2,086	2,528	2,845	3,850
21	0,127	0,686	1,063	1,323	1,721	2,080	2,518	2,831	3,819
22	0,127	0,686	1,061	1,321	1,717	2,074	2,508	2,819	3,792
23	0,127	0,685	1,060	1,319	1,714	2,069	2,500	2,807	3,768
24	0,127	0,685	1,059	1,318	1,711	2,064	2,492	2,797	3,745
25	0,127	0,684	1,058	1,316	1,708	2,060	2,485	2,787	3,725
26	0,127	0,684	1,058	1,315	1,706	2,056	2,479	2,779	3,707
27	0,127	0,684	1,057	1,314	1,703	2,052	2,473	2,771	3,689
28	0,127	0,683	1,056	1,313	1,701	2,048	2,467	2,763	3,674
29	0,127	0,683	1,055	1,311	1,699	2,045	2,462	2,756	3,660
30	0,127	0,683	1,055	1,310	1,697	2,042	2,457	2,750	3,646
40	0,126	0,681	1,050	1,303	1,684	2,021	2,423	2,704	3,551
80	0,126	0,678	1,043	1,292	1,664	1,990	2,374	2,639	3,416
120	0,126	0,677	1,041	1,289	1,658	1,980	2,358	2,617	3,373
∞	0,126	0,675	1,037	1,282	1,645	1,960	2,327	2,577	3,293

Figure2 : Loi de Student pour α= 0.01

La formule que nous utiliserons pour definir l'intervalle est la suivante :

$$[\bar{X} - R, \bar{X} + R]$$

ou $\bar{X} - R$ est la limite inférieure de l'intervalle et $\bar{X} + R$ est la limite supérieure de l'intervalle.

4.2 Algorithme utlisé et resultats des expériences

En suivant le principe de l'intervalle de confiance énoncé , L'algorithme suivant a été elaboré pour realiser nos experiences :

```
*** Intervalle_confiance(Tab_ep, nb_exp) ***
 Variables :
 Tab_exp : tableau contenant les résultats de toutes les expériences
 nb_exp : nombre total d'expériences réalisées
 Début
// Calcul de la moyenne des résultats de toutes les expériences
X_bar ← 0
Pour i de 0 à nb_exp-1
X_bar ← X_bar + Tab_exp[i]
X_bar ← X_bar / nb_exp
// Calcul de la variance
s ← 0
Pour i de 0 à nb_exp-1
s ← s + (Tab_exp[i] - X_bar)^2
s ← s / (nb_exp - 1)
// Valeur de t pour alpha=0.01 et n = nb_exp
Si nb_exp = 30 Alors
t ← 2.750
Sinon Si
   Afficher "Aucune valeur de t n'est définie pour ce nombre d'observations"
Retourner
FinSi
// Calcul de l'intervalle de confiance
R ← t * sqrt(s / nb_exp)
IC_lower ← X_bar - R
IC_upper ← X_bar + R
// Affichage des résultats
Afficher "Intervalle de confiance : [", IC_lower, ", ", IC_upper, "]"
```

Pour nos différentes test realisés , voici les resultats obtenus : :

*** Intervalle_confiance(Tab_ep, nb_exp) ***

Pour nb_exp= 10

t= 3.169

Nbr_point= 1000

Intervalle de confiance= [3.1067, 3.1892]

Nbr_point= 1000000

Intervalle de confiance=[3.1395, 3.1430]

Nbr_point= 1000000000

Intervalle de confiance= [3.141526 , 3.141599]

*** Intervalle_confiance(Tab_ep, nb_exp) ***

Pour nb_exp= 20

t= 2.845

Nbr_point= 1000

Intervalle de confiance=[3.1111 , 3.1692]

Nbr_point= 1000000

Intervalle de confiance=[3.1401 , 3.1421]

Nbr_point= 1000000000

Intervalle de confiance= [3.141558 , 3.141614]

*** Intervalle_confiance(Tab_ep, nb_exp) ***

Pour nb_exp= 30

t= 2.750

Nbr_point= 1000

Intervalle de confiance=[3.1172, 3.1622]

Nbr_point= 1000000

Intervalle de confiance=[3.1402, 3.1420]

Nbr_point= 1000000000

Intervalle de confiance=[3.141558 , 3.141610]

*** Intervalle_confiance(Tab_ep, nb_exp) ***

Pour nb_exp= 40

t= 2.704

Nbr_point= 1000

Intervalle de confiance= [3.1218, 3.1629]

Nbr_point= 1000000

Intervalle de confiance=[3.1404, 3.1419]

Nbr_point= 1000000000

Intervalle de confiance=[3.141563 , 3.141609]

Constat : L'intervalle de confiance calculé dépend de la valeur de t utilisée, qui est choisie en fonction de la taille de l'échantillon (nombre d'expériences). Pour nos test,les quatre valeurs de t qui nous sont fournies sont basé sur la distribution de Student.On remarque que plus le nombre d'échantillons est grand, plus la distribution se rapproche d'une distribution normale. Cela signifie que pour une taille d'échantillon plus petite, une plus grande incertitude est associée à l'estimation de la moyenne de Pi et donc, un intervalle de confiance plus large est nécessaire pour atteindre un niveau de confiance donné. Selon les resultats, pour n=30, et pour t choisie a 2.75, il est suffisant pour nous d'atteindre un niveau de confiance de 99% avec un alpha de $0,01$

5 Conclusion

Au vu des expériences realisées dans notre TP3 concernant la simulation de Monte carlo et l'intervalle de confiance, il en ressort que L'expérience consistant à estimer la valeur de Pi en utilisant la méthode de Monte-Carlo, avec un nombre croissant de points, permet d'obtenir une approximation de plus en plus précise de la vraie valeur de Pi. En effet, plus le nombre de points utilisés dans l'expérience est grand, plus la précision de l'estimation de Pi est élevée.
Et quant à L'intervalle de confiance autour de la moyenne Pi, il est également intéressant à considérer. Il permet de déterminer la marge d'erreur possible dans l'estimation de Pi. Plus l'intervalle de confiance est petit, plus la précision de l'estimation est élevée.
En conclusion, cette méthode de **Monte-Carlo** permet d'obtenir une estimation de la valeur de Pi avec une précision accrue en utilisant un nombre croissant de points, et permet de déterminer la marge d'erreur possible grâce à **l'intervalle de confiance** autour de la moyenne Pi.