Compte rendu TP - B3246

Baptiste PAULETTO & Louis UNG

9 et 17 mai 2019

Partie 1 : Tests de générateurs pseudos aléatoires

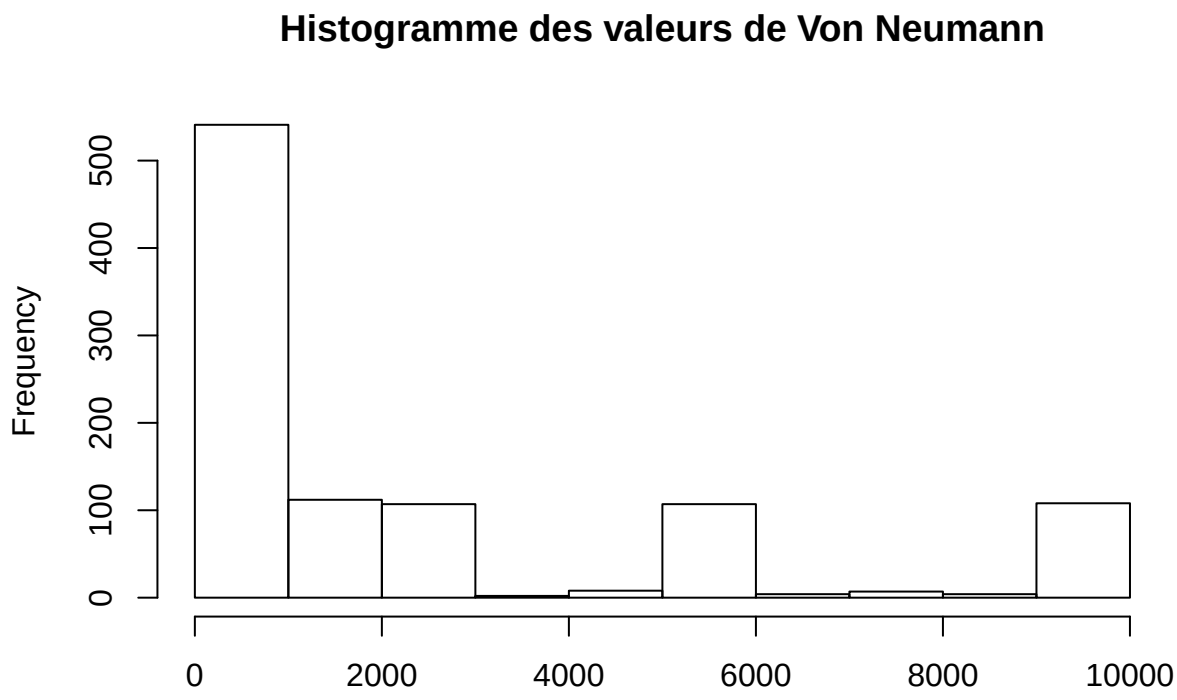
Test visuel

Pour ce premier test, il sera réalisé sur une séquence de 1000 valeurs.

Question 2.1 :

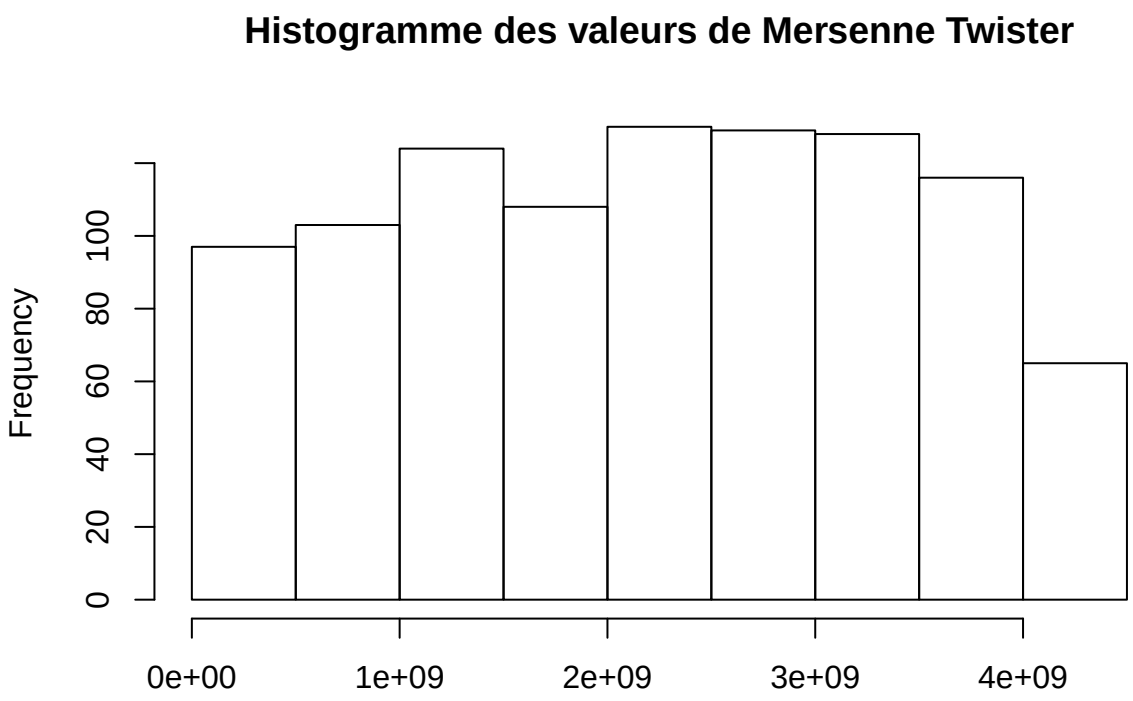
– Von Neumann : La répartition des valeurs avec cette méthode est très hétérogène, l'essentiel des résultats se trouvent en dessous de 1000 (plus de la moitié d'entre eux).
Explication : La manière de calculer les valeurs est incorrecte, ôter des deux côtés du nombre des chiffres jusqu'à être dans l'intervalle {0,9999} n'est pas une bonne idée, en effet, tous les nombres composés d'un nombre de chiffres impair se retrouvent dans l'intervalle {0, 999}, ce qui explique la présence d'autant de résultats dans cet intervalle.

```
hist(vn[,1],xlab='',main='Histogramme des valeurs de Von Neumann')
```



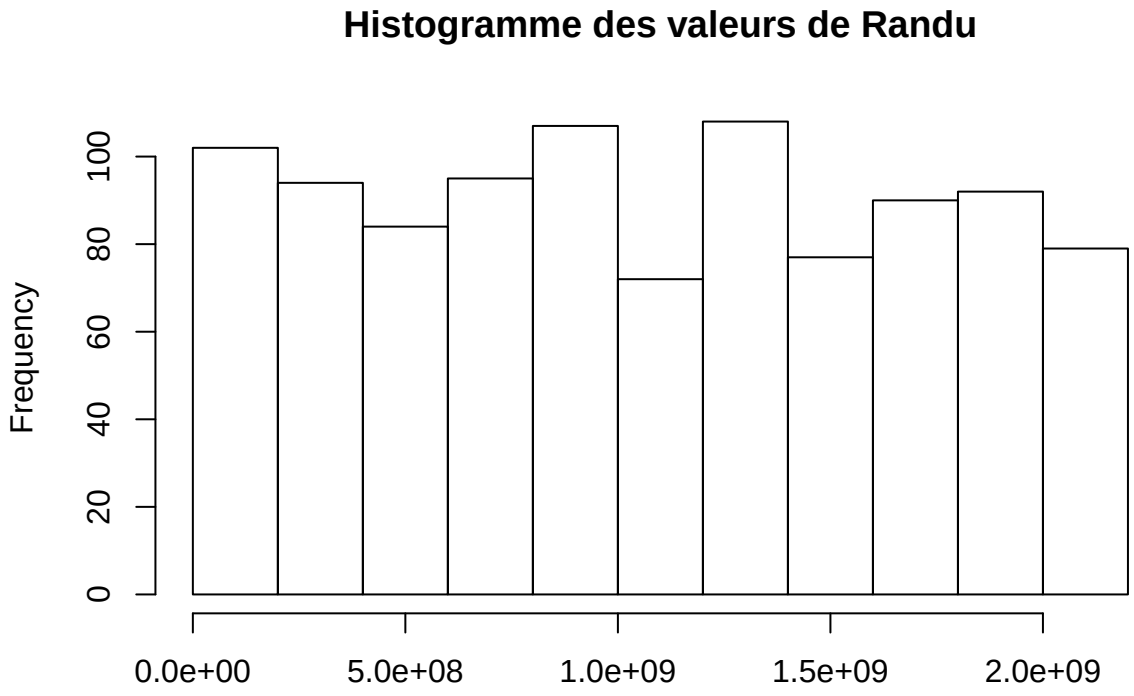
– Mersenne-Twister: La répartition des valeurs avec cette méthode est homogène et étalée dans un grand intervalle, {0 à 4*10^9}, ce qui nous laisse à penser que l'on peut obtenir des valeurs allant de 0 à 2^{32} et qui se révèle particulièrement intéressant pour les questions suivantes.
Explication : Nous pensons que cela vient de la définition et du fonctionnement de Mersenne Twister, qui est uniformément distribué ce qui en fait un générateur de nombres pseudo-aléatoire particulièrement efficace.

```
hist(mt[,1],xlab='',main='Histogramme des valeurs de Mersenne Twister')
```



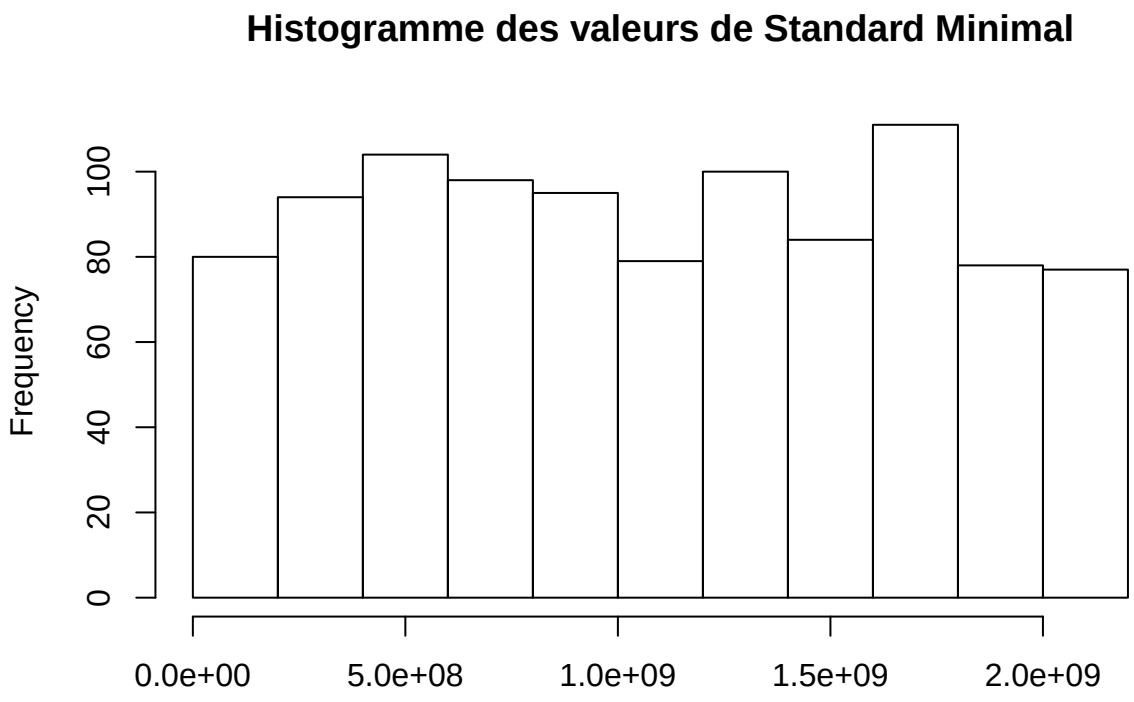
– Randu : Répartition assez homogène même si l'on retrouve régulièrement des trous, étalée dans l'intervalle : (0 à 2*10^9).
Explication : Les résultats fournis semblent uniformément distribué mais on remarque notamment sur cet histogramme qu'il possède un certain nombre de biais, engendrant alors une génération de valeurs qui ne correspond pas à ce que l'on pourrait attendre d'un générateur pseudo-aléatoire. En effectuant quelques recherches à son sujet, nous remarquons qu'il est particulièrement décrié à cause de son manque de qualité dû aux choix des variables a, c et m.

```
hist(rnd[,1],xlab='',main='Histogramme des valeurs de Randu')
```



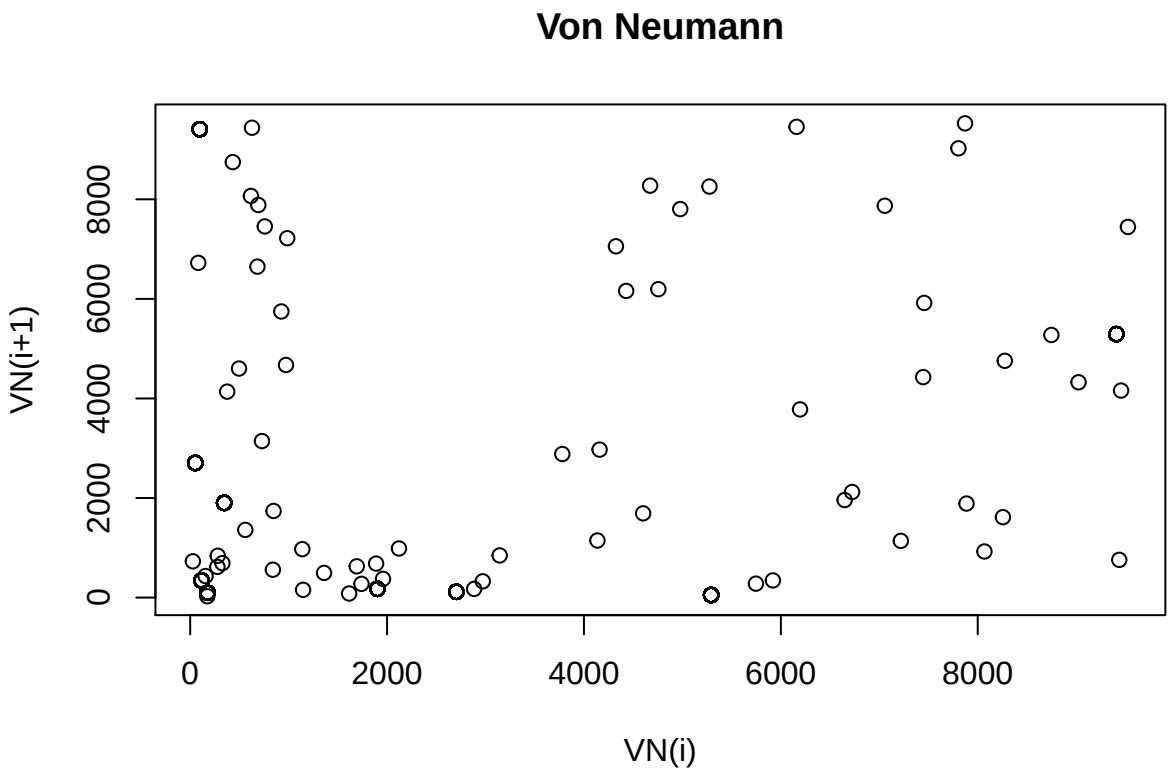
– StandardMinimal: Répartition assez homogène et disposant de peu de “trous” comparé à RANDU, elle est également étalée dans l’intervalle 0 à 2*10^9.
Explication : Les résultats obtenus avec Standard Minimal semblent plus cohérents et représentatifs de ce que pourrait renvoyer un générateur de congruence linéaire homonyme. En effet, c’est de par sa définition (soit les valeurs a, c et m choisies) qu’il nous permet d’obtenir des valeurs exploitables pour du pseudo-aléatoire.

```
hist(std[,1],xlab='',main='Histogramme des valeurs de Standard Minimal')
```



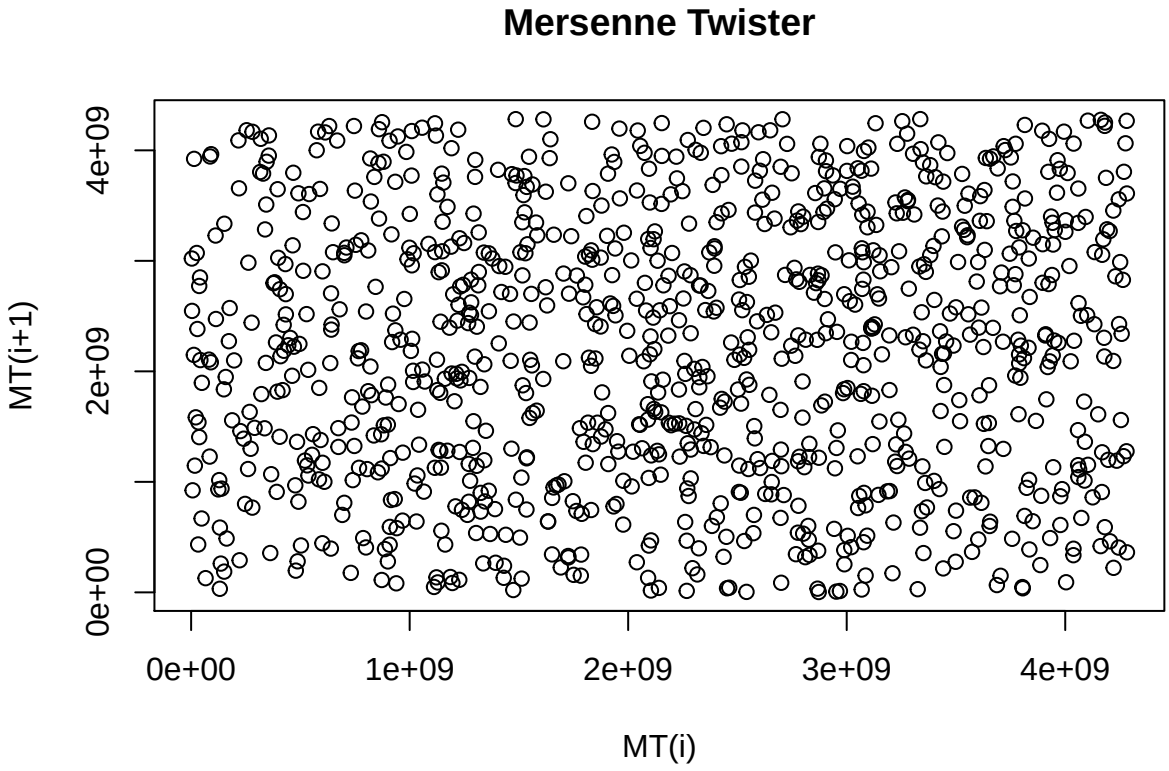
Question 2.2 :

```
plot(vn[1:(Nsimu-1),1],vn[2:Nsimu,1],xlab='VN(i)', ylab='VN(i+1)', main='Von Neumann')
```



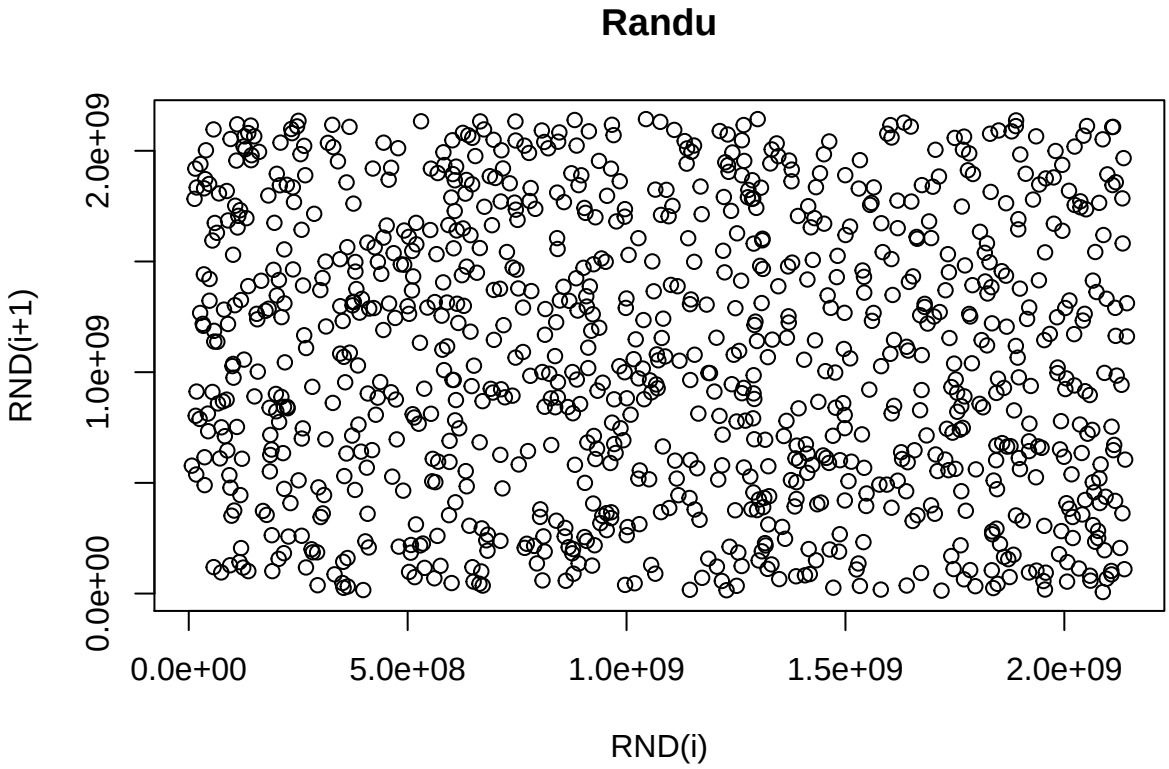
– Commentaire :

```r
plot(mt[1:(Nsimu-1),1],mt[2:Nsimu,1],xlab='MT(i)', ylab='MT(i+1)', main='Mersenne Twister')
```



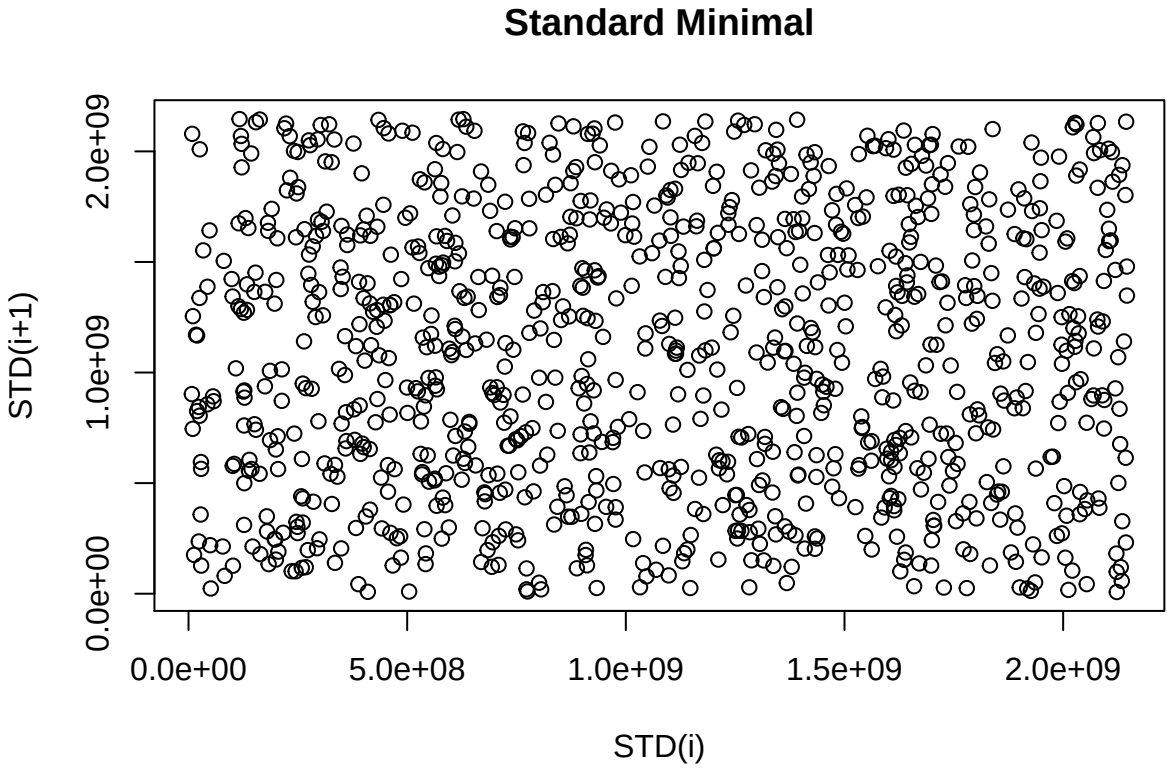
– Commentaire :

```r
plot(rnd[1:(Nsimu-1),1],rnd[2:Nsimu,1],xlab='RND(i)', ylab='RND(i+1)', main='Randu')
```



– Commentaire :

```
plot(std[1:(Nsimu-1),1],std[2:Nsimu,1],xlab='STD(i)', ylab='STD(i+1)', main='Standard Minimal')
```



– Commentaire :

Test de fréquence monobit

Pour ce test de fréquence monobit ainsi que pour les deux suivants, la séquence sera de 1000 valeurs également mais avec 100 initialisations différentes.

Question 3 :

Fonction utilisée	Nombre de bits considérés	Fréquence obtenue	Observation du test
Von Neumann	14	0.0082607	Proportion 0/1 très très inégale
Mersenne Twister	32	0.49102	Proportion 0/1 très satisfaisante
Randu	31	0.19066	Proportion 0/1 très peu égale
Standard Minimal	31	0.46116	Proportion 0/1 satisfaisante

– Commentaire : Dans l'ensemble, les résultats obtenus confirment nos attentes par rapport au test précédemment réalisé, en effet, les fonctions Mersenne Twister et Standard Minimal sont en tête du classement avec respectivement 0.49102 et 0.46116 soit quasiment une proportion de bit à 0 et 1 équivalente.
De plus, notre hypothèse concernant rendu se confirme, la proportion est tout à fait inégale et on se rend véritablement compte qu'il ne peut faire office de générateur de pseudo-aléatoire convenablement.
Le cas de Von Neumann était déjà écarté, mais ce n'est qu'une confirmation de plus concernant son incapacité à produire du pseudo-aléatoire de qualité.

Test des runs

Question 4 :

Fonction utilisée	Nombre de bits considérés	Fréquence obtenue	Observation du test
Von Neumann	14	4.658169e-08	Inférieur à 0.01
Mersenne Twister	32	0.5111088	Supérieur à 0.01
Randu	31	0.3616877	Supérieur à 0.01
Standard Minimal	31	0.5863881	Supérieur à 0.01

– Commentaire : Lors de ce test des runs, on cherche à connaître la longueur des "runs", soit les suites consécutives de 0 ou de 1. Afin de s'assurer de la qualité de l'aléatoire dans la séquence de bits observée, on doit obtenir un résultat à 0.01.
A nouveau, Mersenne Twister ainsi que Standard Minimal sont en tête du classement mais Randu retourne lui aussi une valeur bien supérieure à 0.01 ce qui ne nous permet alors pas de l'eloigner dans le cadre de ce test. Von Neumann, fidèle à lui même, prouve une nouvelle fois la mauvaise qualité des séquences qu'il produit avec un résultat très inférieur à 0.01.

Test d'ordre

Question 5 :

Fonction utilisée	Fréquence obtenue	Observation du test
Von Neumann	NA / 0	Inférieur à 0.01
Mersenne Twister	0.49	Supérieur à 0.01
Randu	0.46	Supérieur à 0.01
Standard Minimal	0.48	Supérieur à 0.01

– Commentaire : Dans ce test d'ordre qui vise à étudier directement la suite de nombre obtenus et non les bits générés, nous fixons la valeur de d à 4. et de manière conforme au test précédent, nous retrouvons des valeurs supérieures à 1% pour Mersenne Twitser, Randu et Standard Minimal et une valeur non attribuée pour Von Neumann et parfois 0. Il est intéressant de remarquer que Randu passe également ce test et pourrait nous laisser penser qu'il est, en réalité, capable de produire des suite de nombres conforme au pseudo-aléatoire si nous n'avions pas réalisé le premier test de fréquence monobit.