Université Assane Seck de Ziguinchor

U.F.R des Sciences et Technologies
Département de Mathématiques

Mémoire de Master

DOMAINE : Sciences et Technologies
Mention : Mathématiques et Applications
Spécialité : Mathématiques Appliquées
Option : statistique

Titre :

Méthodes et Tests de génération de nombres pseudo-aléatoires

Présenté par : Tidiane BA

Sous la direction de : Dr Emmanuel Nicolas CABRAL

Sous la supervision de : Pr Alassane DIÉDHIOU

Soutenu le 18 Novembre 2023 devant le jury composé de :

Prénom(s) et Nom(s)	**Grade**	**Qualité**	**Ètab.**
M. Clément MANGA	Professeur Assimilé	Président	UASZ
M. Alassane DIÉDHIOU	Professeur Titulaire	Superviseur	UASZ
M. Mor NDONGO	Maître de Conférences Titulaire	Examinateur	UASZ
M. Emmanuel Nicolas CABRAL	Maître de Conférences Titulaire	Directeur	UASZ

Année universitaire 2021-2022

DEDICACES

Je dédie ce travail :

À ma chère maman pour ses sacrifices depuis qu'elle m'a mise au monde et à mon père, mon meilleur qui m'a toujours soutenu et aidé à affronter les difficultés de la vie. Puisse **ALLAH** me donner la force et le courage de leur rendre fier de moi.

À toute ma famille plus particulièrement à mon frère Thierno Mama BA, celui qui est toujours présent pour le bon déroulement de mes études, à mon frère Abdou Salam GUEYE pour ces loyaux services rendus.
À mes frères Papis BA, Abou BA, Oumar NDIAYE, Adama DEME.
À ma grande soeur Aminata CISSE et qui ne cesse de prier pour moi.

À Monsieur le maire et D.G de la S.A.E.D Aboubacry SOW qui n'a ménagé aucun effort pour la réussite de la tenue de ma soutenance.

À toute ma famille Diola de Ziguinchor, plus particulièrement Ta Amy SAGNA, Mère Sali BODIAN, mon père Saidou DIÉDHIOU, ma sœur Mamy DIATTA, mon frère Ibrahima DIALLO, mon frère Cheikh Sadibou DIÉDHIOU.

REMERCIEMENTS

Tout d'abord, je remercie **ALLAH**, le tout puissant de m'avoir donné la foi, le courage et la patience pour l'achèvement de ce travail.

Je voudrais exprimer toute ma reconnaissance à mon directeur de mémoire, Mr Emmanuel Nicolas CABRAL, avec qui j'ai eu un énorme plaisir à travailler, en plus de ses conseils et de ses suggestions, j'ai toujours bénéficié de ses encouragements et de sa disponibilité.
Veuillez recevoir ma profonde gratitude et ma parfaite considération.

Je tiens à remercier vivement le professeur Clément MANGA, qui m'a fait l'honneur de présider le jury de ce mémoire.
Je remercie sincèrement le professeur Alassane DIÉDHIOU d'avoir accepté de superviser ce travail. Je remercie également le professeur Mor NDONGO qui a bien accepté d'examiner ce travail.

Mes plus vifs remerciements à mon Papa et ma Maman, mes frères et sœurs, mes cousins et cousines pour le soutien qu'ils m'ont apporté durant toutes mes années d'études, ainsi que toute ma famille.

Mes remerciements s'adressent spécialement aux docteurs Souhaibou SAMBOU et Abdoulaye DIOUF pour leur aide et encouragement. Merci pour le temps que vous m'avez accordé.

J'exprime ma reconnaissance envers l'Université Assane Seck de Ziguinchor, l'UFR des Sciences et Technologies, le Département de Mathématiques.

Enfn, je souhaiterais remercier mes camarades de promotion, plus particulièrement à Saliou DIAW, Seydi Diamil DIOUF, Amadou BALDE et Fatou DIENG.

Résumé

Notre travail consiste en l'étude des méthodes et tests de génération de nombres pseudo-aléatoires.
Les nombres aléatoires peuvent être fournis par des procédés naturels. Cependant nous avons constaté que ces procédés présentent des limites. En effet ces limites ont conduit aux scientifiques à concevoir et mettre en œuvre d'autres procédés de génération des nombres aléatoires. Ces procédés sont de type algorithmique, d'où les sorties de ces procédés sont appelées nombres pseudo-aléatoires parce qu'elles ne sont pas entièrement aléatoires mais seulement s'approchent des propriétés idéales des sources complètement aléatoires.
Nous étudions en détails ces méthodes ou procédés de génération et proposons quelques propriétés.
Les nombres pseudo-aléatoires issus de ces méthodes sont subis à plusieurs tests pour vérifier leur uniformité et leur indépendance.
Notre objectif est de générer des nombres pseudo-aléatoires uniformes, identiquement distribués et indépendants sur l'intervalle $[0, 1]$.

Table des matières

Table des figures

Introduction générale

L'intérêt scientifique de disposer de séquences de nombres aléatoires est contemporain au développement des calculateurs sophistiqués que sont nos ordinateurs modernes. Les premières machines, par exemple Electronic Discrete Variable Automatic Computer, étaient lentes et peu performantes en comparaison des standards d'aujourd'hui. Aussi, effectuer des calculs complexes était une tâche tellement difficile que l'option de recourir à des processus stochastiques fut considérée attentivement pour éviter les écueils qui en découlaient. Aussi saugrenue que l'idée puisse sembler, confier les sciences exactes au hasard allait donner des résultats plus qu'impressionnants (voir [5]).
Pour simuler un modèle stochastique, il est nécessaire de disposer d'une source de nombres au hasard susceptibles de jouer en pratique le rôle des variables aléatoires qui interviennent dans la définition du modèle. La génération de tels nombres s'accomplit en général en deux étapes de nature différente :

- la génération de nombres au hasard U_1, U_2, . . . jouant le rôle de variables aléatoires indépendantes, identiquement distribuées et uniformes sur l'intervalle $[0, 1]$,
- la transformation des nombres U_1, U_2, . . . précédents en des nombres susceptibles de jouer le rôle des variables aléatoires intervenant dans la définition du modèle, et qui ne sont en général ni indépendantes identiquement distribuées, ni uniformes sur $[0, 1]$.

Nous parlons en général de nombres pseudo-aléatoires pour désigner les nombres au hasard qui apparaissent au cours de ces deux étapes. Nous étudierons en détail dans la suite ces nombres pour souligner leur différence vis-à-vis de véritables suites de variables aléatoires indépendantes identiquement distribuées.
Nous distinguons deux grandes familles de générations de nombres pseudo-aléatoires :

- les générateurs uniformes,

- les générateurs non-uniformes.

Les premiers visent à générer des nombres indépendants, identiquement distribués et uniformes sur l'intervalle $[0, 1]$. Ils forment la base fondamentale de toutes les théories stochastiques. Aussi, un corpus scientifique important leur est consacré.
Les seconds sont désignés par une définition négative car ils visent à générer toutes les distributions autres que l'uniforme. Ces générateurs exploitent généralement les générateurs uniformes pour parvenir à leur fin.
Ce mémoire est organisé comme suit. Nous commençons d'abord par rappeler dans le premier chapitre quelques outils probabilistes. Dans le deuxième chapitre, nous allons étudier les méthodes de générations des nombres pseudo-aléatoires. Le troisième chapitre sera consacré à l'étude des différents tests des séquences de nombres générés par ces méthodes. Et enfin le quatrième chapitre est réservé à une application numérique.

Chapitre 1

Rappels d'outils probabilistes

Nous allons commencer par énoncer quelques notions de probabilités utiles pour la suite du travail. La théorie des probabilités est l'étude mathématique des phénomènes caractérisés par le hasard et l'incertitude. Le calcul des probabilités a commencé avec Blaise Pascal, Pierre Fermat, Christian Huygens et Jacques Bernoulli par l'analyse des jeux dits de hasard (voir [15]).
Nous allons présenter ici quelques définitions utiles pour la suite du travail.

Définition 1.
Si X est une variable aléatoire réelle, la fonction de répartition de X est la fonction $F_X : \mathbb{R} \to [0,1]$ définie par

$$F_X(t) = \mathbb{P}_X(]-\infty, t]) = \mathbb{P}(X \leq t) \ , \ \forall t \in \mathbb{R}.$$

Cette fonction de répartition vérifie certaines propriétés :

- $0 \leq F_X(t) \leq 1$,
- $\lim\limits_{x \to -\infty} F_X(x) = 0$ et $\lim\limits_{x \to +\infty} F_X(x) = 1$,
- F_X est croissante et continue à droite,
- Pour tout $x \in \mathbb{R}$, $\mathbb{P}(X = x) = F_X(x) - F_X(x^-)$,
  où $F_X(x^-) = \lim\limits_{t \to x,\ t<x} F_X(t)$,
- l'ensemble des points de discontinuité de F est au plus dénombrable,
- si l'ensemble des points de discontinuité est vide, la variable aléatoire est continue et autrement dit sa fonction de répartition est continue,
- Pour $a < b$: $\mathbb{P}(a < X \leq b) = F(b) - F(a)$.

Définition 2.
Une variable aléatoire X est à densité, ou continue, s'il existe une fonction f définie sur $\mathbb{R}$ telle que la fonction de répartition de X s'écrit :

$$\forall x \in \mathbb{R} \quad F_X(x) = \int_{-\infty}^{x} f(t)dt,$$

où f est une fonction intégrable sur $\mathbb{R}$ satisfaisant les conditions suivantes :

- $f(t) \geq 0$ pour tout $t \in \mathbb{R}$,
- $\int_{-\infty}^{+\infty} f(t)dt = 1$.

Une fonction qui vérifie ces deux conditions est appelée densité de probabilité.

Moment et variance d'une variable aléatoire

Définition 3.
On appelle probabilité sur $(\Omega; \mathcal{E})$ *(ou mesure de probabilité) une application* $\mathbb{P}$ *de* $\mathcal{E}$ *dans* $[0, 1]$ *telle que :*

- $\mathbb{P}(\emptyset) = 0$ *et* $\mathbb{P}(\Omega) = 1$,
- *Pour toute suite* $(A_n)_{n\geqslant 0}$ *d'évènements, deux à deux incompatibles :*

$$\mathbb{P}\left(\cup_{n=0}^{+\infty} A_n\right) = \sum_{n=0}^{+\infty} \mathbb{P}\left(A_n\right) \qquad \textit{appelée la } \sigma\textit{-additivité.}$$

On appelle espace probabilisé, le triplet $(\Omega, \mathcal{E}, \mathbb{P})$.

- Cas discret : soit X une variable aléatoire réelle à valeurs dans $\{x_i, i \in I\}$, avec I dénombrable. On dira que X admet une espérance mathématique si la série $\sum_{i\in I} |x_i|\mathbb{P}(X = x_i)$ est convergente.
  Dans ce cas on appelle **espérance mathématique** de X le nombre réel noté $E(X)$ défini par :

$$E(X) = \sum_{i\in I} x_i \mathbb{P}(X = x_i).$$

  On appelle **moment d'ordre** k de X, l'espérance mathématique de X^k(si elle existe) définie par

$$E(X^k) = \sum_{i\in I} x_i^k \mathbb{P}(X = x_i).$$

  On appelle **moment centré** d'ordre $k, k \in \mathbb{N}^*$ de X, l'espérance mathématique de $(X - E(X))^k$ si elle existe.
  On appelle **Variance** de X et on note $V(X)$ ou $Var(X)$, le moment centré d'ordre 2 de X si elle existe, définie par :

$$V(X) = E[X - E(X)]^2.$$

- Cas de variable aléatoire à densité : soit X une variable aléatoire réelle définie sue l'espace probabilisé $(\Omega, \mathcal{E}, \mathbb{P})$ admettant une densité f.
  On dit que X admet une espérance mathématique si l'intégrale généralisée $\int_{\mathbb{R}} xf(x)dx$ est absolument convergente. Ainsi l'espérance mathématique est définie par :

$$E(X) = \int_{\mathbb{R}} xf(x)dx.$$

  Comme dans le cas discret on a le moment d'ordre k suivant :

$$E(X^k) = \int_{\mathbb{R}} x^k f(x)dx.$$

  Ainsi la variance est donnée par cette formule :

$$V(X) = \int_{\mathbb{R}} [x - E(X)]^2 f(x)dx = E(X^2) - [E(X)]^2.$$

Chapitre 2

Génération des nombres pseudo-aléatoires

Il existe plusieurs types d'algorithmes déterministes pour la génération de nombres pseudo-aléatoires (afin d'alléger le texte nous désignerons un générateur de nombres pseudo-aléatoire par générateur de nombres aléatoires ou simplement générateur). Dans [12], [10] et [1], nous trouvons un excellent compte-rendu des principaux algorithmes de génération de nombres pseudo-aléatoires.
Notre travail porte sur la génération de la distribution uniforme et non-uniforme.
Après avoir fait une briève étude sur l'architecture d'un générateur pseudo-aléatoire, nous parlerons en premier lieu de la génération uniforme et en deuxième lieu de la génération non-uniforme.

2.1 Architecture d'un générateur

Définition 4.
Un générateur pseudo-aléatoire est un procédé qui, à partir d'une initialisation de taille fixée (généralement d'une ou quelques centaines de bits) appelée graine ou germe, engendre de manière déterministe une suite de très grande longueur que l'on ne peut pas distinguer d'une suite aléatoire quand on ne connaît pas la graine.

Remarque 1.
Est déterministe ce qui produit les mêmes résultats avec les mêmes données et les mêmes méthodes, en somme, ce qui peut être prédit.

Définition 5.
Une graine est une valeur numérique servant à l'initialisation d'un générateur

de nombres pseudo-aléatoires. Avec une même graine initiale, la séquence de nombres pseudo-aléatoires produite par le générateur de nombres pseudo-aléatoires est déterministe et peut par conséquent être reproduite.

Définition 6.
Une séquence de nombres pseudo-aléatoires est une séquence de nombres générés avec un générateur de nombres pseudo-aléatoires. Cette séquence a pour propriété que l'on peut prédire les nombres à venir à partir des nombres déjà connus et de la graine initiale.

2.1.1 Principe de construction

Soit un espace fini d'états $\mathcal{S}$, une loi de probabilité $\mathbb{P}$ qui permet de choisir un élément de $\mathcal{S}$, une fonction $f : \mathcal{S} \to \mathcal{S}$, appelée fonction de transition, et une fonction $g : \mathcal{S} \to \mathcal{U} \subset \mathbb{R}$, appelée fonction de sortie.

Définition 7.
On appelle générateur de nombres pseudo-aléatoires un algorithme qui choisit un élément $s_0 \in \mathcal{S}$, appelée graine ou germe, grâce à la loi de probabilité $\mathbb{P}$, et qui produit une séquence $u_n = g(s_n) \in \mathcal{U}$; $n \geq 1$, où $s_n = f(s_{n-1})$, qui tente d'imiter une séquence de variables aléatoires indépendantes et uniformes sur $\mathcal{U}$. On note ce générateur pseudo-aléatoire comme suit :

$$G = (\mathcal{S}, \mathbb{P}, f, \mathcal{U}, g).$$

Le générateur produit, a chacune des itérations, un nouvel état $s_n \in \mathcal{S}$ avec f et une sortie $u_n \in \mathcal{U}$, obtenue avec g. L'ensemble d'états étant fini et la séquence étant générée par la fonction de transition f, un générateur uniforme possède une période ρ telle que $X_{i+l\rho} = X_i$ pour tout $i \geq l$ et $l \geq 0$. La constante ρ est appelée la période de la séquence et la constante l la transitoire. Les constantes ρ et l sont des entiers et ρ ne peut excéder le cardinal $|\mathcal{S}|$ de l'ensemble d'états $\mathcal{S}$.

Remarque 2. *Choix de l'état initial*
L'état initial s_0 , qui permet d'amorcer la récurrence, est aussi appelé graine (seed en anglais) ou encore germe du générateur. En dehors de ces besoins particuliers, il est recommandé d'amorcer le générateur avec des graines uniformément distribuées dans l'espace des états, ce qui permet d'envisager, équitablement et sans biais, l'ensemble des évolutions possibles pour le modèle. Comme l'objectif visé est l'analyse d'un phénomène simulé et non pas la sécurité d'un système, on peut engendrer les graines successives avec un générateur pseudo-aléatoire auxiliaire (plus facile à exploiter qu'une source de hasard physique).

2.1.2 Qualités d'un générateur

Cette section discute des qualités que devraient avoir un générateur. Celles-ci permettent de nous guider dans le choix de $\mathcal{S}$, f et g. Au risque de nous répéter, les générateurs de nombres pseudo-aléatoires doivent être construits de façon soigneuse afin de bien imiter une source de hasard parfaite. La très grande majorité des générateurs de nombres pseudo-aléatoires tentent de reproduire une suite de variables aléatoires uniformes et indépendantes dans l'intervalle $[0, 1]$. Ce type de variables aléatoires est très utile puisqu'elle permet de reproduire des variables aléatoires qui suivent n'importe quelle loi de probabilité (voir [9]).
Dans [11] et [12], les auteurs décrivent les principaux critères permettant de définir ce qu'est un bon générateur de nombres aléatoires. Ces critères sont les bonnes propriétés d'uniformités distribuées et d'indépendances statistiques des nombres pseudo-aléatoires successifs. Le générateur devrait passer les tests statistiques pour l'uniformité et l'indépendance.

2.1.3 Domaines d'applications

Les générateurs pseudo-aléatoires ont, à travers le temps, acquis une grande importance dans divers domaines, allant du médical à la sécurisation, et se sont montrés être d'une grande efficacité quand à leurs apports dans ces domaines (voir [5]) :

- **simulation stochastique (Monte Carlo)** : utilisée en sciences statistiques, gestion, etc. On simule un modèle mathématique d'un système complexe, pour mieux comprendre le comportement du système, ou optimiser sa gestion, etc. Souvent, on veut estimer une mesure de performances définies par une espérance mathématique (une intégrale). Ici, on veut que les propriétés statistiques du modèle mathématique soient bien reproduites par le simulateur.
- **confidentialité des échanges sur les réseaux sans fils** : l'échange par réseaux sans fils pose de nombreux problèmes concernant la confidentialité des éléments échangés. Ceci dit, il faut alors un mécanisme qui peut conserver le mieux possible cette confidentialité, ce mécanisme est alors le Wired Equivalent Privacy. Le principe consiste à définir une clé chiffrée sur une longueur allant de 40 à 128 bits. Ensuite, la clé est déclarée au niveau du point d'accès et du client. Le but de la clé est de créer un nombre pseudo-aléatoire d'une longueur égale à celle de la trame de transmission. Le nombre pseudo-aléatoire ainsi généré sert à chiffrer la transmission, et alors à assurer la confidentialité de cette

dernière. Toutefois, le Wired Equivalent Privacy n'assure malheureusement pas une sécurité optimale, les compagnies Fluhrer et Shamir ont montré que la génération de la chaîne pseudo-aléatoire rend possible la découverte de la clé de session en stockant 100 Mo à 1 Go de trafics créé intentionnellement. Notons aussi que 24 bits de la clé sont dédiés à l'initialisation, ce qui veut dire que chiffrer à 128 bits est de loin meilleur que de chiffrer à 64 bits dans la mesure où chiffrer à 128 bits offre une possibilité réelle de chiffrer de 104 bits tandis que chiffrer à 64 bits n'en offre que 40 bits.

- **sécurisation des applications WEB** : les applications WEB qui ont pour utilisateurs les navigateurs WEB ont un système qui sert à protéger leurs clients par la création de sessions. Les sessions sont des espaces de travail qui ont un espace de stockage d'informations connues, elles sont aussi privées et appartiennent à un utilisateur particulier dûment authentifié. Néanmoins, ces sessions ne présentent pas toujours le niveau de sécurité souhaité. C'est ici qu'interviennent les nombres pseudo-aléatoires. L'exemple le plus connu est celui d'attaques par session de type prédiction (méthode de détourner ou de personnifier un utilisateur WEB, elle s'accomplit par la déduction ou l'estimation de la valeur unique qui identifie la session associée). Pour éviter ce genre d'attaques, il est devenu courant d'utiliser un gestionnaire d'applications WEB capable de générer des identifiants qu'on ne peut pas prévoir. Il n'y a plus à chercher, car les applications de générations de nombres pseudo-aléatoires ont été suffisamment développées pour se faire.
- **système de chiffrements** : la cryptanalyse est une discipline récente, et signe un grand niveau de fiabilité quant aux méthodes de chiffrements qu'elle utilise, notamment le chiffrement à flot. Ce denier consiste à additionner bit à bit au message clair une suite binaire pseudo-aléatoire de même longueur appelée suite chiffrant. Ce procédé est reconnu pour sa rapidité et sa compétitivité vis-à-vis des autres procédés dans la mesure où il est peu sensible aux erreurs introduites lors de la transmission. Des registres à décalage avec rétroaction linéaire font souvent partie des générateurs pseudo-aléatoires utilisés pour produire les suites chiffrant.
- **domaine biomédical** : une application importante au niveau biomédical concerne l'amélioration de la modélisation du dépôt de dose dans le corps et aussi pour accélérer le calcul de traitement. En effet, sur les grilles de calcul, il est impératif que chaque processus de calcul puisse disposer d'une sous séquence aléatoire issue d'une séquence globale de nombres aléatoires à générer. L'un des fruits des recherches menées à ce sujet est la plate-forme de simulation Monte Carlo GATE (Geant4

Application for Emission Tomography), qui est un logiciel de calcul de dépôt de dose sur plusieurs grappes, et qui a permis de mettre en évidence des facteurs d'accélérations de l'ordre de 30 du temps de calcul en comparaison avec une utilisation en milieu hospitalier.

2.2 Génération aléatoire

Il existe deux types de générations aléatoires, la génération de la distribution uniforme et non-uniforme.
Les générations de la distribution uniforme et non uniforme n'appartiennent pas à la même méthode de générateurs : il va donc être intéressant dans un premier temps de faire l'étude théorique de leurs familles avant de passer aux tests.

2.2.1 Génération de la distribution uniforme

Quand on considère la génération de distributions non-uniformes, le choix d'un générateur uniforme est généralement laissé à la discrétion de l'usager. Aussi, la disponibilité d'un générateur uniforme idéal est souvent considérée comme une hypothèse de travail raisonnable. Ce qui importe avant tout au concepteur, c'est l'algorithme qui permet de faire le pont entre des échantillons indépendants et identiquement distribués de l'uniforme et la distribution visée. Cependant, les considérations rattachées à la génération de la distribution uniforme dans un environnement numérique (autant logiciel que matériel) se généralisent aisément à la génération de distributions non-uniformes (voir [14] et [15]). La distribution uniforme $\mathcal{U}([0;1])$ est définie par sa fonction de densité de probabilité :

$$f(u) = \begin{cases} 1 & \text{si } 0 \leq u \leq 1 \\ 0 & \text{sinon} \end{cases} \tag{2.1}$$

Pour générer u, il est d'usage de générer un entier x entre 1 et $m-1$ (on choisit m très grand) et d'obtenir u par la fraction $u = \frac{x}{m}$ (voir [6]). La question qui reste à poser est : comment générer x de manière équiprobable sur l'intervalle des entiers de 1 à $m-1$? La réponse à la question est donnée par la remarque de D. Knuth ci-dessous.

2.2.1.1 Générateurs de congruence linéaire

Le but du Générateur de Congruence Linéaire (**GCL**) est de créer une suite de nombres entiers de manière aléatoire ou plutôt avec aussi peu de régularité

que possible. La plupart des logiciels de calcul ou de développement dispose d'un générateur de nombres aléatoires. Pour des raisons principalement historiques, celui-là est souvent de type congruence linéaire.
C'est la méthode la plus utilisée pour générer des nombres aléatoires. Il s'agit de la suite inventée par Derrick Henry Lehmer en 1948 (voir [13]) et définie par :

$$x_n = (ax_{n-1} + c) \ mod \ m, \tag{2.2}$$

où a, c et m sont des entiers positifs appelés respectivement multiplicateur, incrément et module.
On propose quelques propriétés et vocabulaires :

- $x_n < m$,
- le nombre de valeurs possibles pouvant être fournis par un générateur de congruence linéaires est au plus égal à m,
- si un nombre apparait une deuxième fois, tous les nombres le suivant apparaissent aussi une deuxième fois et selon le même ordre, ainsi, un générateur de congruence linéaire est nécessairement périodique, sa période maximale vaut m.

Remarque 3. D. Knuth
Pour obtenir un bon générateur, il y'a quelques critères à respecter. Ainsi, D. Knuth a montré qu'avec $c \neq 0$, on pouvait toujours obtenir une suite de période m (voir [6]).
Ainsi, on obtiendra tous les nombres compris entre 0 et $m-1$. Pour cela, il faut respecter les critères suivants :

- c et m doivent être premiers entre eux c'est-à-dire $pgcd(c, m) = 1$,
- pour tout nombre premier p qui divise m, $(a-1)$ doit être multiple de p,
- si m est multiple de 4, alors $(a-1)$ doit aussi être multiple de 4, pour rappel, on a $(a = i \ mod \ j)$ si a peut s'écrire sous la forme $a = i + kj$ où k est un entier positif ou nul.

Exemple 1.
Soit le générateur de congruence linéaire suivant : $x_n = (ax_{n-1} + 1) \ mod \ 16$. Trouver a et c pour que ce générateur atteigne sa période maximale soit 16.

- Pour vérifier la première condition, on peut prendre $c = 3$. En effet, 3 et 16 sont premiers entre eux.
  On peut noter qu'on aurait pu prendre $c = 5$ ou $c = 7$ mais pas 2 ni 4.
- Le seul nombre premier divisant $m = 16$ est 2. Le nombre $(a-1)$ doit être donc multiple de 2.

D'autre part, $m = 16$ étant multiple de 4, $(a-1)$ doit aussi être multiple de 4.
On peut par conséquent prendre $a = 5$ ce qui permet de vérifier les deux conditions en même temps.

Remarque 4.
En respectant les critères de D. Knuth, nous sommes sûr que tous les nombres sortent. Mais ceci ne garantit pas que le générateur soit suffisamment aléatoire. En général, il est donc assez délicat de dire qu'un générateur de congruence linéaire est bon.
Pour pouvoir qualifier un générateur de bon, il faut absolument faire subir à ce dernier une batterie de tests statistiques variés (voir [8]).

2.2.1.2 Générateurs de congruence multiplicative

Définition 8.
*Lorsque $c = 0$, on parle de Générateur de Congruence Multiplicative (**GCM**) et l'équation* (2.2) *devient :*

$$\forall k \in \mathbb{N}^*, x_k = (ax_{k-1}) \ mod \ m. \tag{2.3}$$

Dans ce cas, l'état 0 est absorbant : si $x_k = 0$, alors les termes suivants dans la suite seront tous nuls. Le générateur doit donc prendre ses valeurs entre 1 et $m - 1$.
D. Knuth dans [6] démontre le lemme 1 suivant :

Lemme 1.
Soit x un GCM défini par (m, a) et x_0. Si m est premier, la période maximale vaut $m - 1$. Elle est atteinte si et seulement si $x_0 \wedge m = 1$ et a est primitif modulo m.
Si $m = 2^n$ (avec $n \geq 4$), la période maximale vaut $\frac{m}{4}$. Elle est atteinte si et seulement si $(x_0 \ mod \ 8 = 1)$ et $(a \ mod \ 8) = \pm 3$.

Exemple 2. *Exemples de générateurs de congruence*
Les générateurs de congruences ont été très utilisés en pratique par les logiciels et les langages de programmation notamment au début de l'ère informatique (voir [13]).
Nous pouvons citer :

- Rand() du langage C ANSI avec $m = 231$, $a = 1103515245$, et $c = 12345$,
- Randu d'International Business Machine, IBM des années 60 avec $m = 231$, $a = 65539$, et $c = 0$,

- Générateur de D. Knuth et Lewis : $x_n = 69069x_{n-1} \bmod 232$,
- La fonction drand48() (en C ANSI) qui utilise les paramètres $m = 248$, $a = 25214903917$ et $c = 11$,
- Le générateur du logiciel MAPLE avec $m = 1012$, $a = 42741969081$ et $c = 0$.

2.2.1.3 Autres générateurs

Ces types de générateurs sont étudiés avec plus de détails dans [7] :

- **Générateur de Congruence Quadratique (GCQ)** : le schéma suivant a été proposé comme alternative aux générateurs de congruence linéaire. Soit $(x_0 \ mod \ 4 = 2)$, et

$$x_{n+1} = x_n(x_n + 1) \ mod \ 2^{\omega}, \tag{2.4}$$

  où ω est un entier positif donné.

- **Méthode de Fibonacci** : Rappelons que la suite de Fibonacci est définie comme

$$x_{n+1} = x_n + x_{n-1}, \tag{2.5}$$

  pour $n = 1, 2, \ldots$ avec $x_0 = 1$ et $x_1 = 1$.
  Le générateur de nombres pseudo-aléatoires de Fibonacci est basé sur

$$x_{n+1} = (x_n + x_{n-1}) \ mod \ m, \tag{2.6}$$

  où x_0 et x_1 sont pris comme graines aléatoires, et m est un grand entier positif donné. Cette méthode est trop coûteuse pour l'application numérique et requiert beaucoup de temps pour donner un résultat.
- **Générateur R de Wichman et Hill** : l'un des générateurs disponibles pour une utilisation dans R est dû à Wichman et Hill en 1982. Les chiffres sont obtenus en prenant la partie fractionnaire de la somme de nombres pseudo-aléatoires provenant de trois générateurs de congruence multiplicative qui n'ont que des valeurs m modérées. On peut montrer que la partie fractionnaire de la somme de trois variables aléatoires uniformes indépendantes sur $[0, 1]$ est, elle-même, une variable aléatoire uniforme, fournissant la justification théorique de la méthode.

2.2.2 Génération de distributions non-uniformes

Les distributions non-uniformes peuvent être utilisées pour simuler différents comportements aléatoires. Par exemple, la loi exponentielle est exploitée pour simuler l'intervalle de temps entre deux événements indépendants, tel que le temps entre deux appels téléphoniques, le temps entre deux requêtes adressées à une imprimante, etc (voir [17]).
Il existe différentes techniques pour générer ces distributions. Parmi lesquelles, nous pouvons citer la méthode de l'inverse, la méthode de rejet et la méthode de Box-Muller.

2.2.2.1 Méthode de l'inverse

Soit F une fonction de répartition bijective, alors si U suit une loi uniforme sur $[0,1]$, la variable aléatoire $X = F^{-1}(U)$ a pour fonction de répartition F. En effet, pour tout réel x,

$$\mathbb{P}(X \leq x) = \mathbb{P}\left(F^{-1}(U) \leq x\right) = \mathbb{P}(U \leq F(x)) = F(x),$$

où l'on a appliqué respectivement l'aspect bijectif de F, sa croissance et la forme spécifique de la fonction de répartition de la loi uniforme.
Pour ce qui nous concerne, l'intérêt de ce résultat est clair : pour que F soit facilement inversible, alors pour générer une variable aléatoire de loi F, il suffit de générer une variable uniforme U et de lui appliquer F^{-1}. On peut en fait généraliser ceci à des fonctions de répartition non bijectives (voir [3]).

Définition 9. (Inverse généralisée)
Soit X une variable aléatoire de fonction de répartition F. On appelle inverse généralisée de F, ou fonction quantile de X, la fonction F^{-1} définie pour tout $u \in]\,0,1[$ par

$$F^{-1}(u) = \inf\{x \in \mathbb{R} : F(x) \geq u\}.$$

Proposition 1.
Soit U une variable uniforme sur $[0,1]$, F une fonction de répartition et F^{-1} son inverse généralisée. Alors $F^{-1}(U)$ a même loi que X.

Preuve.
Soit $X = F^{-1}(U)$ et x réel fixé. Montrons que pour tout $u \in]\,0,1]$, on a

$$F^{-1}(u) \leq x \Longleftrightarrow u \leq F(x). \tag{2.7}$$

Par définition de $F^{-1}(u)$, si $u \leq F(x)$, alors $F^{-1}(u) \leq x$. Inversement, si $F^{-1}(u) \leq x$, alors pour tout $\varepsilon > 0$ on a $F^{-1}(u) < x + \varepsilon$, donc par définition

de $F^{-1}(u)$ et par croissance de F, il vient $u \leq F(x+\varepsilon)$. Puisque F est continue à droite, on en déduit que $u \leq F(x)$ et l'équivalence (2.7) est établie. Dès lors, la fonction de répartition de X se calcule facilement :

$$\mathbb{P}(X \leq x) = \mathbb{P}\left(F^{-1}(U) \leq x\right) = \mathbb{P}(U \leq F(x)) = F(x),$$

la dernière égalité venant de ce que, pour tout $u \in [0,1]$, $\mathbb{P}(U \leq u) = u$. □

2.2.2.2 Méthode de rejet

L'idée de la méthode du rejet est de dominer la loi que l'on cherche à simuler par une autre que l'on sait simuler simplement (typiquement, par inversion). Bien que la méthode fonctionne aussi dans le cas discret, concentrons-nous sur le cas des vecteurs aléatoires à densité. On se donne f et g deux fonctions mesurables positives (et non identiquement nulles) sur $\mathbb{R}^d$ et l'on fait les hypothèses suivantes :

- la fonction g est une densité de probabilité : $\int_{\mathbb{R}^d} g(x)dx = 1$;
- il existe $c > 0$ tel que pour tout $x \in \mathbb{R}^d$, on a $f(x) \leq cg(x)$.

L'objectif est alors de simuler la loi de densité (proportionnelle à) la fonction f. À noter que cette méthode a l'avantage d'être valable dans le cas multi-dimensionnel, même si au final on simule la loi de densité g en utilisant la méthode des conditionnements successifs. Par ailleurs, mentionnons que l'on n'a pas besoin de connaitre l'intégrale de la fonction f sur $\mathbb{R}^d$ pour utiliser la méthode du rejet (d'ailleurs, quitte à remplacer c par c fois cette intégrale, on supposera dans la suite pour simplifier que f est elle aussi une densité). En pratique, cela peut avoir un intérêt lorsque l'on ne connaît la densité de la loi à simuler qu'à une constante près et que le calcul de l'intégrale ci-dessus est coûteux.
Lorsque $g(x) > 0, x \in \mathbb{R}^d$, on pose $\alpha(x) = f(x)/cg(x)$, qui est à valeurs dans l'intervalle $[0,1]$ et que l'on suppose explicite en pratique. Pour un vecteur aléatoire X de densité g, on a évidemment que $\mathbb{P}\left(X \in g^{-1}(\{0\})\right) = 0$, d'où le fait que $\alpha(X)$ soit presque bien défini. On a alors le résultat suivant, dont la première remarque venant après la démonstration justifie la dénomination de méthode du rejet (voir [4]).

Proposition 2.
Considérons une suite $(X_n, U_n)_{n\geq 1}$ de vecteurs aléatoires indépendants (avec de surcroît le vecteur X_n indépendant de la v.a.r. U_n) tels que les X_n suivent la loi de densité g et les U_n la loi uniforme sur $[0,1]$. Posons

$$T = \inf\left\{k \geq 1 : U_k \leq \alpha\left(X_k\right)\right\},$$

le premier instant k pour lequel $U_k \leq \alpha(X_k)$. Alors la v.a. entière T suit une loi géométrique sur $\mathbb{N}^*$ de paramètre

$$p = \mathbb{P}(U_1 \leq \alpha(X_1)),$$

appelé la probabilité d'acceptation. Le vecteur aléatoire X_T (qui est donc bien défini car T est p.s. finie) a pour densité la fonction f.

Remarque 5.
Si $p \in]0,1[$, nous disons que N suit une loi géométrique de paramètre p si N est à valeurs dans $\mathbb{N}^\star$, avec pour tout $n \in \mathbb{N}^\star : \mathbb{P}(N = n) = p(1-p)^{n-1}$, ce qui équivaut à dire que $\mathbb{P}(N > n) = (1-p)^n$. L'autre convention, qui est par exemple celle de R pour la fonction rgeom, consiste à considérer N à valeurs dans $\mathbb{N}$, avec pour tout $n \in \mathbb{N} : \mathbb{P}(N = n) = p(1-p)^n$. La seconde variable se déduit tout simplement de la première en enlevant 1 .

Preuve.
Tout d'abord, notons que la probabilité d'acceptation vaut, par indépendance de X_1 et de U_1,

$$p = \mathbb{P}(U_1 \leq \alpha(X_1)) = \int_{\mathbb{R}^d} \left(\int_0^1 1_{\{u \leq \alpha(x)\}} du \right) g(x)dx = \frac{\int_{\mathbb{R}^d} f(x)dx}{c} = \frac{1}{c}.$$

Montrons à présent que T suit une loi géométrique sur $\mathbb{N}^*$ de paramètre p. La suite $(X_n, U_n)_{n \in \mathbb{N}^*}$ étant formée de couples indépendants et identiquement distribués, on a pour tout $n \in \mathbb{N}^*$,

$$\begin{aligned}
\mathbb{P}(T > n) &= \mathbb{P}\left(\bigcap_{k=1}^{n} \{U_k > \alpha(X_k)\} \right) \\
&= \prod_{k=1}^{n} \mathbb{P}(U_k > \alpha(X_k)) \\
&= \mathbb{P}(U_1 > \alpha(X_1))^n \\
&= (1-p)^n.
\end{aligned}$$

On en déduit que la fonction de répartition de la variable aléatoire T est bien celle d'une loi géométrique, d'où le résultat. En particulier, T est presque finie.
Il reste à montrer que le vecteur aléatoire X_T a bien la fonction f pour densité. Soit $\varphi : \mathbb{R}^d \to \mathbb{R}$ une fonction mesurable positive ou bornée. Alors,

pour tout $n \in \mathbb{N}^*$, on a

$$\begin{aligned}\mathbb{E}\left[\varphi\left(X_n\right)\mathbf{1}_{\{U_n\leq\alpha(X_n)\}}\right] &= \int_{\mathbb{R}^d}\left(\int_0^1 \mathbf{1}_{\{u\leq\alpha(x)\}}du\right)\varphi(x)g(x)dx \\ &= \int_{\mathbb{R}^d}\alpha(x)\varphi(x)g(x)dx \\ &= \frac{1}{c}\int_{\mathbb{R}^d}\varphi(x)f(x)dx,\end{aligned}$$

d'où l'identité suivante :

$$\mathbb{E}\left[\varphi\left(X_n\right) \mid U_{\mathrm{n}} \leq \alpha\left(X_n\right)\right] = \mathbb{E}[\varphi(Y)]$$

où Y est un vecteur aléatoire ayant pour densité la fonction f : on en déduit que la loi conditionnelle de X_n sachant l'événement $\{U_n \leq \alpha\left(X_n\right)\}$ a pour densité f, et ce pour tout $n \in \mathbb{N}^*$. Enfin, on calcule $\mathbb{E}\left[\varphi\left(X_T\right)\right]$ en utilisant cette propriété pour aboutir à la dernière ligne ci-dessous :

$$\begin{aligned}\mathbb{E}\left[\varphi\left(X_T\right)\right] &= \sum_{n=1}^{+\infty}\mathbb{E}\left[\varphi\left(X_n\right)\mathbf{1}_{T=n}\right] \\ &= \sum_{n=1}^{+\infty}\mathbb{E}\left[\varphi\left(X_n\right)\mathbf{1}_{\{U_n\leq(\alpha X_n)\}}\mathbf{1}_{\cap_{k=1}^{n-1}\{U_k>(\alpha X_k)\}}\right] \\ &= \sum_{n=1}^{+\infty}\mathbb{E}\left[\varphi\left(X_n\right)\mathbf{1}_{\{U_n\leq(\alpha X_n)\}}\right]\times\mathbb{P}\left(\bigcap_{k=1}^{n=1}\{U_k>(\alpha X_k)\}\right) \\ &= \sum_{n=1}^{+\infty}\mathbb{E}\left[\varphi\left(X_n\right) \mid U_n \leq \alpha\left(X_n\right)\right]\times p(1-p)^{n-1} \\ &= \sum_{n=1}^{+\infty}\mathbb{E}[\varphi(Y)]\times p(1-p)^{n-1} \\ &= \quad \mathbb{E}[\varphi(Y)].\end{aligned}$$

Ce qui donne la conclusion désirée. A noter que pour obtenir les troisième et quatrième lignes, nous avons utilisé le fait que les couples de la suite $(X_n, U_n)_{n\in\mathbb{N}^*}$ étaient indépendantes et identiquement distribuées. La démonstration est enfin achevée. □

2.2.2.3 La méthode de Box-Muller

La distribution normale (ou loi Gaussienne) est certainement la plus célèbre des distributions non-uniformes, mais elle est également l'une des plus utilisées, étant donné la difficulté de trouver pour elle une mise en œuvre simple

dans le cadre des méthodes universelles (inversion ou rejet), un important effort scientifique a été consenti durant les cinquante dernières années pour trouver des méthodes spécifiques à la gaussienne qui soient adéquates (voir [16]).
La méthode de Box-Muller (George Edward Pelham Box et Mervin Edgar Muller, 1958) consiste à générer des paires de nombres aléatoires à distribution normale centrée réduite, à partir d'une source de nombres aléatoires de loi uniforme.La transformation prend communément deux formes :

- La forme simple transforme des coordonnées cartésiennes uniformément distribuées dans le cercle unité en des coordonnées normalement distribuées,
- La forme polaire transforme des coordonnées polaires uniformément distribuées en des coordonnées cartésiennes normalement distribuées.

Cette méthode repose sur le lemme 2 :

Lemme 2.
Si U et V sont de loi uniforme sur $\mathcal{U}([0;1])$ et indépendantes alors $\sqrt{-2\log(U)}\cos(2\pi V)$ et $\sqrt{-2\log(U)}\sin(2\pi V)$ est un vecteur gaussien centré de matrice de covariance

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$\big($donc les deux composantes sont indépendantes et de loi $\mathcal{N}(0,1)\big)$.

Preuve.
Soit $f \in \mathbb{C}_b^+(\mathbb{R}^2)$ $\big($où $\mathbb{C}_b^+(\mathbb{R}^2)$ est l'ensemble des fonctions positives, continues et bornées de $\mathbb{R}^2$ dans $\mathbb{R}\big)$. Notons X et Y deux variables indépendantes de loi $\mathcal{N}(0,1)$. Nous avons :

$$\mathbb{E}(f(X,Y)) = \int_{\mathbb{R}^2} f(x,y)\frac{e^{-\left(x^2+y^2\right)/2}}{2\pi}dxdy.$$

Nous voulons effectuer le changement de variable $x = r\cos(\theta), y = r\sin(\theta)$ $(r \in \mathbb{R}_+, \theta \in [0;2\pi])$ (c'est le passage usuel en coordonnées polaires). Nous obtenons la matrice jacobienne

$$J_1(x,y) = \begin{bmatrix} \frac{\partial x}{\partial r} & \frac{\partial y}{\partial r} \\ \frac{\partial x}{\partial \theta} & \frac{\partial y}{\partial \theta} \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -r\sin(\theta) & r\cos(\theta) \end{bmatrix},$$

de déterminant $\det(J_1) = r\cos^2(\theta) + r\sin^2(\theta) = r$. Donc

$$\mathbb{E}(f(X,Y)) = \int_{(r,\theta)\in\mathbb{R}_+\times[0;2\pi]} f(r\cos(\theta), r\sin(\theta))\frac{e^{-r^2/2}}{2\pi}|r|drd\theta.$$

Nous effectuons maintenant un nouveau changement de variable :
$r = \sqrt{-2\ln(u)},\ \theta = 2\pi v\ \left((u,v) \in [0;1]^2\right)$. Nous obtenons la matrice jacobienne

$$J_2(r,\theta) = \begin{bmatrix} -\frac{2}{u}\frac{1}{2\sqrt{-2\log(u)}} & 0 \\ 0 & 2\pi \end{bmatrix},$$

de déterminant $\det(J_2) = \frac{2\pi}{u\sqrt{-2\log(u)}}$. Donc

$$\begin{aligned}
\mathbb{E}(f(X,Y)) =& \int_{(u,v)\in[0;1]^2} f(\sqrt{-2\ln(u)}\cos(2\pi v), \sqrt{-2\ln(u)}\sin(2\pi v))\frac{u}{2\pi}\sqrt{-2\ln(u)} \\
&\times \frac{2\pi}{u\sqrt{-2\ln(u)}} dudv \\
=& \int_{(u,v)\in[0;1]^2} f(\sqrt{-2\ln(u)}\cos(2\pi v), \sqrt{-2\ln(u)}\sin(2\pi v)) dudv \\
=& \mathbb{E}(f(\sqrt{-2\ln(U)}\cos(2\pi V), \sqrt{-2\ln(U)}\sin(2\pi V))).
\end{aligned}$$

Ceci est vrai pour toute fonction $f \in \mathbb{C}_b^+(\mathbb{R}^2)$,
donc (X,Y) et $\left(\sqrt{-2\ln(u)}\cos(2\pi v),\ \sqrt{-2\ln(u)}\sin(2\pi v)\right)$ ont même loi.

□

Remarque 6.
Pour simuler une variable X de loi $\mathcal{N}(0,1)$, il suffit donc de prendre U et $V \sim \mathcal{U}([0;1])$ indépendantes et poser $X = \sqrt{-2\ln(U)}\cos(2\pi V)$. Pour simuler une variable aléatoire X de loi $\mathcal{N}(\mu,\sigma^2)$, il suffit de prendre $X = \mu + \sigma Y$ avec $Y \sim \mathcal{N}(0,1)$.

2.2.3 Vérification par histogramme

Quand on simule une variable aléatoire réelle à l'aide d'une des méthodes ci-dessus, on peut vérifier empiriquement que la loi simulée est bien celle que l'on voulait. Soient $X_1, X_2, \ldots$ des variable aléatoires réelles indépendantes et identiquement distribuées de loi de densité f (avec une dérivée f' vérifiant $\|f'\|_\infty < C$ pour une certaine constante C). Pour tout $a < b$

$$\frac{1}{n(b-a)} \sum_{i=1}^{n} \mathbf{1}_{[a;b]}(X_i) \underset{n\to+\infty}{\overset{p.s.}{\to}} \frac{\mathbb{P}(X_1 \in [a;b])}{b-a}$$

$$\text{et} \quad \frac{\mathbb{P}(X_1 \in [a;b])}{b-a} = \frac{1}{b-a}\int_a^b f(t)dt \quad \text{donc}$$

$$\begin{aligned}
\left|\frac{\mathbb{P}(X_1 \in [a;b])}{b-a} - f(a)\right| &= \left|\frac{1}{b-a}\int_a^b \big(f(t) - f(a)\big)dt\right| \\
&\leq \frac{1}{b-a}\int_a^b |f(t) - f(a)|dt \\
&\leq \frac{1}{b-a}\int_a^b C|t-a|dt \\
&= \frac{C(b-a)}{2} \underset{b\searrow a}{\longrightarrow} 0.
\end{aligned}$$

Donc l'histogramme (correctement renormalisé) des n valeurs simulées $X_1, X_2, \ldots, X_n$ est proche de la densité f (voir [16]).

Chapitre 3

Tests de génération des nombres

Posséder une très grande période et l'atteindre est, pour un générateur de nombres pseudo-aléatoire, une condition nécessaire à remplir mais pas suffisante. En effet, un générateur peut satisfaire cette condition sans pour autant fournir des nombres présentant un caractère aléatoire.
La question qui se pose maintenant est comment savoir qu'une suite de nombres fournis par un générateur a ou non un caractère aléatoire. D'une manière plus précise, il s'agit de vérifier si les nombres donnés par le générateur en question peuvent être considérés comme des réalisations indépendantes d'une variable aléatoire réelle suivant la loi uniforme continue sur l'intervalle $[0, 1]$.
S'agissant du domaine de l'aléatoire, la vérification ne peut être réalisée que par le biais de tests d'hypothèses. Les hypothèses à tester ici concernent deux aspects en même temps : l'uniformité et l'indépendance.
Plusieurs tests sont développés dans la littérature statistique. Ils n'ont pas la même puissance et ne concernent souvent qu'un seul aspect du problème : l'uniformité ou l'indépendance. Aussi il convient en général d'utiliser plusieurs tests pour accepter un générateur comme un bon générateur et l'utiliser par la suite pour produire des nombres aléatoires.
Dans ce qui suit, nous présentons en détails quelques tests. Il s'agit du tracé de l'histogramme et du test de Kolmogrov-Smirnov pour vérifier l'uniformité et du test d'indépendance d'auto-corrélation.

3.1 Test d'uniformité : L'histogramme

La vérification la plus simple de l'hypothèse de la distribution uniforme consiste à tracer un histogramme d'un échantillon provenant d'un générateur.

Une distribution rectangulaire qui apparait parfaitement est souvent le symptôme d'un ordre trop important dans le générateur, tandis que les lacunes dans l'histogramme sont également un indicateur que le générateur ne respecte pas la distribution uniforme exigée.

Exemple 3. *Les séquences de nombres pseudo-aléatoires représentées dans* x_1, x_2 *et* x_3 *:*
Ces graphiques sont tracés par un générateur de congruence multiplicative $x_1 = GCM(a = 32377, c = 0, m = 32378, n = 1000)$, $x_2 = GCM(a = 41, c = 0, m = 2000, n = 1000)$ et $x_3 = runif(1000)$, $runif$ est une fonction qui génère des valeurs aléatoires à partir d'une distribution uniforme dans *rstudio*. L'histogramme x_1 présente une mauvaise répartition des séquences

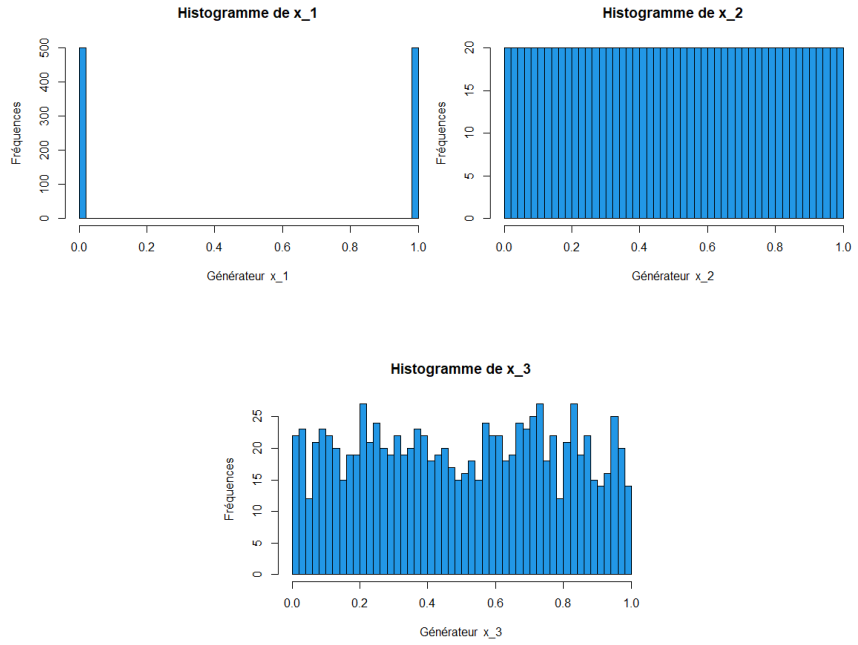


FIGURE 3.1 – Histogrammes de trois ensembles de nombres pseudo-aléatoires

de nombres pseudo-aléatoires, les histogrammes x_2 et x_3 réussissent le test bien que l'histogramme x_3 est en fait trop parfaite pour être crédible.

Parfois, l'utilisation d'un test de qualité d'ajustement de kolmogorov-smirnov (voir [2]) est préconisée, mais cela serait généralement une perte de temps et d'efforts parce qu'un coup d'œil sur l'histogramme est tout ce qui est vraiment nécessaire pour vérifier l'uniformité. De plus, la question la plus importante

et difficile est liée à l'indépendance de la séquence de valeurs générées. Il faut plus de soin pour évaluer cet aspect d'un générateur (voir [7]).

3.2 Test d'indépendance

Il existe dans la littérature plusieurs types de tests d'indépendance. Nous présentons dans ce qui suit la visualisation de la sortie d'un générateur avec un tracé de décalages et la fonction d'auto-corrélation (voir [7]).

3.2.1 Visualisation de la sortie d'un générateur avec un tracé de décalage

Nous pouvons avoir une idée de l'indépendance si nous générons des nombres aléatoires en traçant des paires successives de nombres avec un tracé d'un décalage. Des modèles spécifiques indiquent que le générateur ne produit pas de nombres indépendants, si on voit les points s'aligner, notre générateur est en panne.

Exemple 4. $x_4 = GCM(x_0 = 12, a = 15, m = 511, n = 100)$

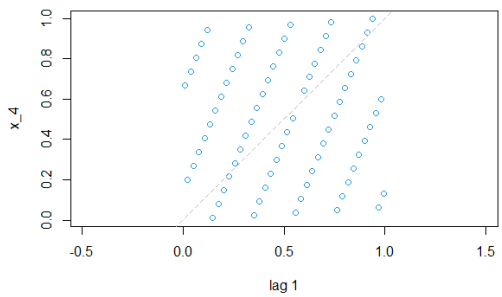


FIGURE 3.2 – Diagramme de décalage pour un générateur de nombres pseudo-aléatoires avec de mauvaises propriétés

En observant le graphique, nous voyons des points situés sur des lignes clairement séparées dans le tracé de décalage affiché à la figure 3.2. Ce générateur est très pauvre.

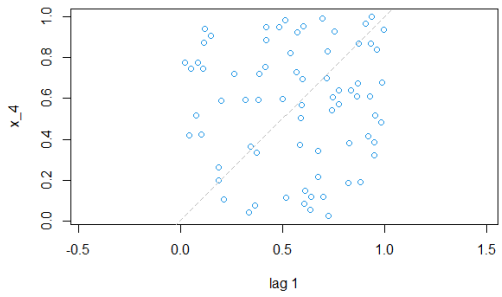


FIGURE 3.3 – Diagramme de décalage pour un générateur de petite période m de nombres pseudo-aléatoires avec de bonnes propriétés

La figure 3.3 montrent les points tracés semblent être plus aléatoires dans ce deuxième exemple. En modifiant le paramètre a, nous obtenons la figure 3.4.

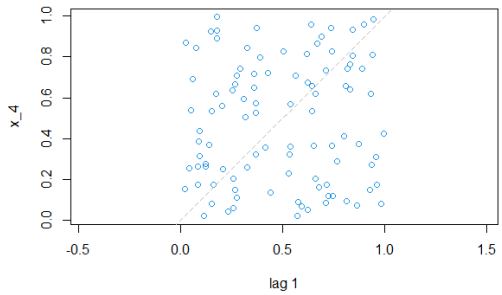


FIGURE 3.4 – Diagramme de décalage pour un générateur avec m plus grande de nombres pseudo-aléatoires avec de bonnes propriétés

Dans notre troisième exemple, nous prenons une valeur beaucoup plus grande de m, ainsi qu'une valeur de a qui est de l'ordre de racine carrée de m. Ce type de combinaison peut conduire à un meilleur comportement, et l'image du diagramme de décalage de la figure 3.4 le confirme.

3.2.2 La fonction d'auto-corrélation

La fonction d'auto-corrélation, ACF en anglais, résume numériquement ce qui peut être observée graphiquement sur un tracé de décalages. L'auto-corrélation à un décalage particulier, disons p, est la corrélation entre la $n + p^{ieme}$ valeur et la n^{ieme} valeur, ou plus précisément, la corrélation entre le vecteur $(x_1, \ldots, x_{n-p})$ et le vecteur $(x_p, x_{p+1}, \ldots, x_n)$.
Les auto-corrélations, comme les corrélations, peuvent prendre des valeurs comprises entre -1 et 1, où les valeurs positives indiquent les points tracés se dispersent autour d'une ligne de pente positive, et les valeurs négatives indiquent les points tracés se dispersent autour d'une ligne de pente négative.

Exemple 5.
Observons les 6 premiers tracés de retard pour l'une des séquences générées précédemment du générateur x_2. En d'autres termes, les figures montrent les tracés des $n + 1^{ers}$ éléments par rapport au n^{ieme}, $n + 2^{ieme}$ par rapport au n^{ieme}, $n + 3^{ieme}$ par rapport au n^{ieme} et ainsi de suite.

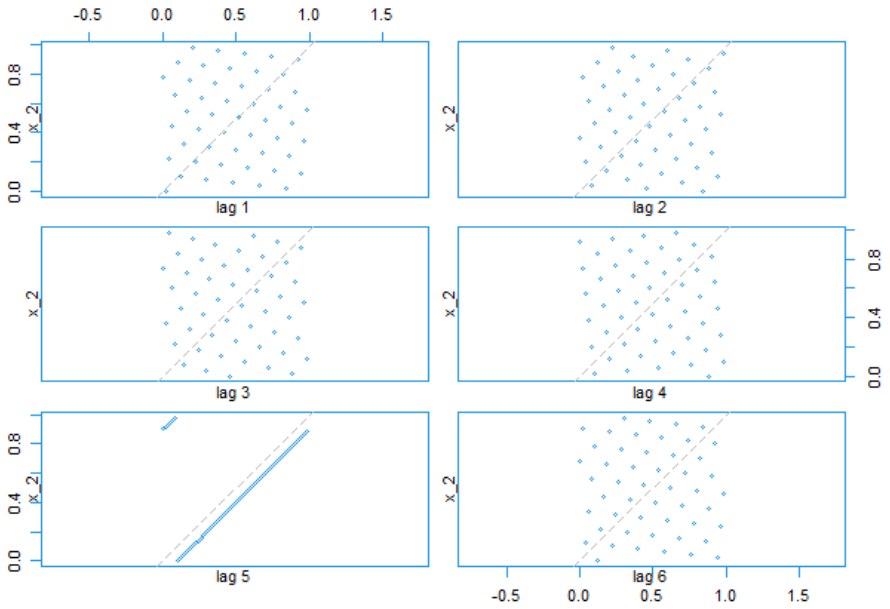


FIGURE 3.5 – Diagrammes de décalage pour les 6 premiers décalages

Aux décalages 1, 2, 3, 4 et 6, il serait difficile de prédire les valeurs actuelles de x_2, mais le tracé du décalage 5 montre que les valeurs actuelles du générateur x_2 dépendent beaucoup de la valeur 5 unités de temps plus tôt.

Les auto-corrélations pour les 5 premiers retards sont représentées dans la figure 3.6.

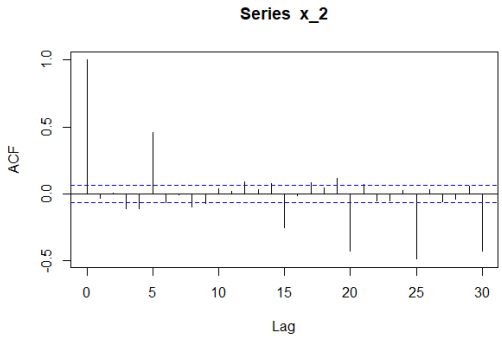


FIGURE 3.6 – Auto-corrélations pour la séquence x_2

Les valeurs de cette fonction d'auto-corrélation sont tracées dans la figure 3.6. Notez la taille du 5^{ieme}. Cela dit que chaque 5^{ieme} valeur de la séquence est dépendante. Observons les auto-corrélation de la sequence x_3.

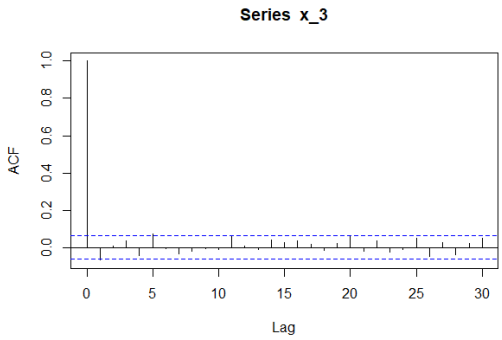


FIGURE 3.7 – Auto-corrélations pour la séquence x_3

Nous observons bien que toutes les valeurs de x_3 sont situées à l'intérieur des pointillés bleus (ce qui constitue l'intervalle de confiance) sur la figure 3.7. Ce modèle réussit le test.

Chapitre 4

Application Numérique

Dans ce chapitre, nous allons d'abord générer des séquences de nombres aléatoires à partir des méthodes énumérées dans le chapitre 2.
Nous allons ensuite soumettre ces séquences de nombres aux tests énoncés dans le chapitre 3 pour discuter de l'uniformité et de l'indépendance de ces nombres.
L'application numérique se fera avec le logiciel R et se basera sur des données simulées et des données réelles.

4.1 Méthodes de la distribution uniforme

Dans chaque exemple de générateurs, nous proposons une visualisation des nombres générés et leurs tests pour vérifier leur uniformité et leur indépendance.

4.1.1 Générateurs de congruence linéaire

- **Exemple du GCL**$(a = 1 + 2^{30}, c = 3, m = 2^{30})$

Puisque les nombres générés ne sont pas entièrement aléatoires mais s'approchent seulement des propriétés idéales des sources complètement aléatoires, leur répartition dans l'espace ne sera pas parfait mais peut s'approcher de l'aléa parfait comme le montre la figure 4.1.
En appliquant le test d'uniformité par l'histogramme, nous voyons bien que ces nombres sont uniformes comme le montre la figure 4.2a.
Sur la figure 4.2b, nous voyons que tous les coefficients d'auto-corrélations se trouvent en dehors de l'intervalle de confiance, donc sont différents de zero, alors les nombres ne sont pas indépendants.

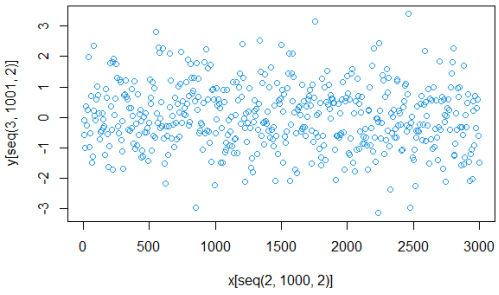
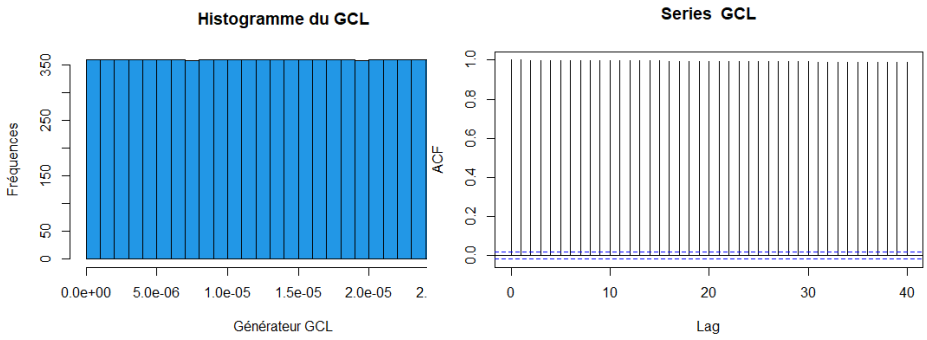


FIGURE 4.1 – Répartition des nombres dans l'espace



(a) test d'uniformité par l'histogramme

(b) test d'indépendance par la fonction ACF

FIGURE 4.2 – Tests des nombres générés par le GCL

4.1.2 Générateurs de congruence multiplicative

- **Exemple du GCM**($a = 65539, c = 0, m = 2^{31}$)

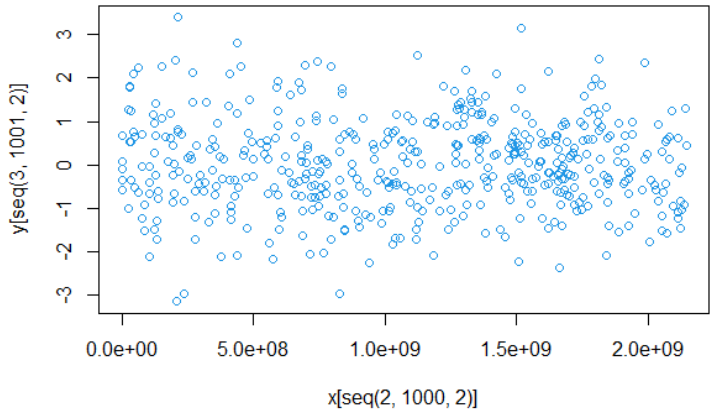
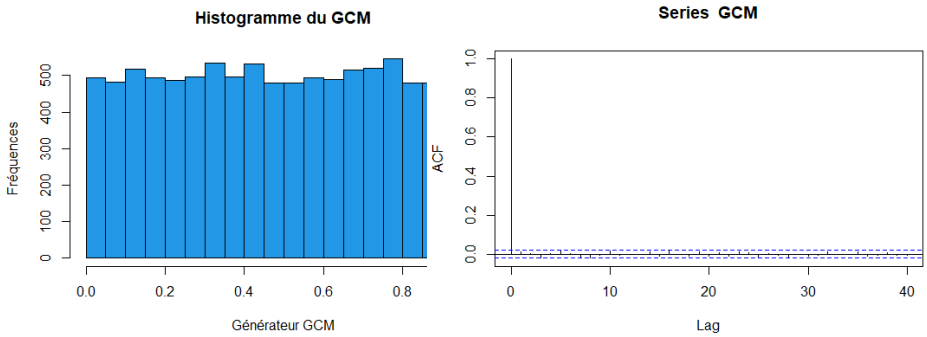


FIGURE 4.3 – Répartition des nombres dans l'espace



(a) test d'uniformité par l'histogramme

(b) test d'indépendance par la fonction ACF1

FIGURE 4.4 – Tests des nombres générés par le GCM

La figure 4.3 nous donne une idée de la répartition des nombres dans l'espace, nous observons bien le caractère aléatoire.
Les nombres générés sont uniformes et indépendants comme le montre la figure 4.4.

4.1.3 Autres générateurs de nombres pseudo-aléatoires

- **Exemple du Générateur de congruence quadratique (GCQ)**

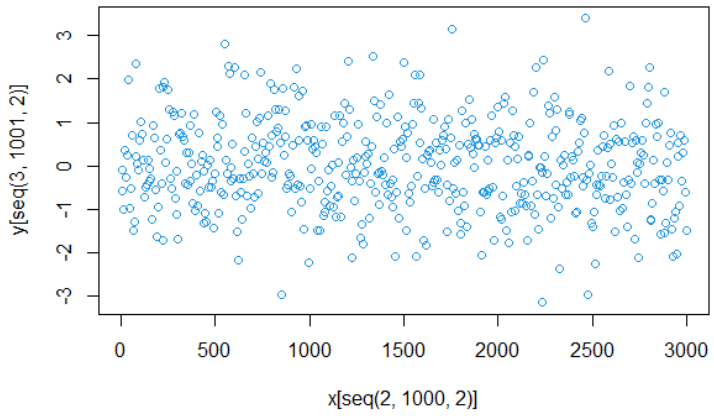


FIGURE 4.5 – Répartition des nombres

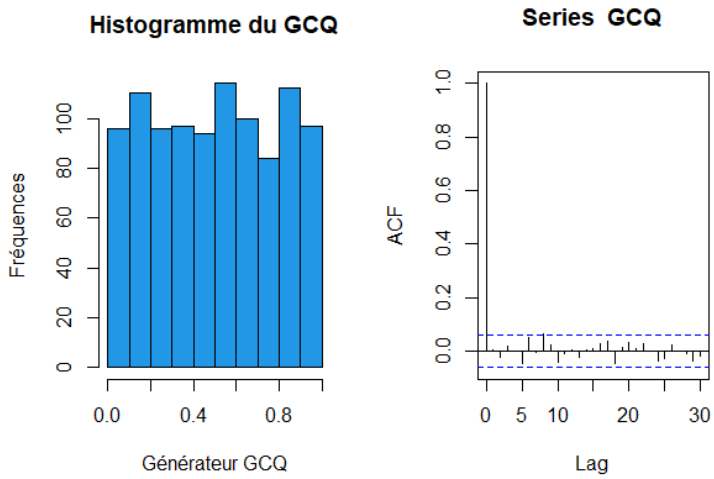


FIGURE 4.6 – Test d'uniformité et d'indépendance par l'histogramme et la fonction ACF

Les nombres sont repartis dans l'espace de manière aléatoire comme le montre la figure 4.5. Ces nombres sont aussi uniformes et indépendants comme le montre la figure 4.6.

- **Exemple du Générateur Wichman et Hill (GWH)**

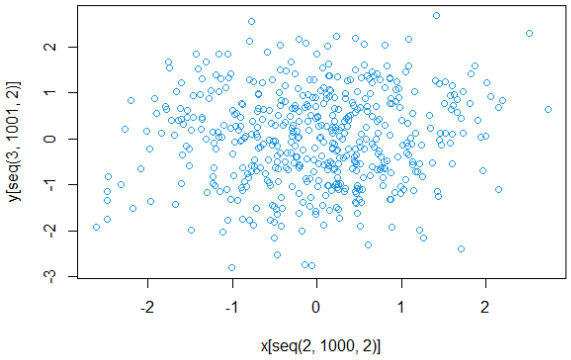


FIGURE 4.7 – Répartition des nombres avec $n = 10^3$

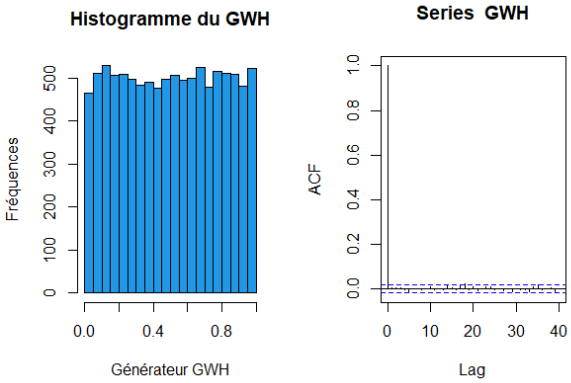


FIGURE 4.8 – Test d'uniformité et d'indépendance par l'histogramme et la fonction ACF

Les nombres sont bien répartis et ils occupent bien l'espace de manière aléatoire comme nous le voyons sur la figure 4.7. Nous voyons bien tous les pics se situent à l'intérieur de l'intervalle de confiance. Les pseudo-aléatoires générés sont indépendants comme nous le voyons sur la figure 4.8.

4.2 Les méthodes de génération des distributions non-uniformes

Nous proposons la simulation des différentes méthodes de génération des distributions non-uniformes avec le tracé de l'histogramme pour la vérification des fonctions cibles que nous nous donnons.

4.2.1 Méthode de l'inverse

Exemple 6. *Simulation de la loi exponentielle*
Nous rappelons que X suit une loi exponentielle de paramètre λ (avec $\lambda > 0$) si pour tout $t \in \mathbb{R}_+$,

$$\mathbb{P}(X > t) = \exp(-\lambda t).$$

Nous noterons cette loi par $\mathcal{E}(\lambda)$. Si F est la fonction de répartition de X, nous avons alors $F(t) = 1 - \exp(-\lambda t)$ (pour $t \geq 0$) et

$$F^{-1}(x) = -\frac{\log(1-x)}{\lambda}.$$

Si $U \sim \mathcal{U}([0;1])$, nous avons, d'après le résultat ci-dessus : $\frac{-\log(1-U)}{\lambda} \sim \mathcal{E}(\lambda)$. Remarquons que $1-U$ a même loi que U et donc

$$-\frac{\log(U)}{\lambda} \sim \mathcal{E}(\lambda).$$

L'application numérique est donnée par la figure 4.9 :

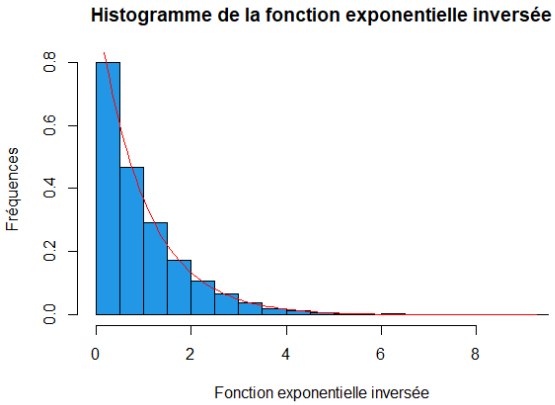


FIGURE 4.9 – Simulation de la fonction d'inverse

Nous pouvons dire en conclusion que nous avons bien obtenu un échantillon distribué de manière exponentielle avec $\lambda = 1$.

4.2.2 Méthode de rejet

La plupart des fonctions de répartition n'ont pas d'inverses connus ou disons leurs inverses ne sont pas faciles à calculer.
Il faut donc avoir d'autres méthodes qui nécessitent pas l'inversion de la fonction de répartition. D'où la méthode de rejet.
Nous nous donnons une fonction cible :

$$f(x) = \frac{\sqrt{2}}{\pi} \times \exp(\frac{-x^2}{2}),\ x \in \mathbb{R},$$

une fonction dont l'inverse n'est pas à calculer explicitement. Et soit g une fonction que nous nous donnons et facile à simuler :

$$g(x) = \exp(-x),\ x \in [0, +\infty] \text{ et } \lambda = 1.$$

L'application numérique est donnée par la figure 4.10 :

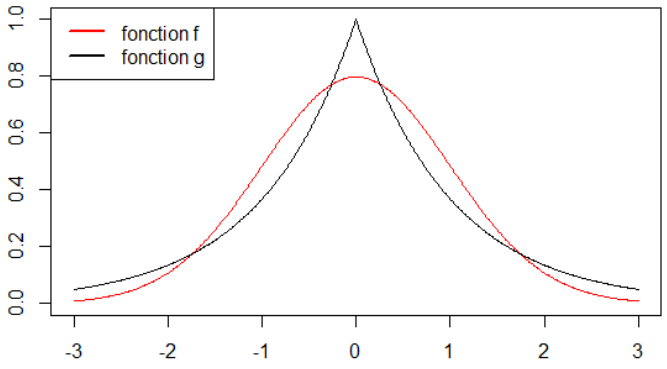


FIGURE 4.10 – Simulation de la fonction cible f et de la fonction g

La méthode consiste à accepter tous les points situés sous la courbe en rouge, donc si la courbe g enveloppe la courbe f, nous réussissons le pari. Nous devons multiplier la densité g par une constante pour nous assurer que la courbe de g sera toujours au-dessus de la courbe de la fonction cible.
Calculons le maximum M_{max} de $\frac{f}{g}$.
Ce calcul se fait comme suit :
Faisons le rapport de f et g, nous dérivons le rapport $\frac{f}{g}$ par rapport à x que nous annulons.
Ce qui donne $x = 1$ puisque la quantité exponentielle est non nulle.

Ce qui donne $M_{max} = \frac{f(1)}{g(1)}$
Alors l'application numérique devient :

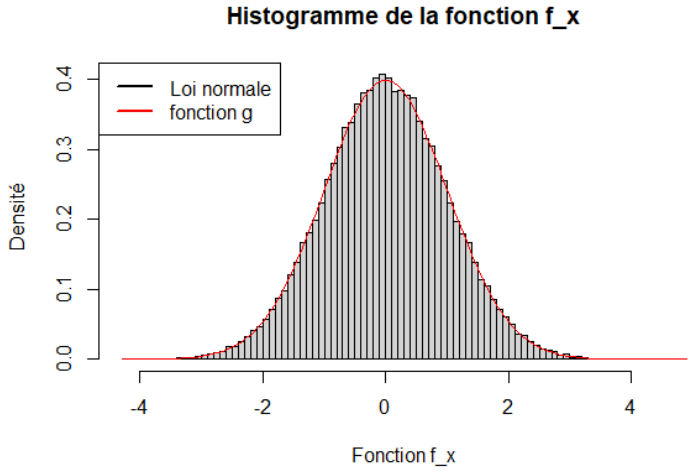


FIGURE 4.11 – simulation complète de la méthode de rejet

Maintenant, nous pouvons voir que les nombres obtenus à partir de cet algorithme semble bien être une distribution normale (ce que nous voyons bien en traçant la courbe de la loi normale) conformément à notre fonction cible.

4.2.3 Méthode de Box-Muller

Nous voulons étudier la façon de générer un échantillon aléatoire à partir de deux distributions normales.
Prenons deux échantillons X et Y à partir d'une distribution normale standard et X et Y sont indépendantes afin que nous puissions ré-écrire la fonction composante comme le produit de deux distributions normales centrées et réduites.
Alors nous avons notre fonction cible suivante :

$$f(x,y) = \frac{1}{2\pi} \exp\left(\frac{-(x^2+y^2)}{2}\right), \text{ avec } (x,y) \in \mathbb{R} \times \mathbb{R}.$$

En appliquant les changements de variables énoncés dans la preuve du lemme 2, la fonction devient :

$$f(r,\theta) = f\big(x(r,\theta), y(r,\theta)\big) \times r.$$

L'application numérique est donnée par la figure 4.12 :

$$\begin{cases} \Theta \sim \mathcal{U}([0; 2\pi]). \\ R \sim \mathcal{E}(\frac{1}{2}). \end{cases}$$

En observant la figure 4.12, les valeurs bleues sont l'histogramme des échan-

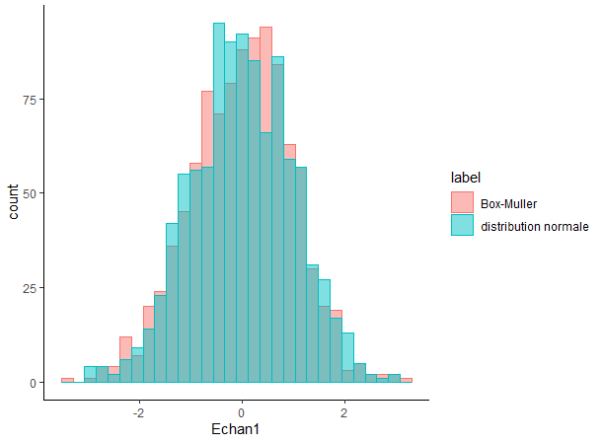


FIGURE 4.12 – simulation de la méthode de Box-muller

tillons générés à partir de la transformation de Box-muller puis les valeurs rouges sont des échantillons générés à partir de la distribution normale standard. Nous pouvons voir que ces deux histogrammes se chevauchent et ils semblent à peu près être normalement distribués.

Conclusion générale

Dans notre travail, nous avons fait l'étude de quelques méthodes et tests de générations des nombres aléatoires. Tous les calculs reposent à la base sur la génération de nombres suivant une distribution uniforme et non-uniforme. L'histoire encore récente sur les méthodes de générations de nombres aléatoires montre que tous les générateurs ne se valent pas et que la façon de les programmer influe significativement sur leurs performances. En d'autres termes, on ne devra pas tenir pour acquis les résultats d'un logiciel ou d'une publication qui ne mentionnent pas clairement quel est le générateur employé et quel germe a été employé afin de garantir une certaine "traçabilité des calculs". D'autre part, les sources des programmes étant rarement accessibles, il faudra envisager de mettre en œuvre des dossiers de validation des générateurs sur les logiciels employés.

Bibliographie

[1] Bastin F. (2013). *Cours de Simulations et modèles*, Année 2012 – 2013, Université de Montréal, disponible sur **http ://www.iro.umontreal.ca/bastin/ift3245/notes.pdf**.

[2] Granger L. (2007). *Cours IF505 Simulation par événements discrets*, Année 2006 – 2007, École Polytechnique de Montréal, disponible sur **http ://www.apprendre-en-ligne.net/random/simu030.pdf**.

[3] Guyader A. (2022). *Cours de Méthode Monte-Carlo*, Année 2021 – 2022, Université de Sorbonne, disponible sur **http ://perso.lpsm.paris/aguyader/files/teaching/MonteCarlo /MonteCarlo.pdf**.

[4] Joulin A. (2020). *Cours de Méthode de Monte-Carlo*, Année 2019 – 2020, INSA Toulouse, disponible sur **http ://perso.math.univ-toulouse.fr/joulin/files/2020/04/Monte-Carlo.pdf**.

[5] Krim M. (2010). *Implémentation des générateurs pseudo-aléatoires : Études et Applications*. Mémoire de magister, Option Systèmes de Communication Modernes (SCM), Université des Sciences et de la Technologie d'Oran (Algérie).

[6] Knuth D.E. (1998). *The Art of Computer Programming : seminumerical algorithms*, Vol 2, Third Edition, Addison-Wesley, Reading, Massachusetts.

[7] Leonelli M. (2021). *Module Simulation and Modelling to Understand Change*, Année 2020 – 2021, School of Human Sciences and Technology at IE University, Madrid, Spain, disponible sur **http ://bookdown.org/manuele_leonelli/SimBook/**.

[8] Lapeyre B., Pardoux E., Sentis R. (1997). *Méthodes de Monte-Carlo pour les équations de transport et de diffusion*. Mathématiques et Applications, Springer-Verlag.

[9] Law A.M., Kelton W.D. (2000). *Simulation Modeling and Analysis*. McGraw-Hill, New york, Third Edition.

[10] L'Ecuyer P.W. (2006). Random number generation. In *Elsevier Handbooks in Operations Research and Management Science : Simulation*, S.G. Henderson and B.L. Nerson, eds., Elsevier Science, Amsterdam, $55 - 81$.

[11] L'Ecuyer P.W. (1998). Random number generation. In *Handbook on Simulation*, Jerry Banks Ed., Wiley, $93 - 137$.

[12] L'Ecuyer P.W. (1994). Uniform random number generation. *Annals of Operations Research* 53, $77 - 120$.

[13] Mathlouthi H. (2015). *Cours de Méthodes de Simulations*, Année $2014 - 2015$, École Supérieure de Statistique et d'Analyse de l'Information (ESSAIT), disponible sur **http ://nanopdf.com/cours-de-méthodes-de-simulations**.

[14] Mazen Y. (2009). *Modélisation, Simulation et Optimisation des architectures de récepteur pour les techniques d'accès WCDMA*. Doctorat de l'Université Paul Verlaine, Metz, 08 juin 2009.

[15] Norbert M. (1996). L'émergence d'une mathématique du probable au $XVII^e$ siècle, *Revue d'histoire des mathématiques*, Vol.2, $119 - 147$.

[16] Rubenthaler S. (2019). *Cours M1-IM Méthodes de Monte-Carlo*, Année $2018 - 2019$, Université Nice Sophia Antipolis, disponible sur **http ://math.univ-cotedazur.fr/rubentha/enseignement/poly-cours-monte-carlo-m1-im.pdf**.

[17] Tarek O.B. (2008). *Génération de nombres pseudo-aléatoires suivant une distribution non-uniforme par circuits intégrés programmables*. Mémoire de maîtrise en Sciences Appliquées. Ecole polytechnique de Montréal, Canada, Août 2008.