Génération de Valeurs Aléatoires Non Uniformes

On dispose d'un générateur fournissant des v.a. i.i.d. $U(0,1)$.

On veut générer des v.a. selon différentes lois: normale, Weibull, Poisson, etc.
Ou encore d'autres "objets" aléatoires: processus stochastiques, points sur une sphère, matrices aléatoires, arbres aléatoires, etc.

Principales références:

L. Devroye, "Non-Uniform Random Variate Generation", Springer, 1986.
http://cg.scs.carleton.ca/~luc/rnbookindex.html

W. Hörmann, J. Leydold, G. Derflinger, "Automatic Nonuniform Random Variate Generation", Springer-Verlag, 2004.

Propriétés recherchées

Méthode correcte (ou très bonne approximation).

Simple: facile à comprendre et à implanter.

Rapide: temps d'initialisation ("setup"), si requis, et temps marginal par appel.
Parfois: compromis è faire entre les deux.

Mémoire utilisée: on peut souvent améliorer la vitesse au prix de stocker de gros tableaux.

Robuste: l'algorithme doit être précis et efficace pour toutes les valeurs des paramètres qui peuvent nous intéresser.

Compatible avec les méthodes de réduction de variance.
Par exemple, on préfère l'inversion parce que cela facilite la synchronisation lorsque l'on compare deux système ou lorsqu'on veut utiliser des variables de contrôle, ou quasi-Monte Carlo, etc.

On sera prêt à sacrifier un peu la vitesse pour préserver l'inversion. Les méthodes de réduction de la variance nous feront souvent gagner beaucoup plus de vitesse que ce que l'on aura sacrifié pour préserver l'inversion.

Inversion

On veut générer une v.a. X de fonction de répartition F.

Soit $U \sim U(0,1)$ et

$$X = F^{-1}(U) = \min\{x : F(x) \geq U\}.$$

Alors

$$P[X \leq x] = P[F^{-1}(U) \leq x] = P[U \leq F(x)] = F(x),$$

i.e., X a la fonction de répartition voulue.

Avantage de l'inversion: monotone, un seul U pour chaque X.

Désavantage: pour certaines lois, F est très difficile à inverser.
Mais souvent, on peut quand même approximer F^{-1} numériquement.

Exemple: la loi normale

Si $Z \sim N(0,1)$, alors $X = \sigma Z + \mu : N(\mu, \sigma^2)$.
Donc il suffit de savoir générer une $N(0,1)$.
Densité: $f(x) = (2\pi)^{-1/2} e^{-x^2/2}$. Pas de formule pour $F(x)$ ou $F^{-1}(x)$.
Par contre, on sait que quand x est grand, $F(x)$ ressemble à $\tilde{F}(x) = 1 - e^{-x^2/2}$
dont la densité est $\tilde{f}(x) = xe^{-x^2/2}$ et l'inverse est $\tilde{F}^{-1}(u) = \sqrt{-2\ln(1-u)}$.

Exemple: la loi normale

Si $Z \sim N(0,1)$, alors $X = \sigma Z + \mu : N(\mu, \sigma^2)$.
Donc il suffit de savoir générer une $N(0,1)$.
Densité: $f(x) = (2\pi)^{-1/2} e^{-x^2/2}$. Pas de formule pour $F(x)$ ou $F^{-1}(x)$.
Par contre, on sait que quand x est grand, $F(x)$ ressemble à $\tilde{F}(x) = 1 - e^{-x^2/2}$
dont la densité est $\tilde{f}(x) = xe^{-x^2/2}$ et l'inverse est $\tilde{F}^{-1}(u) = \sqrt{-2\ln(1-u)}$.

Idée: pour $x > 0$, approximer $x = F^{-1}(u)$ par $y = \tilde{F}^{-1}(u)$, plus un quotient de 2 polynômes en y (fonction rationnelle) qui approxime la différence. Pour $U > 1/2$:

$$
\begin{aligned}
Y &= \sqrt{-2\ln(1-U)}, \\
X &= Y + \frac{p_0 + p_1 Y + p_2 Y^2 + p_3 Y^3 + p_4 Y^4}{q_0 + q_1 Y + q_2 Y^2 + q_3 Y^3 + q_4 Y^4}
\end{aligned}
$$

Les p_i et q_i sont choisis pour que l'approx. soit excellente pour tout $U > 1/2$.
Si $U < 1/2$, on utilise la symétrie: on calcule X pour $1 - U$ puis on retourne $-X$.

Dans SSJ, on utilise une meilleure approx. de l'inverse, basée sur une approximation rationelle de Chebychev qui donne 16 décimales de précision, et proposée par Blair, Edwards, et Johnson (1976).

Dans SSJ, on utilise une meilleure approx. de l'inverse, basée sur une approximation rationelle de Chebychev qui donne 16 décimales de précision, et proposée par Blair, Edwards, et Johnson (1976).

Chi-deux, gamma, beta, etc.: les choses se compliquent car la forme de F^{-1} depend des paramètres.

Inversion par recherche de racine pour lois continues

Parfois on sait mieux approximer F que F^{-1}.

Pour un U donné, on cherche X tel que $F(X) - U = 0$.

Intervalle borné $[x_{\min}, x_{\max}]$: $F(x_{\min}) = 0$ et $F(x_{\max}) = 1$.

Inversion par recherche de racine pour lois continues

Parfois on sait mieux approximer F que F^{-1}.

Pour un U donné, on cherche X tel que $F(X) - U = 0$.

Intervalle borné $[x_{\min}, x_{\max}]$: $F(x_{\min}) = 0$ et $F(x_{\max}) = 1$.

Recherche binaire pour loi continue;

generate $U \sim U(0,1)$;
let $x_1 = x_{\min}$ and $x_2 = x_{\max}$;
while $(x_2 - x_1 > \epsilon_{\mathrm{x}})$ and $(F(x_2) - F(x_1) > \epsilon_{\mathrm{u}})$ do
    $x = (x_1 + x_2)/2$;
    if $F(x) < U$ then $x_1 = x$ else $x_2 = x$;
    /* Here, X always belongs to $[x_1, x_2]$. */
return $x = (x_1 + x_2)/2$.

À chaque itération, on gagne 1 bit de précision sur la valeur de la racine.
Autrement dit, l'erreur est divisée par 2.

Et si le support de X n'est pas borné?

Choisir $x_{\min} < x_{\max}$ tels que:
$x_{\min} < 0$ si $F(0) > 0$, $x_{\max} > 0$ si $F(0) < 1$,
$F(x_{\min})$ est proche de 0, et $F(x_{\max})$ est proche de 1.

Et si le support de X n'est pas borné?

Choisir $x_{\min} < x_{\max}$ tels que:
$x_{\min} < 0$ si $F(0) > 0$, $x_{\max} > 0$ si $F(0) < 1$,
$F(x_{\min})$ est proche de 0, et $F(x_{\max})$ est proche de 1.

Recherche binaire pour loi continue;

generate $U \sim U(0,1)$;
let $x_1 = x_{\min}$ and $x_2 = x_{\max}$;
while $F(x_2) < U$ do
    $x_1 = x_2$ and $x_2 = 2x_2$; /* valid if $x_2 > 0$ */
while $F(x_1) > U$ do
    $x_2 = x_1$ and $x_1 = 2x_1$; /* valid if $x_1 < 0$ */
while $(x_2 - x_1 > \epsilon_{\mathrm{x}})$ and $(F(x_2) - F(x_1) > \epsilon_{\mathrm{u}})$ do
    $x = (x_1 + x_2)/2$;
    if $F(x) < U$ then $x_1 = x$ else $x_2 = x$;
    /* Invariant: at this stage, X always belongs to $[x_1, x_2]$. */
return $x = (x_1 + x_2)/2$.

Autres méthodes de recherche de racine

Newton-Raphson

generate $U \sim U(0,1)$;
let $x = x_m$;
while $|F(x) - U| > \epsilon_{\mathrm{u}}$ do $x = x - (F(x) - U)/f(x)$;
return x.

Si f' est bornée et monotone, + autres conditions: convergence quadratique; l'erreur est mise au carré à chaque itération (à peu près, lorsqu'on est proche).

Autres méthodes de recherche de racine

Newton-Raphson

generate $U \sim U(0,1)$;
let $x = x_m$;
while $|F(x) - U| > \epsilon_{\mathrm{u}}$ do $x = x - (F(x) - U)/f(x)$;
return x.

Si f' est bornée et monotone, + autres conditions: convergence quadratique; l'erreur est mise au carré à chaque itération (à peu près, lorsqu'on est proche).

Mais attention: peut diverger en général.

Autres méthodes de recherche de racine

Newton-Raphson

generate $U \sim U(0,1)$;
let $x = x_m$;
while $|F(x) - U| > \epsilon_{\mathrm{u}}$ do $x = x - (F(x) - U)/f(x)$;
return x.

Si f' est bornée et monotone, + autres conditions: convergence quadratique; l'erreur est mise au carré à chaque itération (à peu près, lorsqu'on est proche).

Mais attention: peut diverger en général.

Aussi: regula falsi, sécante, algorithme de Brent-Dekker (que je recommande).

Autres méthodes de recherche de racine

Newton-Raphson

generate $U \sim U(0,1)$;
let $x = x_m$;
while $|F(x) - U| > \epsilon_{\mathrm{u}}$ do $x = x - (F(x) - U)/f(x)$;
return x.

Si f' est bornée et monotone, + autres conditions: convergence quadratique; l'erreur est mise au carré à chaque itération (à peu près, lorsqu'on est proche).

Mais attention: peut diverger en général.

Aussi: regula falsi, sécante, algorithme de Brent-Dekker (que je recommande).

Pour accélérer, on peut précalculer et mettre dans un tableau les valeurs de $x_s = F^{-1}(s/c)$ pour $s = 0, \ldots, c$. Si $c = 2^e$, il suffit de regarder les e premiers bits de U pour connaitre l'intervalle $[x_s, x_{s+1}]$ où chercher.

Autres méthodes de recherche de racine

Newton-Raphson

generate $U \sim U(0,1)$;
let $x = x_m$;
while $|F(x) - U| > \epsilon_{\mathrm{u}}$ do $x = x - (F(x) - U)/f(x)$;
return x.

Si f' est bornée et monotone, + autres conditions: convergence quadratique; l'erreur est mise au carré à chaque itération (à peu près, lorsqu'on est proche).

Mais attention: peut diverger en général.

Aussi: regula falsi, sécante, algorithme de Brent-Dekker (que je recommande).

Pour accélérer, on peut précalculer et mettre dans un tableau les valeurs de $x_s = F^{-1}(s/c)$ pour $s = 0, \ldots, c$. Si $c = 2^e$, il suffit de regarder les e premiers bits de U pour connaitre l'intervalle $[x_s, x_{s+1}]$ où chercher.

On peut aussi interpoler F^{-1} dans chacun des intervalles.

Inversion pour les lois discrètes

$p(x_i) = P[X = x_i]$ pour $i = 0, 1, \ldots, k-1$; $\quad F(x) = \sum_{x_i \le x} p(x_i)$.

Générer U, trouver $I = \min\{i | F(x_i) \ge U\}$ et retourner x_I.

Recherche séquentielle;

generate $U \sim U(0,1)$;
let $i = m$;
while $F(x_i) < U$ do $i = i + 1$;
while $F(x_{i-1}) \ge U$ do $i = i - 1$;
return x_i.

x_0	x_1	x_2	x_3	x_4	$\cdots$	x_{k-1}
$F(x_0)$	$F(x_1)$	$F(x_2)$	$F(x_3)$	$F(x_4)$	$\cdots$	$F(x_{k-1})$

Exige $O(k)$ itérations en pire cas.

Recherche binaire (exige $\lfloor \log_2 k \rfloor$ ou $\lceil \log_2 k \rceil$ itérations);

generate $U \sim U(0,1)$;
let $i = 0$ and $j = k$;
while $i < j - 1$ do
$m = \lfloor (i + j)/2 \rfloor$;
if $F(x_{m-1}) < U$ then $i = m$ else $j = m$;
/* Invariant: at this stage, I is in $\{i, \ldots, j-1\}$. */
return x_i.

Exige $\approx \log_2 k$ itérations en pire cas et aussi en moyenne.

Mais un peu plus de travail par itération. Comparaison?

Recherche binaire (exige $\lfloor \log_2 k \rfloor$ ou $\lceil \log_2 k \rceil$ itérations);

generate $U \sim U(0,1)$;
let $i = 0$ and $j = k$;
while $i < j - 1$ do
$m = \lfloor (i + j)/2 \rfloor$;
if $F(x_{m-1}) < U$ then $i = m$ else $j = m$;
/* Invariant: at this stage, I is in $\{i, \ldots, j-1\}$. */
return x_i.

Exige $\approx \log_2 k$ itérations en pire cas et aussi en moyenne.

Mais un peu plus de travail par itération. Comparaison?

Si $k = \infty$, on commence avec un intervalle fini, et on l'agrandit au besoin.

Exemple: $X \sim \text{Poisson}(2500)$ (discuter).

Recherche par index.

On partitionne $(0,1)$ en c intervalles de longueur $1/c$.
Soit $i_s = \inf\{i : F(x_i) \geq s/c\}$ pour $s = 0, \ldots, c$.
Si $s = \lfloor cU \rfloor$, alors $U \in [s/c,\, (s+1)/c)$ et on a $I \in \{i_s, \ldots, i_{s+1}\}$.

Recherche par index.

On partitionne $(0,1)$ en c intervalles de longueur $1/c$.
Soit $i_s = \inf\{i : F(x_i) \geq s/c\}$ pour $s = 0, \ldots, c$.
Si $s = \lfloor cU \rfloor$, alors $U \in [s/c, (s+1)/c)$ et on a $I \in \{i_s, \ldots, i_{s+1}\}$.

Il suffit alors de chercher dans cet intervalle, par rech. séquentielle ou binaire.

Recherche par index (combiné avec recherche séquentielle);

generate $U \sim U(0,1)$;
let $s = \lfloor cU \rfloor$ and $i = i_s$;
while $F(x_i) < U$ do $i = i + 1$;
return x_i.

Recherche par index.

On partitionne $(0,1)$ en c intervalles de longueur $1/c$.
Soit $i_s = \inf\{i : F(x_i) \geq s/c\}$ pour $s = 0, \ldots, c$.
Si $s = \lfloor cU \rfloor$, alors $U \in [s/c,\, (s+1)/c)$ et on a $I \in \{i_s, \ldots, i_{s+1}\}$.

Il suffit alors de chercher dans cet intervalle, par rech. séquentielle ou binaire.

Recherche par index (combiné avec recherche séquentielle);

generate $U \sim U(0,1)$;
let $s = \lfloor cU \rfloor$ and $i = i_s$;
while $F(x_i) < U$ do $i = i + 1$;
return x_i.

Le nombre espéré d'itérations du "while" est environ k/c.
Si on choisit $c \approx k$ ou $c \approx 2k$, par exemple, alors on a un algor. super rapide.
On paye un peu en termes de mémoire.

Méthode alias

Permet de générer des v.a. discrètes en temps $O(1)$ par variable en pire cas, après une initialisation de tableaux qui prend un temps en $O(k)$.

Rapide, mais la transformation de U vers X n'est pas monotone.
Donc ce n'est pas l'inversion!

Composition.

Si F est une combinaison convexe de plusieurs fonctions de répartition (i.e., la loi de X est un mélange de lois):

$$F(x) = \sum_j p_j F_j(x).$$

Générer $J = j$ avec probabilité p_j; puis générer X selon F_J.
Requiert deux uniformes pour chaque v.a.
Exemples: Loi hyperexponentielle, processus de Poisson composé, probit mixte, etc.

Composition.

Si F est une combinaison convexe de plusieurs fonctions de répartition (i.e., la loi de X est un mélange de lois):

$$F(x) = \sum_j p_j F_j(x).$$

Générer $J = j$ avec probabilité p_j; puis générer X selon F_J.
Requiert deux uniformes pour chaque v.a.
Exemples: Loi hyperexponentielle, processus de Poisson composé, probit mixte, etc.

Convolution

$$X = Y_1 + Y_2 + \ldots + Y_n,$$

où les Y_j sont indépendantes, de lois spécifiées.
On génère les Y_j et on somme.

Composition.

Si F est une combinaison convexe de plusieurs fonctions de répartition (i.e., la loi de X est un mélange de lois):

$$F(x) = \sum_j p_j F_j(x).$$

Générer $J = j$ avec probabilité p_j; puis générer X selon F_J.
Requiert deux uniformes pour chaque v.a.
Exemples: Loi hyperexponentielle, processus de Poisson composé, probit mixte, etc.

Convolution

$$X = Y_1 + Y_2 + \ldots + Y_n,$$

où les Y_j sont indépendantes, de lois spécifiées.
On génère les Y_j et on somme.

Exemples: Erlang (somme d'exponentielles de même moyenne), binomiale.

Acceptation/rejet

Technique la plus importante après l'inversion.
Peut fournir une solution efficace lorsque l'inversion est trop difficile ou coûteuse.

On veut générer X selon une densité f. La surface sous f est:

$$\mathcal{S}(f) = \{(x, y) \in \mathbb{R}^2 : 0 \leq y \leq f(x)\}.$$

Proposition. Si (X, Y) est uniforme sur $\mathcal{S}(f)$, alors X a la densité f.

Preuve. Si (X, Y) est uniforme sur $\mathcal{S}(f)$, alors $\mathbb{P}[X \leq x]$ est égal à la surface de $\{(z, y) \in \mathcal{S}(f) : z \leq x\}$, qui est $\int_{-\infty}^{x} f(z)dz$.

Acceptation/rejet

Technique la plus importante après l'inversion.
Peut fournir une solution efficace lorsque l'inversion est trop difficile ou coûteuse.

On veut générer X selon une densité f. La surface sous f est:

$$\mathcal{S}(f) = \{(x, y) \in \mathbb{R}^2 : 0 \leq y \leq f(x)\}.$$

Proposition. Si (X, Y) est uniforme sur $\mathcal{S}(f)$, alors X a la densité f.

Preuve. Si (X, Y) est uniforme sur $\mathcal{S}(f)$, alors $\mathbb{P}[X \leq x]$ est égal à la surface de $\{(z, y) \in \mathcal{S}(f) : z \leq x\}$, qui est $\int_{-\infty}^{x} f(z)dz$.

On va générer (X, Y) uniformément sur $\mathcal{S}(f)$. Comment, si $\mathcal{S}(f)$ est compliqué?

Acceptation/rejet

Technique la plus importante après l'inversion.
Peut fournir une solution efficace lorsque l'inversion est trop difficile ou coûteuse.

On veut générer X selon une densité f. La surface sous f est:

$$\mathcal{S}(f) = \{(x, y) \in \mathbb{R}^2 : 0 \leq y \leq f(x)\}.$$

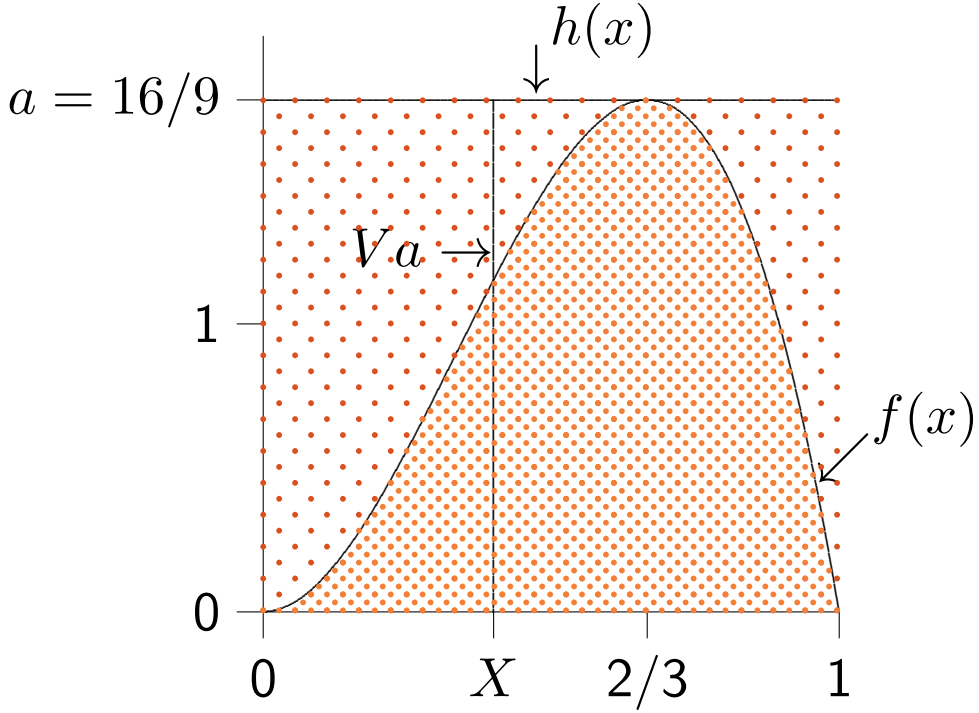
Proposition. Si (X, Y) est uniforme sur $\mathcal{S}(f)$, alors X a la densité f.

Preuve. Si (X, Y) est uniforme sur $\mathcal{S}(f)$, alors $\mathbb{P}[X \leq x]$ est égal à la surface de $\{(z, y) \in \mathcal{S}(f) : z \leq x\}$, qui est $\int_{-\infty}^{x} f(z)dz$.

On va générer (X, Y) uniformément sur $\mathcal{S}(f)$. Comment, si $\mathcal{S}(f)$ est compliqué?

Idée: choisir une surface simple $\mathcal{B}$ qui contient $\mathcal{S}(f)$, puis générer (X, Y) uniformément dans $\mathcal{B}$. Si $(X, Y) \in \mathcal{S}(f)$, c'est bon, sinon on recommence.
On va montrer que le point (X, Y) retenu suit une loi uniforme sur $\mathcal{S}(f)$.

Exemple:
On veut générer $X \sim \text{Beta}(3,2)$, de densité $f(x) = 12x^2(1-x)$ sur $(0,1)$.
La densité est maximale à $x_* = 2/3$. On a $f(2/3) = 16/9 \approx 1.77778$.
Alors on peut prendre $\mathcal{B} = \{(x,y) : 0 \le x \le 1,\ 0 \le y \le 16/9\}$ (un rectangle).



Pour générer un point dans $\mathcal{B}$, on génère deux uniformes indépendantes U et V, et on pose $(X,Y) = (U, aV)$ où $a = 16/9$.

La probabilité que le point soit dans $\mathcal{S}(f)$ est $1/a = 9/16$.
Le nombre espéré de points (X,Y) qu'il faudra générer est $a = 16/9$.

Méthode de rejet générale

On veut générer un point uniformément dans un ensemble $\mathcal{A} \subset \mathbb{R}^d$.

On choisit un ensemble $\mathcal{B}$ plus "simple" tel que $\mathcal{A} \subset \mathcal{B}$.

On génère des points indépendants dans $\mathcal{B}$, et on retient le premier qui tombe dans $\mathcal{A}$.

Proposition. Le point retenu suit la loi uniforme sur $\mathcal{A}$.

Preuve. Pour tout $\mathcal{D} \subseteq \mathcal{A}$, on a $\mathbb{P}[\mathbf{X} \in \mathcal{D}] = \mathrm{vol}(\mathcal{D})/\mathrm{vol}(\mathcal{B})$ et donc

$$\mathbb{P}[\mathbf{X} \in \mathcal{D} \mid \mathbf{X} \in \mathcal{A}] = \frac{\mathbb{P}[\mathbf{X} \in \mathcal{D} \cap \mathcal{A}]}{\mathbb{P}[\mathbf{X} \in \mathcal{A}]} = \frac{\mathrm{vol}(\mathcal{D})/\mathrm{vol}(\mathcal{B})}{\mathrm{vol}(\mathcal{A})/\mathrm{vol}(\mathcal{B})} = \frac{\mathrm{vol}(\mathcal{D})}{\mathrm{vol}(\mathcal{A})}.$$

Ainsi la loi de $\mathbf{X}$ conditionelle à $\mathbf{X} \in \mathcal{A}$ est uniforme sur $\mathcal{A}$.

Méthode de rejet avec fonction chapeau.

Pour générer X selon la densité f, on choisit une autre densité g et une constante $a \geq 1$ telle que

$$f(x) \leq h(x) \stackrel{\text{def}}{=} ag(x)$$

pour tout x, et telle qu'il est facile de générer X selon g.
La fonction h est la fonction chapeau.

Méthode de rejet avec fonction chapeau.

Pour générer X selon la densité f, on choisit une autre densité g et une constante $a \geq 1$ telle que

$$f(x) \leq h(x) \stackrel{\text{def}}{=} ag(x)$$

pour tout x, et telle qu'il est facile de générer X selon g.
La fonction h est la fonction chapeau.

On applique la méthode de rejet avec $\mathcal{A} = \mathcal{S}(f)$ et

$$\mathcal{B} = \mathcal{S}(h) = \{(x, y) \in \mathbb{R}^2 : 0 \leq y \leq h(x)\},$$

la surface sous h.

Algorithme de rejet;

repeat

generate X from the density g and $V \sim U(0, 1)$, independent;

until $Vh(X) \leq f(X)$;

return X.

Proposition. La v.a. X retournée a la densité f.

Algorithme de rejet;

repeat

generate X from the density g and $V \sim U(0,1)$, independent;

until $Vh(X) \leq f(X)$;

return X.

A chaque tour de boucle, la probabilité d'accepter X est $1/a$.
Le nombre R de tours de boucle avant l'acceptation est une v.a. géométrique de paramètre $p = 1/a$.

Le nombre moyen de tours de boucle par v.a. est $1/p = a$.

On veut donc $a \geq 1$ le plus petit possible.

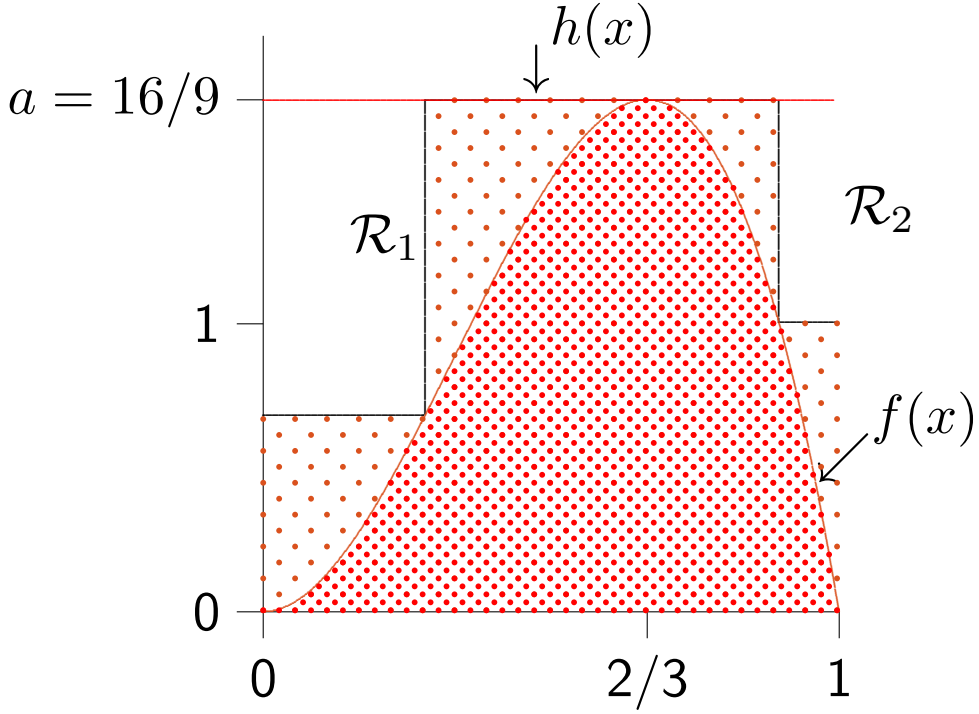
Compromis entre diminuer a et garder g simple.

Exemple: $X \sim \text{Beta}(3, 2)$ (suite).

Pour réduire a, on peut prendre la fonction chapeau:

$$h(x) = \begin{cases} f(x_1) & \text{pour } x < x_1; \\ 16/9 & \text{pour } x_1 \le x \le x_2; \\ f(x_2) & \text{pour } x > x_2, \end{cases}$$

où x_1 et x_2 satisfont $0 < x_1 < 2/3 < x_2 < 1$.



La surface sous h est minimisée en prenant $x_1 = 0.281023$ et $x_2 = 0.89538$. Elle est alors réduite de 1.77778 à 1.38997.

La fonction de répartition inverse de g est linéaire par morceaux.

Pourquoi ne pas prendre une fonction h linéaire par morceaux au lieu de constante par morceaux? Le calcul de G^{-1} demande alors des racines carrées... et on perd en efficacité.

Découpage ("thinning") pour un processus de Poisson

Il s'agit d'une variante de la technique d'acceptation/rejet.

Supposons qu'on a un processus de Poisson non stationnaire avec fonction de taux cumulé $\lambda(t)$ très compliquée (le taux cumulé $\Lambda(t)$ difficile à inverser).

Soit $\overline{\lambda}$ tel que $\lambda(t) \leq \overline{\lambda}$ pour tout t.

Algorithme de découpage ("thinning"):
Générer des pseudo-arrivées selon un processus de Poisson de taux $\overline{\lambda}$.
S'il y a une pseudo-arrivée au temps t, on l'accepte avec probabilité $\lambda(t)/\overline{\lambda}$
(elle devient une vraie arrivée), sinon on la rejette.

Changement de variable

En deux dimensions, soit une bijection $\varphi : \mathbb{R}^2 \to \mathbb{R}^2$. On pose

$$(x, y) = (\varphi_1(u, v),\, \varphi_2(u, v)) = \varphi(u, v)$$

où $\varphi_1 : \mathbb{R}^2 \to \mathbb{R}$ et $\varphi_2 : \mathbb{R}^2 \to \mathbb{R}$. (X, Y) a la densité f ssi (U, V) a la densité

$$t(u, v) = f(\varphi_1(u, v),\, \varphi_2(u, v))|J(u, v)| = f(x, y)|J(u, v)|,$$

où $J(u, v)$ est le Jacobien de la transformation φ, i.e., le déterminant de la matrice des dérivées partielles de x et y en fonction de u et v:

$$J(u, v) = \frac{\partial \varphi_1(u, v)}{\partial u}\frac{\partial \varphi_2(u, v)}{\partial v} - \frac{\partial \varphi_2(u, v)}{\partial u}\frac{\partial \varphi_1(u, v)}{\partial v}.$$

Changement de variable

En deux dimensions, soit une bijection $\varphi : \mathbb{R}^2 \to \mathbb{R}^2$. On pose

$$(x, y) = (\varphi_1(u, v),\ \varphi_2(u, v)) = \varphi(u, v)$$

où $\varphi_1 : \mathbb{R}^2 \to \mathbb{R}$ et $\varphi_2 : \mathbb{R}^2 \to \mathbb{R}$. (X, Y) a la densité f ssi (U, V) a la densité

$$t(u, v) = f(\varphi_1(u, v),\ \varphi_2(u, v))|J(u, v)| = f(x, y)|J(u, v)|,$$

où $J(u, v)$ est le Jacobien de la transformation φ, i.e., le déterminant de la matrice des dérivées partielles de x et y en fonction de u et v:

$$J(u, v) = \frac{\partial\varphi_1(u, v)}{\partial u}\frac{\partial\varphi_2(u, v)}{\partial v} - \frac{\partial\varphi_2(u, v)}{\partial u}\frac{\partial\varphi_1(u, v)}{\partial v}.$$

Idée: générer (U, V) selon la densité t, choisie pour que ce soit plus facile, et calculer $(X, Y) = \varphi(U, V)$ pour avoir un vecteur de densité f.

Changement de variable

En deux dimensions, soit une bijection $\varphi : \mathbb{R}^2 \to \mathbb{R}^2$. On pose

$$(x, y) = (\varphi_1(u, v),\ \varphi_2(u, v)) = \varphi(u, v)$$

où $\varphi_1 : \mathbb{R}^2 \to \mathbb{R}$ et $\varphi_2 : \mathbb{R}^2 \to \mathbb{R}$. (X, Y) a la densité f ssi (U, V) a la densité

$$t(u, v) = f(\varphi_1(u, v),\ \varphi_2(u, v))|J(u, v)| = f(x, y)|J(u, v)|,$$

où $J(u, v)$ est le Jacobien de la transformation φ, i.e., le déterminant de la matrice des dérivées partielles de x et y en fonction de u et v:

$$J(u, v) = \frac{\partial\varphi_1(u, v)}{\partial u}\frac{\partial\varphi_2(u, v)}{\partial v} - \frac{\partial\varphi_2(u, v)}{\partial u}\frac{\partial\varphi_1(u, v)}{\partial v}.$$

Idée: générer (U, V) selon la densité t, choisie pour que ce soit plus facile, et calculer $(X, Y) = \varphi(U, V)$ pour avoir un vecteur de densité f.

En particulier, si t est la densité uniforme sur $\mathcal{C}$, alors (X, Y) suit la loi uniforme sur $\varphi(\mathcal{C})$ ssi $|J(u, v)|$ est constant sur $\mathcal{C}$.

Changement de variable

En deux dimensions, soit une bijection $\varphi : \mathbb{R}^2 \to \mathbb{R}^2$. On pose

$$(x, y) = (\varphi_1(u, v),\ \varphi_2(u, v)) = \varphi(u, v)$$

où $\varphi_1 : \mathbb{R}^2 \to \mathbb{R}$ et $\varphi_2 : \mathbb{R}^2 \to \mathbb{R}$. (X, Y) a la densité f ssi (U, V) a la densité

$$t(u, v) = f(\varphi_1(u, v),\ \varphi_2(u, v))|J(u, v)| = f(x, y)|J(u, v)|,$$

où $J(u, v)$ est le Jacobien de la transformation φ, i.e., le déterminant de la matrice des dérivées partielles de x et y en fonction de u et v:

$$J(u, v) = \frac{\partial\varphi_1(u, v)}{\partial u}\frac{\partial\varphi_2(u, v)}{\partial v} - \frac{\partial\varphi_2(u, v)}{\partial u}\frac{\partial\varphi_1(u, v)}{\partial v}.$$

Idée: générer (U, V) selon la densité t, choisie pour que ce soit plus facile, et calculer $(X, Y) = \varphi(U, V)$ pour avoir un vecteur de densité f.

En particulier, si t est la densité uniforme sur $\mathcal{C}$, alors (X, Y) suit la loi uniforme sur $\varphi(\mathcal{C})$ ssi $|J(u, v)|$ est constant sur $\mathcal{C}$.

Se généralise bien sûr à d dimensions.

Example: Box-Muller.

Générer X selon la loi normale est difficile.
Box et Muller (1958) proposent de générer (X, Y) selon la densité binormale standard

$$f(x, y) = \frac{1}{2\pi} e^{-(x^2+y^2)/2}$$

sur $\mathbb{R}^2$, en changeant les coordonnées (x, y) en coordonnées polaires (r, θ), via $r^2 = x^2 + y^2$ et $\cos\theta = x/r$.

La transformation inverse: $x = r\cos\theta$ et $y = r\sin\theta$.

Example: Box-Muller.

Générer X selon la loi normale est difficile.
Box et Muller (1958) proposent de générer (X, Y) selon la densité binormale standard

$$f(x, y) = \frac{1}{2\pi} e^{-(x^2+y^2)/2}$$

sur $\mathbb{R}^2$, en changeant les coordonnées (x, y) en coordonnées polaires (r, θ), via $r^2 = x^2 + y^2$ et $\cos\theta = x/r$.

La transformation inverse: $x = r\cos\theta$ et $y = r\sin\theta$.

Idée: générer (R, Θ) selon la bonne densité $t(r, \theta)$, puis retourner $(X, Y) = (R\cos\Theta,\ R\sin\Theta)$.

Example: Box-Muller.

Générer X selon la loi normale est difficile.
Box et Muller (1958) proposent de générer (X, Y) selon la densité binormale standard

$$f(x, y) = \frac{1}{2\pi} e^{-(x^2+y^2)/2}$$

sur $\mathbb{R}^2$, en changeant les coordonnées (x, y) en coordonnées polaires (r, θ), via $r^2 = x^2 + y^2$ et $\cos\theta = x/r$.

La transformation inverse: $x = r\cos\theta$ et $y = r\sin\theta$.

Idée: générer (R, Θ) selon la bonne densité $t(r, \theta)$, puis retourner $(X, Y) = (R\cos\Theta,\ R\sin\Theta)$.

On a

$$t(r, \theta) = f(\varphi_1(r, \theta),\ \varphi_2(r, \theta))|J(r, \theta)| = f(r\cos\theta,\ r\sin\theta)|J(r, \theta)|$$

où

$$J(r, \theta) = (\cos\theta)(r\cos\theta) - (\sin\theta)(-\sin\theta) = r(\cos^2\theta + \sin^2\theta) = r.$$

Donc

$$t(r, \theta) = r f(r\cos\theta,\ r\sin\theta) = (r/2\pi)e^{-r^2/2}.$$

Puisque cette densité ne dépend pas de θ, on a $\Theta \sim \text{Uniforme}[0, 2\pi]$.
Puis en intégrant par rapport à θ, on trouve que la densité de R est

$$f_R(r) = re^{-r^2/2} \quad \text{pour } r \geq 0.$$

On a ainsi $\mathbb{P}[R \leq r] = F_R(r) = 1 - e^{-r^2/2}$ et

$$F_R^{-1}(u) = \sqrt{-2\ln(1-u)}.$$

On peut donc facilement générer (R, Θ), puis transformer en (X, Y). On obtient deux normales indépendantes.

Puisque cette densité ne dépend pas de θ, on a $\Theta \sim \text{Uniforme}[0, 2\pi]$.
Puis en intégrant par rapport à θ, on trouve que la densité de R est

$$f_R(r) = re^{-r^2/2} \quad \text{pour } r \geq 0.$$

On a ainsi $\mathbb{P}[R \leq r] = F_R(r) = 1 - e^{-r^2/2}$ et

$$F_R^{-1}(u) = \sqrt{-2\ln(1-u)}.$$

On peut donc facilement générer (R, Θ), puis transformer en (X, Y). On obtient deux normales indépendantes.

Méthode de Box-Muller pour générer deux normales indép.;

Générer $U_1 \sim U(0,1)$ et $U_2 \sim U(0,1)$, indép.;
poser $\Theta = 2\pi U_1$ et $R = \sqrt{-2\ln(1-U_2)}$;
retourner $(X, Y) = (R\cos\Theta,\ R\sin\Theta)$.

Puisque cette densité ne dépend pas de θ, on a $\Theta \sim \text{Uniforme}[0, 2\pi]$.
Puis en intégrant par rapport à θ, on trouve que la densité de R est

$$f_R(r) = re^{-r^2/2} \quad \text{pour } r \geq 0.$$

On a ainsi $\mathbb{P}[R \leq r] = F_R(r) = 1 - e^{-r^2/2}$ et

$$F_R^{-1}(u) = \sqrt{-2\ln(1-u)}.$$

On peut donc facilement générer (R, Θ), puis transformer en (X, Y). On obtient deux normales indépendantes.

Méthode de Box-Muller pour générer deux normales indép.;

Générer $U_1 \sim U(0,1)$ et $U_2 \sim U(0,1)$, indép.;
poser $\Theta = 2\pi U_1$ et $R = \sqrt{-2\ln(1-U_2)}$;
retourner $(X, Y) = (R\cos\Theta,\ R\sin\Theta)$.

Une méthode semblable existe aussi pour la loi de Student.

Par contre, le calcul de sin, cos, log, et sqrt sont coûteux.

Mais considérons le point $(V_1, V_2) = (D\cos\Theta,\ D\sin\Theta)$, où R est remplacé par $D = \sqrt{U_2}$. Ce point, situé à distance D de l'origine, suit la loi uniforme sur le disque de rayon 1 centré à l'origine.

À noter que $X = R\cos\Theta = RV_1/D$ et $Y = R\sin\Theta = RV_2D$.

On peut générer (V_1, V_2) directement en générant des points au hasard dans le carré $[-1, 1]^2$ et en retenant le premier point qui tombe dans le disque, i.e., pour lequel $U_2 = V_1^2 + V_2^2 < 1$. C'est une méthode de rejet.

La probabilité d'accepter est $\pi/4$, et donc il faudra en moyenne $4/\pi \approx 1.273$ tirages pour trouver (V_1, V_2).

Ensuite on calcule $R/D = \sqrt{-2\ln(1-U_2)/U_2}$, $X = V_1R/D$, et $Y = V_2R/D$.

Cet algorithme s'appelle la méthode polaire. Elle élimine le calcul de cos et sin, mais pas log et sqrt.

Rejet avec changement de variable

Original: générer (X, Y) uniforme dans $\mathcal{S}(f)$.
Changement de variable: générer (U, V) uniforme dans $\mathcal{C}$,
et retourner $(X, Y) = \varphi(U, V)$.

Rejet avec changement de variable

Original: générer (X, Y) uniforme dans $\mathcal{S}(f)$.
Changement de variable: générer (U, V) uniforme dans $\mathcal{C}$,
et retourner $(X, Y) = \varphi(U, V)$.

On suppose que $\varphi(\mathcal{C}) = \mathcal{S}(f)$ et que $|J(u, v)| = K$; veut dire que φ transforme chaque morceau de taille ϵ dans $\mathcal{C}$ en un morceau de taille ϵ/K dans $\mathcal{S}(f)$, et vice-versa.

Rejet avec changement de variable

Original: générer (X, Y) uniforme dans $\mathcal{S}(f)$.
Changement de variable: générer (U, V) uniforme dans $\mathcal{C}$,
et retourner $(X, Y) = \varphi(U, V)$.

On suppose que $\varphi(\mathcal{C}) = \mathcal{S}(f)$ et que $|J(u, v)| = K$; veut dire que φ transforme chaque morceau de taille ϵ dans $\mathcal{C}$ en un morceau de taille ϵ/K dans $\mathcal{S}(f)$, et vice-versa.

Rejet dans l'espace (U, V): on cherche un ensemble $\bar{\mathcal{C}}$ contenant $\mathcal{C}$, mais tout juste, tel qu'il est facile de générer uniformément dans $\bar{\mathcal{C}}$.

Méthode de rejet en coordonnées transformées;
    répéter
        générer (U, V) uniformément dans $\bar{\mathcal{C}}$;
    jusqu'à $(U, V) \in \mathcal{C}$;
    retourner $X = \varphi_1(U, V)$.

Proposition. Le X retourné a la densité f.

Rejet avec densité transformée

Soit X de densité f.
Si $\tau : \mathbb{R} \to \mathbb{R}$ et $X = \tau(U)$, alors $U = \tau^{-1}(X)$ a la densité

$$t(u) = f(\tau(u))\tau'(u).$$

Pour générer X selon f, on génère U selon t et on retourne $X = \tau(U)$.

Rejet avec densité transformée

Soit X de densité f.
Si $\tau : \mathbb{R} \to \mathbb{R}$ et $X = \tau(U)$, alors $U = \tau^{-1}(X)$ a la densité

$$t(u) = f(\tau(u))\tau'(u).$$

Pour générer X selon f, on génère U selon t et on retourne $X = \tau(U)$.

Méthode de rejet: générer (U, V) dans la surface $S(t)$ sous t, définie par

$$\mathcal{S}(t) = \varphi^{-1}(\mathcal{S}(f)) = \{(u, v) : 0 \leq v \leq f(\tau(u))\tau'(u)\},$$

en utilisant une fonction chapeau pour la densité t.

Rejet avec densité transformée

Soit X de densité f.
Si $\tau : \mathbb{R} \to \mathbb{R}$ et $X = \tau(U)$, alors $U = \tau^{-1}(X)$ a la densité

$$t(u) = f(\tau(u))\tau'(u).$$

Pour générer X selon f, on génère U selon t et on retourne $X = \tau(U)$.

Méthode de rejet: générer (U, V) dans la surface $S(t)$ sous t, définie par

$$\mathcal{S}(t) = \varphi^{-1}(\mathcal{S}(f)) = \{(u, v) : 0 \leq v \leq f(\tau(u))\tau'(u)\},$$

en utilisant une fonction chapeau pour la densité t.

Si $\tau^{-1} \approx F$ (dist. de X), alors $U \approx U(0, 1)$ et on peut prendre une fonction chapeau uniforme sur $(0, 1)$.

Example (Wallace 1976).

On veut X de densité normale sur $[0, \infty)$: $f(x)$ est proportionnel à $\tilde{f}(x) = \exp(-x^2/2)$, for $x > 0$.

Transformation $\tau : [0, 1) \to [0, \infty)$ définie par

$$\tau(u) = 1.22u(1 + 0.14/(1 - u)).$$

Example (Wallace 1976).

On veut X de densité normale sur $[0, \infty)$: $f(x)$ est proportionnel à $\tilde{f}(x) = \exp(-x^2/2)$, for $x > 0$.

Transformation $\tau : [0, 1) \to [0, \infty)$ définie par

$$\tau(u) = 1.22u(1 + 0.14/(1 - u)).$$

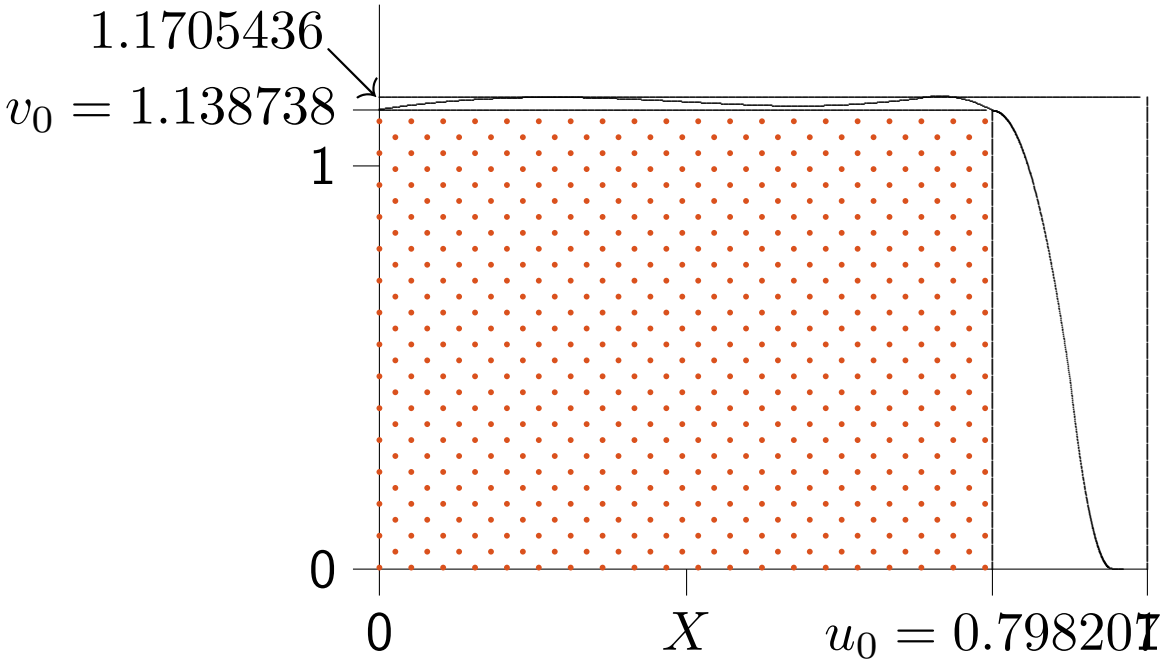
On a $\tau'(u) = 1.22(1 + 0.14/(1 - u)^2)$ et la densité $t(u)$ est proportionnelle à

$$\tilde{t}(u) = \tilde{f}(\tau(u))\tau'(u)/1.22,$$

dont le max. est $\tilde{a} = 1.17054363007$, atteint à $u = 0.73791696460$.

Fonction chapeau: $h(u) = \tilde{a}$.

On peut ajouter un squeeze.



Algorithme: Transformed rejection to generate a positive standard normal;

repeat

generate $U \sim U(0,1)$ and $V \sim U(0,1)$;

let $X = 1.22\, U(1 + 0.14/(1-U))$;

if $U \leq 0.798207256116$ and $V \leq \frac{1.138738153941}{1.17054363007}$ return X;

until $1.17054363007\, V \leq \exp(-X^2/2)(1 + 0.14/(1-u)^2)$;

return X.

Point au hasard sur la surface d'une hypersphère

Sphère de rayon 1 centrée à l'origine, en d dimensions.
Il suffit de génénér un point dans $\mathbb{R}^d$, selon une densité radialement symétrique (i.e., qui ne dépend que de la distance à l'origine).
Par exemple, la densité multinormale standard.

Point au hasard sur la surface d'une hypersphère

Sphère de rayon 1 centrée à l'origine, en d dimensions.
Il suffit de généner un point dans $\mathbb{R}^d$, selon une densité radialement symétrique (i.e., qui ne dépend que de la distance à l'origine).
Par exemple, la densité multinormale standard.

On génère $\mathbf{Z} = (Z_1, \ldots, Z_d) \sim N(\mathbf{0}, \mathbf{I})$; On pose $\mathbf{X} = \mathbf{Z}/\|\mathbf{Z}\|_2$.

Point au hasard sur la surface d'une hypersphère

Sphère de rayon 1 centrée à l'origine, en d dimensions.
Il suffit de génénér un point dans $\mathbb{R}^d$, selon une densité radialement symétrique (i.e., qui ne dépend que de la distance à l'origine).
Par exemple, la densité multinormale standard.

On génère $\mathbf{Z} = (Z_1, \ldots, Z_d) \sim N(\mathbf{0}, \mathbf{I})$; On pose $\mathbf{X} = \mathbf{Z}/\|\mathbf{Z}\|_2$.

En deux dimensions: on peut générer Θ uniformément sur $(0, 2\pi)$, puis poser $(X_1, X_2) = (\cos\Theta, \sin\Theta)$.

Point au hasard sur la surface d'une hypersphère

Sphère de rayon 1 centrée à l'origine, en d dimensions.
Il suffit de généner un point dans $\mathbb{R}^d$, selon une densité radialement symétrique (i.e., qui ne dépend que de la distance à l'origine).
Par exemple, la densité multinormale standard.

On génère $\mathbf{Z} = (Z_1, \ldots, Z_d) \sim N(\mathbf{0}, \mathbf{I})$; On pose $\mathbf{X} = \mathbf{Z}/\|\mathbf{Z}\|_2$.

En deux dimensions: on peut générer Θ uniformément sur $(0, 2\pi)$, puis poser $(X_1, X_2) = (\cos\Theta, \sin\Theta)$.

En trois dimensions: chaque coordonnée X_j est uniforme sur $(-1, 1)$!
On peut générer $X_3 \sim \text{Uniforme}(-1, 1)$, puis (X_1, X_2) sur le cercle qui reste, dont le rayon est $\tilde{r} = \cos(\arcsin x_3)$.
Pour cela, générer $\Theta \sim \text{Uniforme}(0, 2\pi)$, et poser $(X_1, X_2) = (\tilde{r}\cos\Theta, \tilde{r}\sin\Theta)$.