Projet Monte-Carlo

Emma Kopp - Emile Kosseim

Contents

```r
set.seed(1987)
```

Exerice 1

Soit f une densité de $\mathbb{R}^2$ définie pour $(x,y) \in \mathbb{R}^2$ par $f(x,y) = a\psi(x,y)$ avec $a \in \mathbb{R}_+^*$, et

$$\psi(x,y) = \left[\left|\sin\left(\frac{2}{\pi}x^2 - \frac{\pi}{4}\right)\right| + 4\cos(x)^2 + y^4\right] e^{-2(x+|y|)} \mathbb{1}_{\{x \ \in \ [-\frac{\pi}{2},\frac{\pi}{2}]\}} \mathbb{1}_{\{y \ \in \ [-1,1]\}}$$

Objectif: Pour (X,Y) de densité f, on cherche à estimer f_X la densité marginale de X.

Simulation suivant la densité f

Question 1

On peut réecrire la fonction ψ de la façon suivante :

$$\begin{aligned}
&\forall (x,y) \in \mathbb{R}\times\mathbb{R}, \\
&\psi(x,y) = \left[\left|\sin\left(\frac{2}{\pi}x^2 - \frac{\pi}{4}\right)\right| + 4\cos(x)^2 + y^4\right] (e^{\pi} - e^{-\pi})(1-e^{-2})\frac{1}{2} \times \frac{2e^{-2(x+\frac{\pi}{2})}}{1-e^{-2\pi}} \mathbb{1}_{x\in[\frac{-\pi}{2},\frac{\pi}{2}]} \times \frac{e^{-2|y|}}{1-e^{-2}} \mathbb{1}_{y\in[-1,1]} \\
&\quad = \frac{1}{2}(e^{\pi} - e^{-\pi})(1-e^{-2}) \left[\left|\sin\left(\frac{2}{\pi}x^2 - \frac{\pi}{4}\right)\right| + 4\cos(x)^2 + y^4\right] g(x,y)
\end{aligned}$$

où g est la densité du produit d'une loi exponenielle tronquée $\mathcal{E}_{[-\pi/2,\pi/2]}(\ 2\ )$ translatée de $-\frac{\pi}{2}$, et d'une loi de Laplace tronquée $\mathcal{L}_{[-1,1]}(0,\frac{1}{2})$. C'est à dire

$$\forall (x,y) \in \mathbb{R}^2, \quad g(x,y) = \frac{2e^{-2(x+\frac{\pi}{2})}}{1-e^{-2\pi}} \mathbb{1}_{\{x\in[\frac{-\pi}{2},\frac{\pi}{2}]\}} \times \frac{e^{-2|y|}}{1-e^{-2}} \mathbb{1}_{\{y\in[-1,1]\}}$$

On souhaite, à l'aide de cette densité g, utiliser la méthode d'acceptation rejet pour simuler f. On cherche donc une constante $M \geq 1$ telle que

$$\forall (x,y) \in \mathbb{R}^2, \quad f(x,y) \leq Mg(x,y)$$

Ce qui revient à chercher une constante m définie par $M = am$ telle que

$$\forall (x,y) \in \mathbb{R}^2, \quad \psi(x,y) \leq mg(x,y)$$

Étant donné les supports de ψ et g, on définit la fonction h sur $\mathbb{R}\times\mathbb{R}$ tel que :

$$h(x,y) = \begin{cases} \frac{\psi(x,y)}{g(x,y)} & \text{si } (x,y) \in [-\frac{\pi}{2},\frac{\pi}{2}] \times [-1,1] \\ 0 & \text{sinon} \end{cases}$$

Le support de g étant inclus dans celui de ψ, la fonction h est bien définit sur $\mathbb{R}$

On cherche à calculer : $\sup\limits_{(x,y)\in\mathbb{R}\times\mathbb{R}} \left(\frac{\psi(x,y)}{g(x,y)}\right)$

Ce qui revient à calculer :

$$\sup_{(x,y)\ \in\ K} (\ h(x,y)\ ) = \sup_{(x,y)\ \in\ K} \left(\frac{1}{2}(e^{\pi}-e^{-\pi})(1-e^{-2})\left[|\sin\left(\frac{2}{\pi}x^2-\frac{\pi}{4}\right)|+4\cos(x)^2+y^4\right]\right)$$
$$= \frac{1}{2}(e^{\pi}-e^{-\pi})(1-e^{-2}) \sup_{(x,y)\ \in\ K} \left(|\sin\left(\frac{2}{\pi}x^2-\frac{\pi}{4}\right)|+4\cos(x)^2+y^4\right)$$

avec $K = [-\frac{\pi}{2}, \frac{\pi}{2}] \times [-1,1]$

La fonction $(x,y) -> |\sin\left(\frac{2}{\pi}x^2-\frac{\pi}{4}\right)|+4\cos(x)^2+y^4$ est continue sur K qui est un ensemble fermé borné, et donc le sup est atteint.

On a :

$$\begin{aligned}
\sup_{(x,y)\ \in\ K} \left(|\sin\left(\frac{2}{\pi}x^2-\frac{\pi}{4}\right)|+4\cos(x)^2+y^4\right) &= \sup_{x\ \in\ [-\frac{\pi}{2},\frac{\pi}{2}]} \left(|\sin\left(\frac{2}{\pi}x^2-\frac{\pi}{4}\right)|+4\cos(x)^2\right) + \sup_{y\ \in\ [-1,1]} (y^4) \\
&= \sup_{x\ \in\ [-\frac{\pi}{2},\frac{\pi}{2}]} \left(|\sin\left(\frac{2}{\pi}x^2-\frac{\pi}{4}\right)|+4\cos(x)^2\right) + 1 \\
&\leq \sup_{x\ \in\ [-\frac{\pi}{2},\frac{\pi}{2}]} \left(|\sin\left(\frac{2}{\pi}x^2-\frac{\pi}{4}\right)|\right)\ +\ \sup_{x\ \in\ [-\frac{\pi}{2},\frac{\pi}{2}]} (4\cos(x)^2) + 1 \\
&= \sup_{x\ \in\ [-\frac{\pi}{2},\frac{\pi}{2}]} \left(|\sin\left(\frac{2}{\pi}x^2-\frac{\pi}{4}\right)|\right) + 4 + 1 \\
&=\ 5\ +\ \sup_{x\ \in\ [-\frac{\pi}{2},\frac{\pi}{2}]} \left(|\sin\left(\frac{2}{\pi}x^2-\frac{\pi}{4}\right)|\right)
\end{aligned}$$

De plus, par parité on peut se restreindre à $x\ \in\ [0,\frac{\pi}{2}]$, et on a, en utilisant la croissante de *sin* sur $[0,\frac{\pi}{2}]$ que

$$0 \leq x \leq \frac{\pi}{2} \implies 0 \leq x^2 \leq \frac{\pi^2}{4} \implies -\frac{\pi}{4} \leq \frac{\pi}{2}x^2 - \frac{\pi}{4} \leq \frac{\pi}{4} \implies -\frac{\sqrt(2)}{2} \leq \sin\left(\frac{\pi}{2}x^2-\frac{\pi}{4}\right) \leq \frac{\sqrt(2)}{2} \implies \left|\sin\left(\frac{\pi}{2}x^2-\frac{\pi}{4}\right)\right| \leq \frac{\sqrt(2)}{2}$$

Ainsi

$$\sup_{x\ \in\ [-\frac{\pi}{2},\frac{\pi}{2}]} \left(|\sin\left(\frac{2}{\pi}x^2-\frac{\pi}{4}\right)|\right) \leq \frac{\sqrt(2)}{2}$$

Et donc

$$\sup_{(x,y)\ \in\ K} \left(|\sin\left(\frac{2}{\pi}x^2-\frac{\pi}{4}\right)|+4\cos(x)^2+y^4\right) \leq 5 + \frac{\sqrt(2)}{2}$$

On pose donc

$$m := \frac{1}{2}(e^{\pi}-e^{-\pi})(1-e^{-2})\left(5+\frac{\sqrt(2)}{2}\right)$$

Question 2

Pour simuler suivant la loi exponentielle translatée et tronquée $Exp_{[\frac{-\pi}{2},\frac{\pi}{2}]}(\lambda)$ à partir d'un générateur uniforme $[0,1]$, on utilise la méthode de la fonction inverse car on est dans le cas d'une loi continue. La fonction de répartition de cette loi est donnée par :

$$F_{exp}(x) = \frac{1 - e^{-2(x + \frac{pi}{2})}}{1 - e^{-2\pi}}$$

On en déduit que F_{exp} est une fonction bijective de $\mathbb{R}$ dans $\mathbb{R}$. L'inverse généralisée de F_{exp} coïncide donc avec l'inverse au sens de la bijection et est donnée par :

$$F_{exp}^{-1}(y) = -\frac{1}{2}\ln(1 - (1 - e^{-2\pi})y) - \frac{\pi}{2}$$

```r
# ------------------ Loi Exponentielle Tronquée et Translatée ----------------

# Densité
dexp_trans_tronq <- function(x, rate = 2, trans = -pi / 2, tronq = pi / 2) {
  return(rate * exp(-rate * (x - trans)) * (abs(x) <= tronq) / (1 - exp(-2 * pi)))
}

# Fonction Inverse
inv_F_exp_tr <- function(x) {
  return((-0.5 * log(exp(pi) - (x * (1 - exp(-2 * pi)) / exp(-pi)))) * abs(x < pi / 2))
}

# Générateur exponentielle
rgen_exp_tr <- function(n) {
  return(inv_F_exp_tr(runif(n)))
}
```

Pour simuler une lois de Laplace tronquée, on considère X une variable aléatoire de densité de Laplace de paramètre $mu = 0$ et $b = 1/2$ et $U \sim \mathcal{U}[0,1]$. On sait que $-1 \leq X \leq 1$ a même loi que

$$F^{\leftarrow} = [F(-1) + \{F(1) - F(-1)\}U]$$

L'expression de la fonction de répartition d'une $\mathcal{L}(0, \frac{1}{2})$ est :

$$F_{\mathcal{L}(\mu,b)}(x) = \frac{1}{2}[1 + sgn(x - \mu)(1 - e^{-\frac{|x-\mu|}{b}})]$$

$F_{\mathcal{L}(\mu,b)}$ est une fonction bijective de $\mathbb{R}$ dans $\mathbb{R}$. L'inverse généralisée de $F_{\mathcal{L}(\mu,b)}$ coïncide donc avec l'inverse au sens de la bijection et est donnée par :

$F^{-1}_{\mathcal{L}(\mu,b)}(y) = \mu - bsgn(y - 0.5)\ln(1 - 2|y - 0.5|)$

On peut simuler $-1 \leq X \leq 1$ numériquement.

```r
# ------------------ Loi Laplace Tronquée ----------------------

#  Fonction répartition
F_rep_laplace <- function(x, mu=0, b=1/2) {
return(0.5 * (1 + sign(x - mu) * (1 - exp(-abs(x - mu) / b))))
}

# Densité
dlaplace_tronq <- function(x, mu = 0, b = 1/2) {
  return(((0.5 * (1 / b) * exp(-(abs(mu - x)) / b)) * (abs(x) <= 1)) /
```

```
          (F_rep_laplace(1) - F_rep_laplace(-1)))
}

# Méthode de la fonction inverse pour densité de laplace
inv_F_Laplace <- function(x, mu = 0, b = 1/2) {
  return(mu - b * sign(x - 0.5) * log(1 - 2 * abs(x - 0.5)))
}

# Générateur
rgen_Laplace_tronq <- function(n) {
  return(inv_F_Laplace(F_rep_laplace(-1) +
                         (F_rep_laplace(1) - F_rep_laplace(-1))
                       * runif(n)))
}
```

On en déduit le générateur suivant g:

```
# Générateur suivant g
rgen_g <- function(n) {
  return(cbind(rgen_exp_tr(n), rgen_Laplace_tronq(n)))
}
```

On définit les fonctions g, le générateur de g et psi

```
# Densité g
g <- function(x, y) {
  return(dexp_trans_tronq(x) * dlaplace_tronq(y))
}

# Fonction génératrice de g
rgen_g <- function(n) {
  return(cbind(rgen_exp_tr(n), rgen_Laplace_tronq(n)))
}

psi <- function(x, y) {
  return((abs(sin((2 / pi) * x^2 - pi / 4)) + 4 * cos(x) * cos(x) + y^4) *
    exp(-2 * (x + abs(y))) * (abs(x) <= pi / 2) * (abs(y) <= 1))
}
```

On peut donc définir la fonction ρ

```
m <- (5 + sqrt(2)/2) * (exp(pi) - exp(-pi)) * (1 - exp(-2)) * .5

rho_top <- function(x, y) {
  return(psi(x, y) / (m * g(x, y)))
}
```

Nous allons utiliser la méthode d'acceptation rejet. Elle nous permet de :

1. Calculer le temps de calcul de chacune des simulations. On note cette variable aléatoire T.
2. Simuler n variables aléatoire selon la densité f.

On introsuit une fonction nomée calcul_temps permettant d'extraire les simulations de T, notée $(t_1, ..., t_n)$.

```r
calcul_temps <- function(coor) {
  # on crée un vecteur décalé de 1 indice
  coor_dec <- c(0, coor[1:max(0, length(coor) - 1)])
  return(coor - coor_dec)
}
```

On effectue ensuite l'algorithme du rejet pour simuler n variable aléatoire selon f.

Cette fonction renvoie : 1. La matrice X des simulations selon f. X est de taille $n \times 2$

2. Le vecteur Y composé des valeurs du ratio d'acceptation
3. n_l
4. Le vecteur $T = (t_1, ..., t_n)$
5. n_t

```r
# ---------------- Algorithme du rejet --------------------
rgen_f <- function(n) {
  ans <- c() # simulation de psi
  rho_sim <- c() # proba ds'acceptation
  m <- n
  # nl compteur du nombre de fois où on passe dans la boucle
  # nt compteur du nombre de g simulées
  nl <- 1
  nt <- 0
  temp <- 1
  temps_sim <- c() # liste avec les compteurs du nombre de g simulés pour chaque f que l'on simule

  while (m > 0) { # à ce moment il nous reste m variable a simuler suivant f
    u <- runif(floor(m/temp))
    g <- rgen_g(floor(m/temp))
    x <- g[, 1]
    y <- g[, 2]
    vec <- rho_top(x, y) # On le stock dans une variable pour que ça soit plus rapide
    coordonnees <- which(u <= vec)
    temps_sim <- append(temps_sim, calcul_temps(coordonnees))
    rho_sim <- append(rho_sim, rho_top(x, y))
    ans <- rbind(ans, g[coordonnees, ])
    nt <- nt + floor(m/temp)
    m <- n - length(ans[, 1]) # nombre de ligne
    nl <- nl + 1
    #temp <- rho_sim[length(rho_sim)]
    temp <- mean(rho_sim)
  }

  return(list(X = ans[1:n, ], Y = rho_sim, nl = nl, t = temps_sim[1:n], nt = nt))
}
```

Commentaires 1 : La variable temp permet d'estimer le nombre de variables suivant g à simuler. On l'estime à la fin de chaque boucles comme étant égale à $m \times \frac{1}{\text{proba d'accepation}}$

Commentaires 2 : Le calcul des Ti n'est pas indispensable et la donnée de n_t est suffisante, mais, faute de temps, nous n'avons pas eu le temps de l'enlever et de garder de bons résultats.

Question 3

```r
n <- 10000
res <- rgen_f(n)
```

Question 4.a

Méthode 1 - Estimation de a

Les support de ψ et g sont les mêmes ont peux donc effectuer les calculs suivants.

On sait que que

$$\begin{aligned}\int_{\mathbb{R}}\int_{\mathbb{R}} \Psi_(x, v)\mathrm{d}u\mathrm{d}v = \frac{1}{a} &\iff \mathbb{E}_g[\frac{\psi(X,Y)}{g(X,Y)}] = \frac{1}{a} \\ &\iff \mathbb{E}_g[m\psi(X,Y)] = \frac{1}{a} \\ &\iff a = \frac{1}{m\mathbb{E}_g[\rho(X,Y)]} \quad (*)\end{aligned}$$

On pose alors l'estimateur de a suivant :

$$\hat{b}_n = \frac{1}{m \times \frac{1}{n}\sum_{k=1}^{n}(\rho(X_k, Y_k))}$$

Biais : On remarque en appliquant l'inégalité de Jensen associée à la fonction concave $x \mapsto \frac{1}{x}$ sur $\mathbb{R}^+$ que : $\mathbb{E}_g[\frac{1}{\sum_{k=1}^{n}\rho(X_k,Y_k)}] \geq \frac{1}{\mathbb{E}_g[\sum_{k=1}^{n}\rho(X_k,Y_k)]}$

On en déduit

$$\begin{aligned}\mathbb{E}_g[\hat{b}_n] &= \mathbb{E}_g[\frac{n}{m\sum_{k=1}^{n}\rho(X_k,Y_k)}] \\ &\geq \frac{n}{m\mathbb{E}_g[\sum_{k=1}^{n}\rho(X_k,Y_k)]} \\ &= \frac{n}{n}\frac{1}{m\mathbb{E}_g[\sum_{k=1}^{n}\rho(X_1,Y_1)]} \quad \text{car les v.a } (X_k,Y_k)_k \text{sont identiquement distribuées} \\ &= \frac{1}{m\mathbb{E}_g[\sum_{k=1}^{n}\rho(X_1,Y_1)]} \\ &= a \quad \text{par } (*)\end{aligned}$$

On en déduit que $\hat{b}_n$ est un estimateur biaisé de a.

Convergance :

La variables aléatoire $(\frac{\psi(X_n,Y_n)}{g(X_n,Y_n)})_{n\geq 1}$ est une suite de variable aléatoire iid car elles sont fonctions mesurables des variables aléatoires $((X_1,Y_1),...,(X_n,Y_n))$ et d'ésperance finie car $\mathbb{E}_g[\frac{\psi(X_1,Y_1)}{g(X_1,Y_1)}] = \frac{1}{a} \leq +\infty$.

Donc d'après la Loi des Grands Nombres appliquée à la suite $(Y_i)_{i\geq 1}$ on obtient :

$$\frac{1}{n}\sum_{k=1}^{n}\frac{\psi(X_k,Y_k)}{g(X_k,Y_k)} \xrightarrow[n\to+\infty]{\mathbb{P}} \frac{1}{a}$$

De plus la fonction $x \mapsto \frac{1}{mx}$ est continue sur $\mathbb{R}^+$ donc d'après le théorème de continuité on obtient $\hat{b}_n \xrightarrow[n\to+\infty]{\mathbb{P}} a$.

Intervalle de Confiance :

On définit l'estimateur de ρ : $\bar{\rho}_n = \frac{1}{n}\sum_{k=1}^{n} \rho(X_k, Y_k)$

De même que précédamment $(\rho(X_k, Y_k))_{n\geq 1}$ est une suite de variable aléatoire iid par transformation mesurables de la suite iid $(X_k, Y_k)_{n\geq 1}$.

De plus,

$$\begin{aligned}
\mathbb{E}_g[\bar{\rho}_n] &= \mathbb{E}_g[\frac{1}{n}\sum_{k=1}^{n} \rho(X_k, Y_k)] \\
&= \mathbb{E}_g[\rho(X_1, Y_1)] \quad \text{car les v.a } (X_k, Y_k)_k \text{sont identiquement distribuées} \\
&= \int_{\mathbb{R}}\int_{\mathbb{R}} \rho_(u, v)\mathrm{d}u\mathrm{d}v \\
&= \frac{1}{am} \\
&\leq +\infty
\end{aligned}$$

De plus,

$$\begin{aligned}
\mathbb{E}_g[\bar{\rho}_n^2] &= \mathbb{E}_g[(\frac{1}{n}\sum_{k=1}^{n} \rho(X_k, Y_k))^2] \\
&\leq \mathbb{E}_g[(\frac{1}{n}\sum_{k=1}^{n} 1)^2] \qquad \text{car } \rho(x, y) \leq 1 \qquad \forall(x, y) \\
&= 1
\end{aligned}$$

Ainsi, $(\rho(X_k, Y_k))_{n\geq 1}$ est une suite de variable aléatoire iid de variance et d'esperance finie par rappoet à la mesure g. D'après le théorème Central Limite, on peut écrire :

$$\sqrt{n}(\bar{\rho}_n - \frac{1}{am}) \xrightarrow[n\to+\infty]{\mathbb{L}} \mathcal{N}(0, \sigma_p^2)$$

<u>Notation</u> : $\mathbb{V}(\rho(X, Y))) := \sigma_p^2$

On applique la Delta-Méthode pour réussir à trouver un intervalle de confiance sur $\hat{b}_n$. On définit la fonction $\phi : x \mapsto \frac{1}{mx}$ de $\mathbb{R} \mapsto \mathbb{R}$

Cette fonction est dérivable sur $\mathbb{R}^+$. De plus $\phi'(\frac{1}{am}) = -am^2 \neq 0$. Donc

$$\sqrt{n}(\phi(\hat{p}_n) - \phi(\frac{1}{am})) \xrightarrow[n\to+\infty]{\mathbb{L}} \mathcal{N}(0, \sigma_p^2\phi'(\frac{1}{am})^2) \iff \sqrt{n}(\hat{b}_n - a) \xrightarrow[n\to+\infty]{\mathbb{L}} \mathcal{N}(0, \sigma_p^2 a^4 m^2)$$

Cependant on ne peut pas calculer de façon numérique σ_p^2. On choisit un estimateur de cette valeurs. $\hat{\sigma_p}^2 := \frac{1}{n-1}\sum_{k=1}^{n}(\rho(X_k, Y_k) - \bar{\rho}_n)$

On sait que

$$\hat{\sigma_p}^2 \xrightarrow[n\to+\infty]{\mathbb{P}} \sigma_p^2 \text{ et } \hat{b}_n \xrightarrow[n\to+\infty]{\mathbb{P}} a$$

La fonction $x, y :\mapsto (\frac{1}{x} * \frac{1}{y^2})$ est continue de $\mathbb{R}_+^{2*}$ dans $\mathbb{R}$ donc d'après le théoème de continuité

$$\frac{1}{\sqrt{\hat{\sigma_p}^2}} \frac{1}{\hat{b}_n^2} \xrightarrow[n \to +\infty]{\mathbb{P}} \frac{1}{\sqrt{\sigma_p^2}} \frac{1}{a^2}$$

On a donc :

$$\frac{1}{\sqrt{\hat{\sigma_p}^2}} \frac{1}{\hat{b}_n^2} \xrightarrow[n \to +\infty]{\mathbb{P}} \frac{1}{\sqrt{\sigma_p^2}} \frac{1}{a^2}$$

$$\sqrt{n}(\hat{b}_n - a) \xrightarrow[n \to +\infty]{\mathbb{L}} \mathcal{N}(0, \sigma_p^2 a^4 m^2)$$

D'après le théorème de Slutsky :

$$\frac{\sqrt{n}(\hat{b}_n - a)}{\sqrt{\hat{\sigma_p}^2 \hat{b}_n^2 m)}} \xrightarrow[n \to +\infty]{\mathbb{L}} \mathcal{N}(0, 1)$$

On en déduit l'intervalle de confiance de a suivant :
$IC(1-\alpha) = [\hat{b}_n - m\hat{b}_n q_{1-\frac{\alpha}{2}}^{\mathcal{N}(0,1)} \sqrt{\frac{\hat{\sigma_p}^2}{n}}, \hat{b}_n + m\hat{b}_n q_{1-\frac{\alpha}{2}}^{\mathcal{N}(0,1)} \sqrt{\frac{\hat{\sigma_p}^2}{n}}]$

Question 4.b

```
b_hat_estim <- function(x, m, level = .95) {
  b_hat <- 1 / mean(x * m)
  sigma <- var(x)
  tol <- qnorm(.5 * (level + 1)) * m * b_hat * sqrt(sigma / length(x))
  ic_inf_b <- b_hat - tol
  ic_sup_b <- b_hat + tol
  return(data.frame(value = b_hat, sigma = sigma, ic_inf = ic_inf_b, ic_sup = ic_sup_b))
}
(b_hat <- b_hat_estim(res$Y, m))
```

```
##       value     sigma     ic_inf     ic_sup
## 1 0.0670863 0.0488743 0.05857507 0.07559754
```

Question 4.c

L'estimateur $\hat{b}_n$ est biaisé. On cherche donc à estimer son biais que l'on note

$$B(\hat{b}_n, a) = \mathbb{E}_g[\hat{b}_n] - a$$

On en déduit l'estimateur suivant :

$$\hat{B}(\hat{b}_n, a) = \frac{1}{K} \sum_{k=1}^{K} \hat{b}_n^{(k)} - a$$

On considère $\hat{b}_n^{(k)}$ comme une fonction mesurable des observations $((x_1, y_1), ...(x_K, y_K))$. Ces observations sont indépendantes et distribuées selon la loi g.

```r
# Tirage aléatoire
b_simu <- function(n, x) {
  return(sample(x, n, replace = TRUE))
}

# Estimation du biais via Bootstrap
biais_bootstrap <- function(K, n, x, m) {
  ans <- numeric(K)
  for (i in 1:K) {
    ans[i] <- 1 / (m * mean(b_simu(n, x)))
  }
  return(mean(ans) - (1 / (m * mean(x))))
}

# Estimation du biais
K <- 300
(x_boot <- biais_bootstrap(K, n, res$Y, m))
```

```
## [1] 3.48696e-05
```

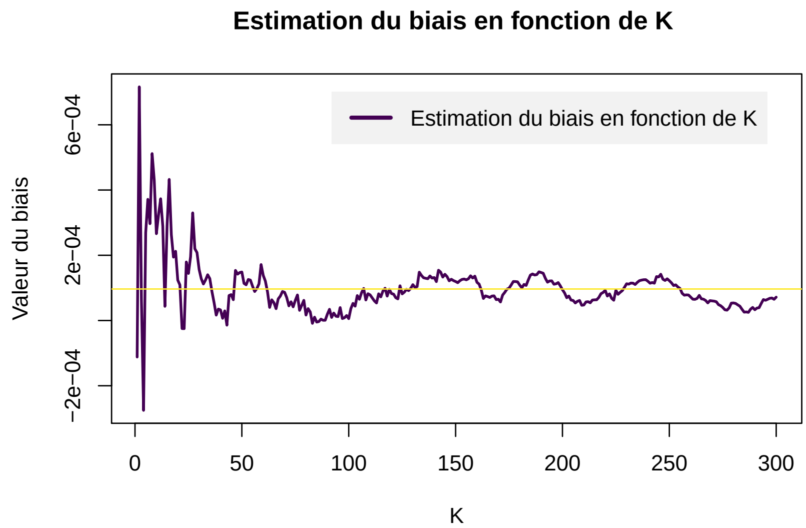
On trace l'évolution de l'estimation du biais en fonction de K.

```r
# Evolution du biais en fonction de K
biais_bootstrap_evol <- function(K, n, x, m) {
  ans <- numeric(K)
  for (i in 1:K) {
    ans[i] <- 1 / (m * mean(b_simu(n, x)))
  }
  return((cumsum(ans) / (1:K)) - (1 / (m * mean(x))))
}


bias_estim_evol <- biais_bootstrap_evol(K, 1000, res$Y, m)
bias_mean <- mean(bias_estim_evol)

# Graphique
library(viridisLite)
palette <- viridis(2)

plot(1:K, bias_estim_evol,
  type = "l", lwd = 2, col = palette[1],
  main = "Estimation du biais en fonction de K ",
  xlab = "K", ylab = "Valeur du biais"
)
abline(h = bias_mean, col = palette[2])
legend("topright", c("Estimation du biais en fonction de K"),
  col = palette[1], lwd = 3, box.lty = 0,
  bg = "gray95", inset = .05
)
```



D'après le graphique, on remarque que la valeure du biais estimé se stabilise autour de K=50 pour une valeur moyenne du biais de 1.82e-05.

Question 5.a

Définissons la variable aléatoire T

$$T := \inf\{n \geq 1, U_n \leq \frac{f(Y_n)}{Mg(Y_n)}\}$$

D'après le cours, on sait que la variable aléatoire T suit une loi géometrique de paramètre $\frac{1}{M}$.

Donc

$$\mathbb{E}_f[T] = M \iff a = \frac{\mathbb{E}_f[T]}{m}$$

On choisis comme estimateur de a : $\hat{a}_n = \frac{1}{m}\frac{1}{n}\sum_{i=1}^{n} T_i$ où les T_i représente le temps de simulation de la ième variable suivant f.

Biais : $\mathbb{E}_f[\hat{a}_n] = \frac{\mathbb{E}_f[T]}{m} = a$ L'estimateur $\hat{a}_n$ est sans biais.

Convergance: La suite $(T_i)_{i\geq 1}$ est une suite de variable aléatoire iid de loi géometrique $\frac{1}{M}$. Donc $\forall i \in \mathbb{N}, T_i \rightsquigarrow \mathcal{G}(\frac{1}{M})$. Donc Les T_i sont d'esperance finie.

Donc d'après la Loi des Grands Nombres appliquée à la suite $(T_i)_{i\geq 1}$ on obtient :

$$\frac{1}{n}\sum_{k=1}^{n} T_k \xrightarrow[n\to+\infty]{\mathbb{P}} \mathbb{E}_f[T_1]$$

$$\iff \frac{1}{n}\sum_{k=1}^{n} T_k \xrightarrow[n\to+\infty]{\mathbb{P}} a$$

Intervalle de confiance: Les $(T_i)_{i\geq 1}$ sont iid et suivent une lois géométrique. Ils sont donc de variance finie. D'après le théorème central limite,

$$\sqrt{n}(\hat{a}_n - a) \xrightarrow[n\to+\infty]{\mathbb{L}} \mathcal{N}(0, \sigma_a^2)$$

Où $\sigma_a^2 := \text{Var}(\frac{T}{m})$

Exactement de la même façon que dans la question 4.a on définit l'estimateur σ_a^2 de la manière suivante :

$\hat{\sigma_a}^2 := \frac{1}{m(n-1)} \sum_{k=1}^{n} (T_k - \bar{T}_n)^2$

En appliquant le théorème de continuité à la fonction $x \mapsto 1/x$ sur $\mathbb{R}^+$ et le théorème de Slutsy de la même manière que dans la question 4.a, on en déduit :

$$\frac{\sqrt{n}(\hat{a}_n - a)}{\hat{\sigma_a}^2} \xrightarrow[n \to +\infty]{\mathbb{L}} \mathcal{N}(0,1)$$

On en déduit l'intervalle de confiance de a suivant : $IC(1-\alpha) = [\hat{a}_n - q_{1-\frac{\alpha}{2}}^{\mathcal{N}(0,1)} \sqrt{\frac{\hat{\sigma}_a^2}{n}}, \hat{a}_n + q_{1-\frac{\alpha}{2}}^{\mathcal{N}(0,1)} \sqrt{\frac{\hat{\sigma}_a^2}{n}}]$

Question 5.b

```r
a_hat_estim <- function(x, m, level = .95) {
  a_hat <- mean(x / m)
  sigma <- var(x / m)
  tol_a <- qnorm(.5 * (level + 1)) * sqrt(var(x / m) / length(x))
  ic_inf_a <- a_hat - tol_a
  ic_sup_a <- a_hat + tol_a
  return(data.frame(value = a_hat, sigma = sigma, ic_inf = ic_inf_a, ic_sup = ic_sup_a))
}
(a_hat <- a_hat_estim(res$t, m))
```

```
##        value       sigma     ic_inf     ic_sup
## 1 0.06641872 0.003304253 0.06529208 0.06754536
```

Question 6.

Choix de n:
Soit $\epsilon > 0$ le seuil de tolérance.

- Pour $\hat{b}_n$: On pose $\epsilon = m\ \hat{b}_n\ q_{1-\frac{\alpha}{2}}^{\mathcal{N}(0,1)} \sqrt{\frac{\hat{\sigma_p}^2}{n}} \iff n = (m\ \hat{b}_n\ q_{1-\frac{\alpha}{2}}^{\mathcal{N}(0,1)})^2\ \frac{\hat{\sigma_p}^2}{\epsilon^2}$

De plus $\hat{b}_n$ est issu de simulations de g. Le coût de simulation suivant g est n_t.

- Pour $\hat{a}_n$: On pose $\epsilon = q_{1-\frac{\alpha}{2}} \sqrt{\frac{\hat{\sigma}_a^2}{n}} \iff n = q_{1-\frac{\alpha}{2}} \frac{\hat{\sigma}_a^2}{\epsilon^2}$

De plus $\hat{a}_n$ est issu de simulations de f. Le coût de simulation suivant f est n.

Rapport des coûts:
On obtient :

$$R = \frac{n \times C_g}{n \times C_f} = m^2\ \hat{b}_n^2\ \frac{\hat{\sigma_p}^2 C_g}{\hat{\sigma}_a^2 C_f}$$

```r
# Rapport des coûts
(R <- (m^2 * b_hat$value^2 * var(res$Y) * n) / (a_hat$sigma * res$nt))
```

```
## [1] 57.07104
```

L'estimateur $\hat{b}_n$ est R_bis fois moins rapide que $\hat{a}_n$.

Question 7.a

On note

$$f_X(x) = \int_{\mathbb{R}} f(x,y)\mathrm{d}y = a\int_{[-1,1]} \left[\left|\sin\left(\frac{2}{\pi}x^2 - \frac{\pi}{4}\right)\right| + 4\cos(x)^2 + y^4\right] e^{-2(x+|y|)} 1_{\{x\ \in\ [-\frac{\pi}{2},\frac{\pi}{2}]\}}\mathrm{d}y$$

Après des intégrations par parties successives, on trouve le résultat suivant :

$$f_X(x) = a1_{\{x\ \in\ [-\frac{\pi}{2},\frac{\pi}{2}]\}} e^{-2x}\left((1-e^{-2})\left[|\sin\left(\frac{2}{\pi}x^2 - \frac{\pi}{4}\right)| + 4\cos(x)^2\right] + \left(\frac{3}{2} - \frac{21}{2}e^{-2}\right)\right)$$

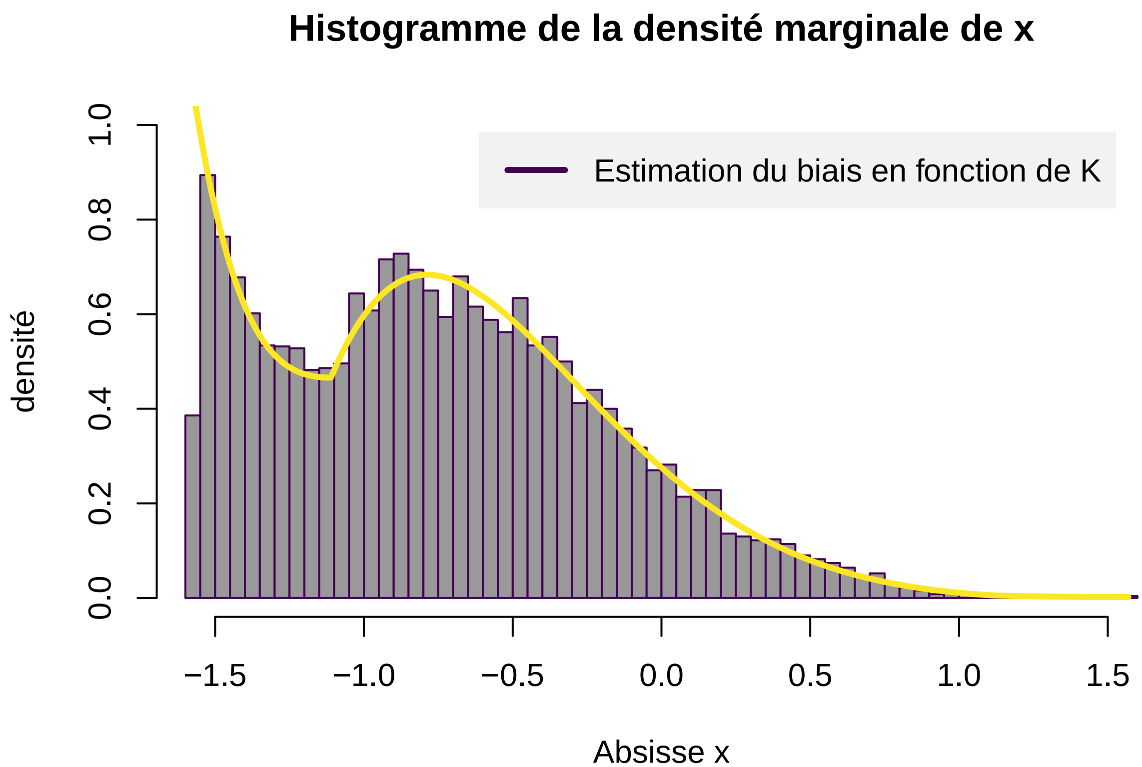
Question 7.b

```r
# Estimateur de Fx
fX_hat <- function(x, a_hat) {
  # Estimation
  value <- a_hat$value * (abs(x) <= pi / 2) * exp(-2 * x) *
    ((1 - exp(-2)) * (abs(sin((2 / pi) * x * x - (pi / 4))) + 4 * cos(x) *
      cos(x)) + (3 / 2) - (21 / 2) * exp(-2))
  # Variance
  s2 <- a_hat$sigma * ((abs(x) <= pi / 2) * exp(-2 * x) * ((1 - exp(-2))
  * (abs(sin((2 / pi) * x * x - (pi / 4))) + 4 * cos(x) *
       cos(x)) + (3 / 2) - (21 / 2) * exp(-2)))^2

  return(data.frame(value = value, sigma = s2))
}
```

```r
hist(res$X[, 1],
  freq = FALSE, main = "Histogramme de la densité marginale de x",
  ylab = "densité", xlim = c(-pi / 2, pi / 2), ylim = c(0, 1),
  col = "grey60", border = palette[1], breaks = 100, xlab = "Absisse x"
)

t <- seq(-pi / 2, pi / 2, 0.01)
lines(t, fX_hat(t, a_hat)$value, col = palette[2], lwd = 3)
legend("topright", c("Estimation du biais en fonction de K"),
  col = palette[1], lwd = 3, box.lty = 0,
  bg = "gray95", inset = .05
)
```



L'histogramme et la densité estimé concordent bien. L'estimation semble juste.

Question 8

Méthode 2 - Estimateur ponctuel

On pose $Z_k = \frac{\psi(x,Y_k)w(X_k)}{\psi(X_k,Y_k)}$ suivant la densité f. $(Z_k)_k$ est une suite de variables iid par transformation mesurable de (X_k, Y_k) qui sont iid.

De plus,

$$\begin{aligned}
\mathbb{E}_f[Z_1] &= \mathbb{E}_f[\frac{\psi(x,Y_1)w(X_1)}{\psi(X_1,Y_1)}] \\
&= \int_{\mathbb{R}}\int_{\mathbb{R}} \frac{\psi(x,v)w(u)}{\psi(u,v)} f(u,v)\mathrm{d}u\mathrm{d}v \\
&= \int_{\mathbb{R}}\int_{\mathbb{R}} a\psi(x,v)w(u)\mathrm{d}u\mathrm{d}v \\
&= \int_{\mathbb{R}} a\psi(x,v)(\int_{\mathbb{R}} w(u)\mathrm{d}u)\mathrm{d}v \quad \text{(Par Fubini intégrable)} \\
&= \int_{\mathbb{R}} a\psi(x,v)\mathrm{d}v) \\
&= f_X(x)
\end{aligned}$$

Ainsi par la loi des grands nombres :

$$\widehat{w}_n(x) \xrightarrow[n\to+\infty]{\mathbb{P}} f_X(x)$$

Les fonctions psi et w sont des densité. Les moments d'ordre 1 et 2 de Z_k sont nécéssairement finis. Donc $\text{Var}(Z_1) \leq +\infty$

D'après le Théorème Central-Limite :

$$\sqrt{n}(\bar{w}_n(x) - f_X(x)) \xrightarrow[n\to+\infty]{\mathbb{L}} \mathcal{N}(0, \sigma_z^2)$$

Où $\sigma_z^2 := \text{Var}(Z_1)$

On définit l'estimateur σ_z^2 de la manière suivante :

$\hat{\sigma_a}^2 := \frac{1}{(n-1)} \sum_{k=1}^{n} (Z_k - \bar{Z}_n)^2$

En appliquant le théorème de continuité à la fonction $x \mapsto 1/x$ sur $\mathbb{R}^+$ et le théorème de Slutsy de la même manière précédemment, on en déduit :

$$\frac{\sqrt{n}(\hat{w}_n(x) - f_X(x))}{\hat{\sigma_z}^2} \xrightarrow[n\to+\infty]{\mathbb{L}} \mathcal{N}(0, 1)$$

On en déduit l'intervalle de confiance de a suivant :

$$IC(1-\alpha) = [\hat{w}_n(x) - q_{1-\frac{\alpha}{2}}^{\mathcal{N}(0,1)} \sqrt{\frac{\hat{\sigma}_w^2}{n}}, \hat{w}_n(x) + q_{1-\frac{\alpha}{2}}^{\mathcal{N}(0,1)} \sqrt{\frac{\hat{\sigma}_w^2}{n}}]$$

Question 9

On cherche ω telle que la variance de l'estimateur est à son minimum.

$$\underset{\omega}{\text{argmin}} \mathbb{V}ar \left[\frac{\psi(x, Y)\omega(X)}{\psi(X, Y)} \right] = \underset{\omega}{\text{argmin}} \mathbb{V}ar \left[\frac{\frac{1}{a} f(x, Y)\omega(X)}{\frac{1}{a} f(X, Y)} \right]$$

Ici on va utiliser le fait que X et Y sont indépendantes donc, en particulier, on a $f_{(X,Y)}(X, Y) = f_X(X) \times f_Y(Y)$ et on peut réécrire notre problème :

$$\underset{\omega}{\text{argmin}} \mathbb{V}ar \left[\frac{\psi(x, Y)\omega(X)}{\psi(X, Y)} \right] = \underset{\omega}{\text{argmin}} \mathbb{V}ar \left[\frac{\omega(X)}{f_X(X)} \right]$$

La variance est positive. La variance d'une constante est nulle. Il suffit de prendre une densité ω proportionnelle à f_X pour obtenir une variance nulle.

Dans la suite, on essaiera donc de prendre une densité ω d'une loi usuelle qui pourrait approcher à un facteur près la densité marginale de X.

Question 10

On écrit l'estimateur de $\hat{w}_n(x)$ en utilisant les résultats de rgen_f :

```
# Estimateur
wn_estim <- function(x, w, level = 0.95) {
  Z_x <- psi(x, res$X[, 2]) * w(res$X[, 1]) / psi(res$X[, 1], res$X[, 2])
  wn_hat <- mean(Z_x)
  sigma <- var(Z_x)
  tol_wn <- qnorm(.5 * (level + 1)) * sqrt(sigma / length(Z_x))
  # Intervalle de confiance
  ic_inf_wn <- wn_hat - tol_wn
```

```r
  ic_sup_wn <- wn_hat + tol_wn
  return(data.frame(value = wn_hat, sigma = sigma, ic_inf = ic_inf_wn, ic_sup = ic_sup_wn))
}
```

On fait un premier test en utilisant la densité w d'une uniforme.

```r
# Estimateur avec la loi uniforme
w_unif <- function(x) {
  return((1 / pi) * (abs(x) <= pi / 2))
}

wn_unif <- wn_estim(x = -1, w_unif)
```

D'après la question 9, la densité optimal w_* telle que $\widehat{w}_n(w_*, x)$ est l'estimateur de variance minimale est obtenu pour $w_* = f_X$, mais f_X est inconnu. Or d'après la question 7.a et l'expression de f_X en e^{-2x} cela nous fait penser à la densité d'une loi de Weibull tronquée sur $[-pi/2, pi/2] $ de paramètres $k = 1$ et $ = 1/2 $.

```r
#------------------ Loi Weibull Tronquée ----------------------

#  Fonction répartition
F_rep_weibull <- function(x, k = 1, lambda = 1/2) {
  return(1 - exp(-x/lambda)^k)
}



# Densité
dweibull_tronq <- function(x, k = 1, lambda = 1/2) {
  return((k/lambda) * (x/lambda)^(k-1) * exp(-(x/lambda)^k) * (abs(x) <= pi/2) /
           (F_rep_weibull(pi/2) - F_rep_weibull(-pi/2)))
}
```

```r
# ------------- Test avec la Weibull ------------
w_opti <- function(x) {
  return(dweibull_tronq(x))
}

(wn_hat_weibull <- wn_estim(-1, w = w_opti))
```

```
##       value     sigma    ic_inf    ic_sup
## 1 0.6074777 0.2109109 0.5984766 0.6164788
```

```r
# Gain entre Uniforme et Weibull
(Gain <- wn_unif$sigma/wn_hat_weibull$sigma)
```

```
## [1] 36.1083
```

En passant d'une loi uniforme à une loi de weibull tronquée, le gain de coùt de variance est de 30 environ.

Question 11

Les deux estimateurs $\widehat{f}_{X,n}(-1)$ et $\widehat{w}_n(-1)$ sont des estimateurs sans biais. De plus, ils ont tout les deux été simulés suivant la densité f donc le rapport des couts est égale à 1. Ainsi, le rapport de coût pour lesquels les deux estimateurs atteignent la même erreur quadratique moyenne est :

$$R(\hat{f}_{X,n}(-1), \hat{w}_n(-1)) = \frac{\mathrm{Var}[\hat{f}_{X,n}(-1)]}{\mathrm{Var}[\hat{w}_n(-1)]}$$

```
# Estimateur de fX
fX_hat_est <-fX_hat(-1, a_hat)

# Rapport des coûts
(R_bis  <- fX_hat_est$sigma/wn_hat_weibull$sigma)
```

```
## [1] 1.266506
```

L'estimateur $\hat{f}_{X,n}(-1)$ est R_bis (R_bis ≥ 1) fois plus rapide que $\hat{w}_n(-1)$.

Exerice 2

```
library(Rfast)
```

```
## Loading required package: Rcpp
```

```
## Loading required package: RcppZiggurat
```

Question 1

La matrice de variance-covariance n'est pas diagonale donc on utilise la décomposition de Cholesky.

```
# Décomposition de Cholesky
mv_norm <- function(n, mu, sigma) {
  d <- length(mu)
  z <- matrix(rnorm(d * n), nrow = d) # vecteur normal de même taille que la mu
  L <- t(chol(sigma))
  return(mu + L %*% z) # L %*% z produit matriciel
}

# Application
mu <- c(0.1, 0, 0.1)
n <- 10000
sigma <- matrix(c(0.047, 0, 0.0117, 0, 0.047, 0, 0.0117, 0, 0.047), nrow = 3)
x <- mv_norm(n, mu, sigma)
```

Question 2.a

La suite $X_i = (X_{1,i}, X_{2,i}, X_{3,i})_{i=1,\dots,n}$ suit la loi $\mathbf{X}$.
On écrit alors :

$$\delta = \mathbb{E}[h(X_i)] = \mathbb{E}[\min(3, \frac{1}{3}\sum_{k=1}^{3} e^{-X_{k,i}})]$$

Et on definit la fonction telle que $h : x = (x_1, x_2, x_3) \mapsto \min(3, \frac{1}{3}(e^{-x_1} + e^{-x_2} + e^{-x_3}))$
On déduit l'estimateur de Monte-Carlo pour δ :

$$\bar{\delta}_n = \bar{h}_n = \frac{1}{n}\sum_{i=1}^{n} \min(3, \frac{1}{3}\sum_{k=1}^{3} e^{-X_{k,i}})$$

Question 2.b

```
# Estimateur de Monte-Carlo
mc_estim <- function(y) {
  # Moyenne
  delta <- mean(y)
  # Variance
  s2 <- var(y)

  return(data.frame(
    n = n, value = delta, sigma2 = s2, mse = s2 / length(y)
```

```
  ))
}
```

```
# Fonction h
h <- function(x) {
  res <- colmeans(exp(-x))
  return(res * (res <= 3))
}

(delta_MC_hat <- mc_estim(h(x)))
```

```
##       n     value     sigma2          mse
## 1 10000 0.9581328 0.01670563 1.670563e-06
```

Question 3.a

D'après le cours si $\mathbf{X} \rightsquigarrow \mathcal{N}(\mu, \Sigma)$, alors $2\mu - \mathbf{X} \rightsquigarrow \mathcal{N}(\mu, \Sigma)$.

On définit alors la transformation mesure $A : \mathbb{R}^3 \to \mathbb{R}^3$ qui à $x \mapsto 2\mu - x$.

La variable aléatoire A($\mathbf{X}$) est la variable antithétique de $\mathbf{X}$ et l'estimateur de la variable antithétique est défini par :

$$\hat{\delta}_n = \frac{1}{n} \sum_{i=1}^{n} \frac{h(\mathbf{X}_i) + h \circ A(\mathbf{X}_i)}{2}$$

De plus, A est une transformation décroissante de $\mathbb{R}^3$ (en chacune de ses coordonnées) et h est une fonction monotone. Donc $\text{Cov}(h(\mathbf{X}_k) + h \circ A(\mathbf{X}_k)) < 0$ et on en déduit :

On calcul à présent le facteur de réduction de variance théorique R_1.

$\bar{\delta}_n$ et $\hat{\delta}_n$ étant deux estimateurs sans biais (cf cours) on a donc :

$$\text{Var}(\bar{\delta}_n) = \frac{1}{n}\text{Var}(h(X_i))$$

$$\text{Var}(\hat{\delta}_n) = \frac{1}{n}\text{Var}(\frac{h(\mathbf{X}_i) + h \circ A(\mathbf{X}_i)}{2})$$

$\hat{\delta}_n$ et $\bar{\delta}_n$ atteignent une erreur quadratique moyenne $\epsilon^2 > 0$ pour

$$n = \frac{\text{Var}(h(X_i))}{\epsilon^2}$$

$$n = \frac{\text{Var}(\frac{h(\mathbf{X}_i)+h\circ A(\mathbf{X}_i)}{2})}{\epsilon^2}$$

Ainsi le facteur de réduction de variance théorique est

$$R_1 = \frac{\text{Var}(h(X_i))}{\text{Var}(\frac{h(\mathbf{X}_i)+h\circ A(\mathbf{X}_i)}{2})}$$

Question 3.b

On calcule d'abord l'estimateur de la variable antithétique

```r
# Fonction A
A <- function(x, mu) {
  return(2 * mu - x)
}
# Estimateur de la variable antithétique
delta_ant_hat <- mc_estim(0.5 * (h(x) + h(A(x, mu))))

# Erreur quadratique moyenne associée
delta_ant_hat$sigma2
```

```
## [1] 0.0003443911
```

Ensuite on calcule le rapport de coût

```r
# Facteur de réduction de variance théorique
(R1 <- delta_MC_hat$sigma2 / delta_ant_hat$sigma2)
```

```
## [1] 48.50772
```

Conclusion : $\bar{\delta}_n$ prend R1 fois plus de temps que $\hat{\delta}_n$ pour atteindre n'importe quelle erreure quadratique moyenne.

Question 4.a

En pratique, on remarque que

$$\forall i \in \{1, ..., n\} \qquad \min(3, \frac{1}{3}\sum_{k=1}^{3} e^{-X_{k,i}}) = \frac{1}{3}\sum_{k=1}^{3} e^{-X_{k,i}}$$

On choisit donc la fonction $h_o : \mathbb{R} \to \mathbb{R}$ telle que

$$h_o : x = (x_1, x_2, x_3) \mapsto \frac{1}{3}(e^{-x_1} + e^{-x_2} + e^{-x_3}))$$

De plus, pour $\mathbf{X} \rightsquigarrow \mathcal{N}(\mu, \Sigma)$,

$$\begin{aligned}
\mathbb{E}[h_o(X)] &= \frac{1}{3}(\mathbb{E}[e^{-X_1}] + \mathbb{E}[e^{-X_2}] + \mathbb{E}[e^{-X_3}]) \\
&= \frac{1}{3}(M_{X_1}(-1) + M_{X_2}(-1) + M_{X_3}(-1)) \quad \text{(où } M_{X_1} \text{ représente la fonction generatrice d'une loi normale réelle)} \\
&:= m
\end{aligned}$$

La variable $h_o(X)$ corresponds à la fonction génératrice des moments d'une loi normale, elle est donc bornée dans $\mathcal{L}^2$. Donc $\text{Var}[h_o(X)] \leq \infty$

```r
# Fonction ho
ho <- function(x) {
  return(colmeans(exp(-x)))
}

# Calcul de E[ho(x)]
# Fonction génératrice des moments d'une loi normale de paramètres mu et sigma
```

```r
gen_norm <- function(t, mu, sigma) {
  return(exp(mu * t + (sigma^2 * t^2) / 2))
}

m <- (1 / 3) * (gen_norm(-1, 0.1, sqrt(.047)) + gen_norm(-1, 0, sqrt(.047)) + gen_norm(-1, 0.1, sqrt(.04
```

Donc pour $b \in \mathbb{R}$, l'estimateur de la variable de contrôle associée à la fonction h_o est

$$\hat{\delta}_n(b) = \frac{1}{n}\sum_{i=1}^{n}(h(X_i) - b\{h_o(X_i) - m\}), \quad \text{où} \quad (X_i)_{i\geq 1} \text{ iid} \rightsquigarrow \mathcal{N}_3(\mu, \Sigma)$$

```r
# Estimateur de contrôle
mc_estim_controle <- function(x, h, ho, b, m) {
  # Moyenne
  delta <- mean(h(x) - b * (ho(x) - m))
  # Variance
  s2 <- var(h(x) - b * (ho(x) - m))

  return(data.frame(
    n = n, value = delta, sigma2 = s2, mse = s2 / length(h(x) - b * (ho(x) - m))
  ))
}

# Condition
(b <- cov(ho(x), h(x)) / var(ho(x)) <=0)
```

```
## [1] FALSE
```

La condition est numériquement validée.

Question 4.b

On sait que l'estimateur de variance minimale $\hat{\delta}_n(b^*)$ est obtenu par

$$b^* = \arg\min_{b\in\mathbb{R}} \text{Var}[\hat{\delta}_n(b)] = \frac{\text{Cov}[h(X), h_o(X)]}{\text{Var}(h_o(X))}$$

En pratique on ne connait pas b^*. On utilise donc la méthode "burn-in" qui consiste à estimer b selon les l (l petit) premiers termes.

On définit l'estimateur de b^* suivant :

$$\hat{b}_l^* = \frac{\sum_{i=1}^{l}(h_o(X_i) - m)(h(X_i) - \bar{h}_l)}{\sum_{i=1}^{l}(h_o(X_i) - m)^2}$$

```r
# Estimateur de b
b_hat <- function(x,h,ho,l,m){
  # On garde les l premières valeurs de x
  x_l <- x[,][,1:l]

  # Estimateur tronqué de delta
```

```
  h_l <- mean(h(x_l))

  # Estimateur de b
  value <- mean((ho(x_l) - m)*(h(x_l) - h_l))/mean((ho(x_l) - m)^2)

  return(data.frame(n = length(x[1,]), l=l, value = value, m = m, h_l = h_l))

}
```

Pour choisir l (que l'on note à présent l^*) de façon "judicieuse", on trace l'évolution de $\hat{b}_l$ en fonction de l. On sait que $\hat{b}_l*$ L'idée est de choisir l^* petit et tel que $\hat{b}_l*$ soit proche de 1.

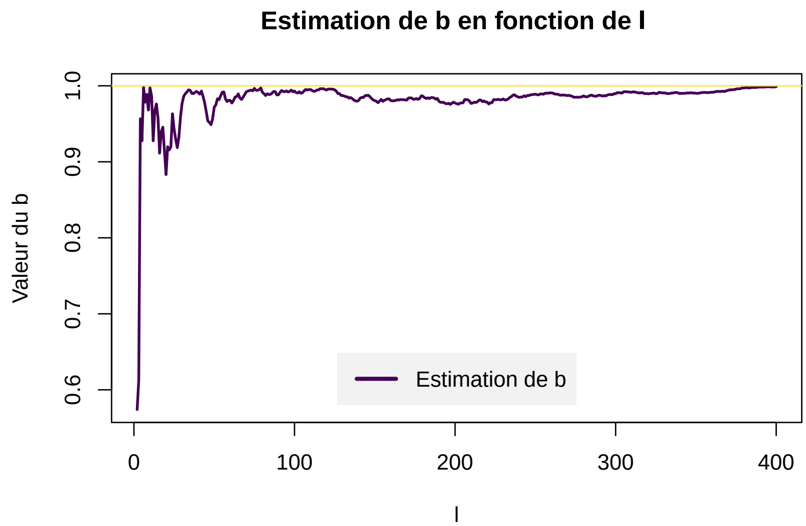
```
# Evolution de b
b_evol <- function(x,h,ho,l,m){
  ans <- numeric(l - 1)
  for (i in 2:l){
    ans[i - 1] <-  b_hat(x,h,ho,i,m)$value
  }
  return(ans)
}
```

Remarque : On choisit l = 400 pour mieux analyser le graphique

```
# Vecteur de b estimée pour i allant de 2 à l
l <- 400
b_estim_evol <- b_evol(x,h,ho,l,m)

# Graphique
palette <- viridis(2)

plot(2:l, b_estim_evol,
  type = "l", lwd = 2, col = palette[1],
  main = "Estimation de b en fonction de l ",
  xlab = "l", ylab = "Valeur du b"
)
abline(h = 1, col = palette[2])
legend("bottom", c("Estimation de b"),
       col = palette[1], lwd = 3, box.lty = 0,
       bg = "gray95", inset = .05
)
```



La valeur de $\hat{b}_n$ se stabilise sur 1 à partir de d'environ l=100. On fixe $l^* := 150$, on calculer $\hat{\delta}^*_{n-l^*}$

```
# Estimateur
l_star <- 100
b_star <- b_hat(x,h,ho,l_star,m)
delta_controle_opt <- mc_estim_controle(x[,l_star:n],h,ho,b_star$value,m)

# Rapport des variance
(R2 <- delta_MC_hat$sigma2 / delta_controle_opt$sigma2)
```

```
## [1] 21470.43
```

Le rapport de variance est très élevé. L'estimateur de la variable de contrôle est plus de $10e9$ fois plus efficace que l'estimateur de Monté-Carlo. Cela s'explique probablement par la corrélation entre ho et h très proche de 1. Cette exemple valide la théorie vue en cours.

Exercice 3

Remarque : La loi geometrique sur prend ses valeurs dans $\{0, 1, 2, ...\}$ or nous utilisons par convention une loi géométrique à valeur dans $\{1, 2, ...\}$ Dans toute la suite de l'exercice, pour générer une loi geométrique de paramètre p, on utilisera la formule rgeom(p) + 1. Pour calculer $\mathbb{P}[X \leq k]$ pour $k \in \mathbb{R}$, on utilisera pgeom(k-1,p).

On note $\hat{\delta}_n^{MC}$ l'estimateur de Monte Carlo. Pour faciliter les notations, on note :

$$\hat{\delta}_n^{MC} = \sum_{k=1}^{n} h(X_k)$$

. On se permet de ne pas explicité la fonction h car elle est compliquée à écrire et sans intêret.

Question 1.

```
# Données
p <- .2
n <- 10000
m <- 2
theta <- 2
```

```
# Simulation de n variable suivant S
rgen_s <- function(n, p = .2, m = 2, theta = 2) {
  s <- numeric(n)
  for (i in 1:n) {
    y <- rgeom(1, p) + 1
    x <- rgamma(y, m, theta)
    s[i] <- sum(log(x + 1))
  }
  return(s)
}

# Estimateur de Monte-Carlo
mc_estim <- function(y) {
  delta <- mean(y)
  s2 <- var(y)
  return(data.frame(
    n = n, value = delta, sigma2 = s2, var_estim = s2 / length(y)
  ))
}

# Simulation et estimation
s <- rgen_s(n)
(mc_est <- mc_estim(s))
```

```
##       n    value   sigma2    var_estim
## 1 10000 3.156177 8.734813 0.0008734813
```

Question 2.a

On écrit

$$\mathbb{E}[S] = \sum_{k=0}^{+\infty} \mathbb{E}[S|Y = k]\mathbb{P}[Y = k]$$

On va apppliquer la méthode de stratification.On choisit Y comme variable de stratification. Y est à valeure dans $\mathbb{N}^*$ donc on doit choisir une partition de $\mathbb{N}^*$. Sous l'hypothèse de l'allocation proportionnelle, on a que le nombre de tirage pour l'évenement Y=k parmis n tirages en tout est

$n_k = n\mathbb{P}(Y = k) = np(1-p)^{k-1}$

On choisit de créer L strates $D_1, ..., D_L$ où $L \in \mathbb{N}^*$. On a alors $\mathbb{N}^* = \cup_{k=1}^{L} D_k$ où

$$D_k = \begin{cases} \{k\} & \text{si } 1 \leq k \leq L-1 \\ \{k \in \mathbb{N}, k \geq L\} & \text{si } k \geq L \end{cases}$$

On en déduit l'allocation:

$$n_k = \begin{cases} n\mathbb{P}[Y = k] & \text{si } 1 \leq k \leq L-1 \\ n\mathbb{P}[Y \geq k] & \text{si } k \geq L \end{cases}$$

n_k n'étant pas un nombre entier, on prend finalement :

$$n_k = \begin{cases} \lfloor n\mathbb{P}[Y = k] \rfloor & \text{si } 1 \leq k \leq L-1 \\ n - \sum_{k=1}^{L-1} \lfloor n_k \rfloor & \text{si } k \geq L \end{cases}$$

On en déduit l'estimateur stratifié suivant

$$\hat{\delta}_n(n_1, ..., n_L) = \frac{1}{n} \sum_{k=1}^{L} \sum_{i=1}^{n_k} S_i^{(k)}$$

où les $S_i^{(k)}$ sont distribués suivant la loi $\mathcal{L}(X|Y \in D_k)$

Question 2.b

Estimateur : On sait que sous l'hypothèse d'allocation proportionelle, la variance de l'estimateur stratifié s'écrit de la façon suivante :

$$\mathbb{Var}[\hat{\delta}_n(n_1, ..., n_L)] = \frac{1}{n} \sum_{k=1}^{L} p_k \sigma_k^2$$

où σ_k représente la variance à l'intérieur de la strate k.

De plus, pour cette fonction, on considère que s correspond aux réalisations de $S|Y \in D_k$, n_k à une allocation et $p_k = \mathbb{P}(Y \in D_k)$

```r
# --- Estimateur stratifié
mc_strat <- function(s, n_k, p_k, level = 0.95) {
  n <- length(s)
  L <- length(n_k)
  # --- Calcul de l'estimateur
  delta <- mean(s)
  # --- Calcul de la variance intra-strate
  s2_intra <- split(s, as.factor(rep(1:L, times = n_k)))
  s2_intra <- sapply(s2_intra, var)
  # --- Calcul de la variance de l'estimateur
  s2 <- sum(p_k * sqrt(s2_intra))^2
```

```
  return(list(ans = data.frame(
    value = delta, var = s2, var_estim = s2 / n), var_intra = s2_intra))
}
```

Pour obtenir des réalisations suivant $S|Y \in D_k$, nous avons deux cas de figures : lorsque $1 \leq k \leq 14$ et lorsque $k = 15$. - Pour $1 \leq k \leq 14$: il suffit de simuler $n_k \times k$ (pour que l'on retombe bien sur un vecteur de taille n à la fin) lois géométriques. - Pour $k = 15$ on doit simuler $S|Y \geq 15$. Pour cela, on utilise la formule vu précédamment pour générer des lois conditionnelles à l'aide d'une loi uniforme et de l'inverse généralisée.

```
# Simulation de S/Y=k
rgen_s_singleton <- function(vect_n_k,k,theta,m){
  x <- c()

  for (i in 1:length(vect_n_k)){
    temp <- log(rgamma(vect_n_k[i]*k[i],m ,theta) + 1)
    temp <- split(temp,rep(seq_len(vect_n_k[i]),each = k[i]))

    x <- c(x,sapply(temp,sum))
  }
  return(x)
}

# Simulation de S/Y>14
rgen_cond_last_opti <- function(vect_n_k,theta,m){
  L <- length(vect_n_k)
  # Dernière strate
  p_lim <- pgeom(13,p)
  # Y / Y > 13
  y_cond <- qgeom(p_lim + (1-p_lim)*runif(vect_n_k[L]),p)
  # Valeur prise par Y / Y > 13
  lvl_y_cond <- as.numeric(levels(as.factor(y_cond)))
  # Nombre occurence de chacune valeurs
  n_lvl <- as.numeric(table(as.factor(y_cond)))
  # Nombre de realisation de
  n_x <- lvl_y_cond * n_lvl
  x <- c()
  for (i in 1:length(lvl_y_cond)){
    temp <- log(rgamma(n_x[i],m ,theta) +1)

    temp <- split(temp,rep(seq_len(n_lvl[i]),each= lvl_y_cond[i]))
    #     g <- split(g,rep(seq_len(vect_n_k[i]),each= k[i]))
    x <- c(x,sapply(temp,sum))
  }
  return(x)
}
```

```
L <- 15
# Pois des strats et allocation
p_k <- dgeom(0:(L - 2), p)
p_k <- c(p_k, 1 - sum(p_k))
n_k <- floor(n * p_k[-L])
n_k <- c(n_k, n - sum(n_k))
```

```r
#x <- rgen_cond_0(n_k[1])
x <- rgen_s_singleton(n_k[-c(15)],1:14, theta,m)
x <- c(x, rgen_cond_last_opti(n_k,theta, m))


(mc_strat_est <- mc_strat(x, n_k, p_k))
```

```
## $ans
##      value       var    var_estim
## 1 3.178142 0.5555138 5.555138e-05
##
## $var_intra
##          1          2          3          4          5          6          7
##  0.1104471  0.2053109  0.3175568  0.4289088  0.5267821  0.6474542  0.8090108
##          8          9         10         11         12         13         14
##  0.8304038  0.8819619  1.1253738  0.9549589  1.2439767  1.1305897  1.8164056
##         15
## 10.1376504
```

Efficacité relative :

$\hat{\delta}_n^{MC}$ et $\hat{\delta}$ sont deux estimateurs sans biais. L'efficacité relative est donnée par:

$$R = \frac{C_s}{C_x} \times \frac{\mathrm{Var}[\hat{\delta}_n^{MC}]}{\mathrm{Var}[\hat{\delta}_n(n_1, ..., n_L)]}$$

On utilise la fonction microbenchmark pour calculer le rapport du cout de simulation des deux variables

```r
# Effet relatif
library(microbenchmark)

# Fonction test Strat
test_strat <- function(){
  x <- rgen_s_singleton(n_k[-c(15)],1:14, theta,m)
  x <- c(x, rgen_cond_last_opti(n_k,theta, m))
  mc_strat(x, n_k, p_k)
}

# Fonction test MC
test_mc <- function(){
  s <- rgen_s(n)
  mc_estim(s)
}

(test <- microbenchmark(test_mc(), test_strat()))
```

```
## Unit: milliseconds
##          expr      min       lq     mean   median       uq       max neval
##     test_mc() 50.72549 62.32208 63.88819 64.20130 65.48973 112.56964   100
##  test_strat() 18.69112 19.42075 21.82033 20.06172 21.36892  34.49998   100
```

On remarque que le rapport de cout est d'environ 60/18. On calcule à présent R

```r
t_mc_classic <- mean(test$time[which(test$expr == "test_mc()")])
t_mc_strat <- mean(test$time[which(test$expr == "test_strat()")])
```

```r
(R3 <- (mc_est$sigma2 / mc_strat_est$ans$var) * t_mc_classic/t_mc_strat)
```

```
## [1] 46.03816
```

Donc l'estimateur stratifié est R fois plus efficace que l'estimateur de Monte Carlo classique.