Optimize

Jérémie Barde

12 January 2023

Résumé

Le présent document contient des exemples de codes R sur les fonctions de probabilité incluses dans R et deux méthode pour inverser numériquement une focntion de répartition.

Table des matières

1 Lois de probabilité et fonctions R

R contient la majorité des lois que nous voyons au baccalauréat. Avec les fonctions suivantes, il est possible d'évaluer certaines quantités de base directement. Voici un exemple avec $X \sim Exp\left(\frac{1}{10}\right)$. Trouver $\Pr(X \leq 5)$:

```
pexp(5, 0.1)
```

```
## [1] 0.3934693
```

Trouver $\Pr(X \geq 5)$:

```
# Deux choix possibles
pexp(5, 0.1, lower.tail = FALSE)
```

```
## [1] 0.6065307
```

```
1 - pexp(5, 0.1)
```

```
## [1] 0.6065307
```

Trouver $F_X^{-1}(0.45)$:

```
qexp(0.45, 0.1)
```

```
## [1] 5.97837
```

Trouver $E[X]$, $Var(X)$ et $E[X^3]$:

```
EX <- mexp(1, 0.1)
VarX <- mexp(2, 0.1) - (mexp(1, 0.1))^2
EX3 <- mexp(3, 0.1)
cbind(EX, EX3, VarX)
```

```
##      EX  EX3 VarX
## [1,] 10 6000  100
```

Trouver $\pi_x(5)$ et $E\left[X \cdot 1_{\{X>5\}}\right]$ et $E\left[X \cdot 1_{\{X<5\}}\right]$, on sait :

$$\begin{aligned} E[X] &= E[\max(X-d,0)] + E[\min(X,d)] \\ \pi_X &= E\left[X \cdot 1_{\{X>d\}} - d\bar{F}_X(d)\right] \\ E[X] &= E\left[X \cdot 1_{\{X>d\}}\right] + E\left[X \cdot 1_{\{X<d\}}\right] \end{aligned}$$

```
SL <- mexp(1, 0.1) - levexp(5, 0.1)
EXTrD <- SL + 5 * pexp(5, 0.1, lower.tail = FALSE)
EXTrU <- mexp(1, 0.1) - EXTrD
cbind(SL, EXTrD, EXTrU)
```

```
##            SL   EXTrD     EXTrU
## [1,] 6.065307 9.09796 0.9020401
```

On peut aussi vouloir simuler des réalisations de X :

```
set.seed(2022)
rexp(5, 0.1)
```

```
## [1]  6.319553  2.945187 22.074866  2.715817 22.682255
```

Exemple pour $X \sim Po(5)$:

```
# P(X = 3), on utilise dpois
dpois(3, 5)
```

```
## [1] 0.1403739
```

On peut utiliser la rubrique d'aide de R pour avoir plus d'informations sur les fonctions et leurs paramètres.

2 Inverser numériquement une fonction de répartition

2.1 Fonction `optimize`

Il n'est pas toujours simple d'inverser une fonction de répartition à la main. Dans le cas de la loi exponentielle, on trouve facilement :

$$F_X^{-1}(u) = -\frac{1}{\lambda}\ln(1-u)$$

Par contre, pour une loi Erlang, il n'est pas possible de le faire à la main. On a donc recours à la fonction `qgamma`. Cependant, ce ne sont pas toutes les lois qui sont programmées en R.

2.1.1 Exemple 1

On a la fonction de répartition suivante :

$$F_X(x) = 0.85 \cdot \left(1 - e^{-0.1x}\right) + 0.1 \cdot \left(1 - e^{-0.5x}\right) + 0.05 \cdot \left(1 - e^{-0.8x}\right)$$

Ce qu'on cherche, c'est :

$$F_X(x) = u$$
$$F_X(x) - u = 0$$

On veut $F_X(x)$ pour que l'équation soit égale à 0. Il faut donc avoir recours à un outil d'optimisation pour inverser cette fonction de répartition; on utilisera `optimize`.

```
P <- c(0.85, 0.1, 0.05)
lam <- c(0.1, 0.5, 0.8)
k <- 0.9

Fx <- function(x) {
    P[1] * (1 - exp(-lam[1] * x)) + P[2] * (1 - exp(-lam[2] *
        x))
    +P[3] * (1 - exp(-lam[3] * x))
}
# Si on reconnait la loi mélange d'exponentielles
Fx <- function(x) {
    P[1] * pexp(x, lam[1]) + P[2] * pexp(x, lam[2]) + P[3] *
        pexp(x, lam[3])
}

# Vérification pour intervalle Fx(20) Fx(100)
Fx_inver <- function(u) optimize(function(x) abs(Fx(x) - u),
    c(0, 100))$min

# Vérification
verif <- cbind(c(Fx_inver(k), Fx(Fx_inver(k)), Fx(20), Fx(100)))
row.names(verif) <- c("Fx_inver", "Fx(Fx_inver)", "Fx(20)", "Fx(100)")
colnames(verif) <- "Résultats"
verif
```

```
##               Résultats
## Fx_inver     21.4008990
## Fx(Fx_inver)  0.9000001
```

```
## Fx(20)         0.8849605
## Fx(100)        0.9999614
```

2.1.2 Exemple 2

On a la fonction de répartition suivante :

$$F_Y(y) = 0.4 \cdot H(y; 3, 0.1) + 0.6 \cdot B(y; 4, 5)$$

Encore une fois impossible à la main et aucune fonction préprogrammée en R, on doit utiliser `optimize`.

```
G <- c(3, 0.1)
B <- c(4, 5)
k <- 0.9

Fy <- function(y) 0.4 * pgamma(y, G[1], G[2]) + 0.6 * pbeta(y,
    B[1], B[2])

Fy_inver <- function(u) optimize(function(y) abs(Fy(y) - u),
    c(0, 120))$min

# Vérification
verif <- cbind(c(Fy_inver(k), Fy(Fy_inver(k))))
row.names(verif) <- c("Fy_inver", "Fy(Fy_inver)")
colnames(verif) <- "Résultats"
verif
```

```
##               Résultats
## Fy_inver     39.2040317
## Fy(Fy_inver)  0.9000001
```

2.2 Fonction `pinv.new`

On utilise les mêmes exemples que pour la section précédente, mais en utilisant la fonction `pinv.new` du package `Runuran`.

2.2.1 Exemple 1

On a la fonction de répartition suivante :

$$F_X(x) = 0.85 \cdot \left(1 - e^{-0.1x}\right) + 0.1 \cdot \left(1 - e^{-0.5x}\right) + 0.05 \cdot \left(1 - e^{-0.8x}\right)$$

```
library(Runuran)
gen <- pinv.new(cdf = Fx, lb = 0, ub = Inf)

Fx_inv <- function(k) uq(gen, k)

# Vérification
verif <- cbind(c(Fx_inv(k), Fx(Fx_inv(k))))
row.names(verif) <- c("Fx_inver", "Fx(Fx_inver)")
colnames(verif) <- "Résultats"
verif
```

```
##              Résultats
## Fx_inver      21.40089
## Fx(Fx_inver)   0.90000
```

2.2.2 Exemple 2

On a la fonction de répartition suivante :

$$F_Y(y) = 0.4 \cdot H(y; 3, 0.1) + 0.6 \cdot B(y; 4, 5)$$

```
library(Runuran)
gen <- pinv.new(cdf = Fy, lb = 0, ub = Inf)

Fy_inv <- function(k) uq(gen, k)

# Vérification
verif <- cbind(c(Fy_inv(k), Fy(Fy_inv(k))))
row.names(verif) <- c("Fx_inver", "Fx(Fx_inver)")
colnames(verif) <- "Résultats"
verif
```

```
##              Résultats
## Fx_inver      39.20402
## Fx(Fx_inver)   0.90000
```

3 Graphique de base

Traçons la fonction de répartition et la fonction inverse de l'exemple 1 et 2.

```
# Fonction de répartition
curve(Fx(x), xlim = c(0, 120), xlab = "x", ylab = "FX(x)", lwd = 2)
curve(Fy(x), col = "blue", lwd = 2, add = TRUE)
legend(90, 0.95, legend = c("FX", "FY"), col = c("black", "blue"),
    lty = 1:1, cex = 0.8)

# Fonction inverse
vk <- seq(0, 1, by = 0.001)
FX_inver <- sapply(vk, function(k) Fx_inver(k))
FY_inver <- sapply(vk, function(k) Fy_inver(k))
matplot(vk, FX_inver, type = "l", xlab = "u", lwd = 2, ylim = c(0,
    120))
matplot(vk, FY_inver, type = "l", xlab = "u", col = "blue", lwd = 2,
    add = TRUE)
legend(0.05, 105, legend = c("F-1X", "F-1Y"), col = c("black",
    "blue"), lty = 1:1, cex = 0.8)
```