Densité avec Réflexion

Eric Marcon

16 décembre 2024

Résumé

Estimation de la densité d'une distribution bornée.

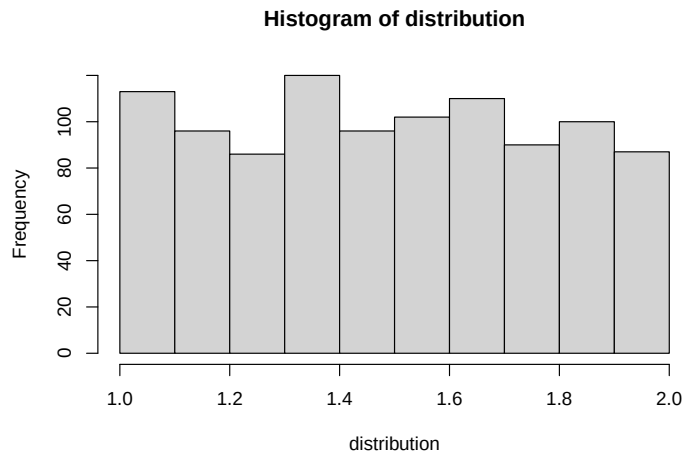
1 Problématique

L'estimation de la densité d'une distribution bornée pose problème parce que la densité estimée n'est pas nulle hors des bornes.

Le calcul de la densité d'une variable est réalisé par la fonction `density()` du package *stats*.

L'exemple suivant estime la densité de 1000 tirages d'une loi uniforme entre 1 et 2. La densité de probabilité théorique de cette distribution est 1 entre 1 et 2, et 0 hors de ces bornes. Comme l'estimation dépend principalement de la bande passante choisie, son choix est fait dès le début. La bande passante optimale (voir l'aide de la fonction `bw.SJ`) selon Sheather and Jones (1991) est calculée. Elle sera retenue par la suite pour permettre de comparer différentes méthodes.

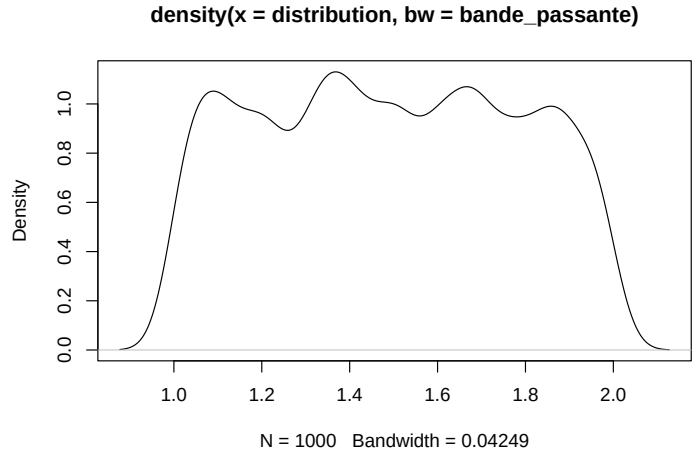
```
# Valeur des bornes
borne_gauche <- 1
borne_droite <- 2
# Tirage de 1000 valeurs
distribution <- runif(1000, min = borne_gauche, max = borne_droite)
# Histogramme de la distribution
hist(distribution)
```




```
# Choix d'une bande passante pour l'estimation
(bande_passante <- bw.SJ(distribution))
```

```
## [1] 0.04249025
```

```
# Estimation de la densité
d_reference <- density(distribution, bw = bande_passante)
plot(d_reference)
```



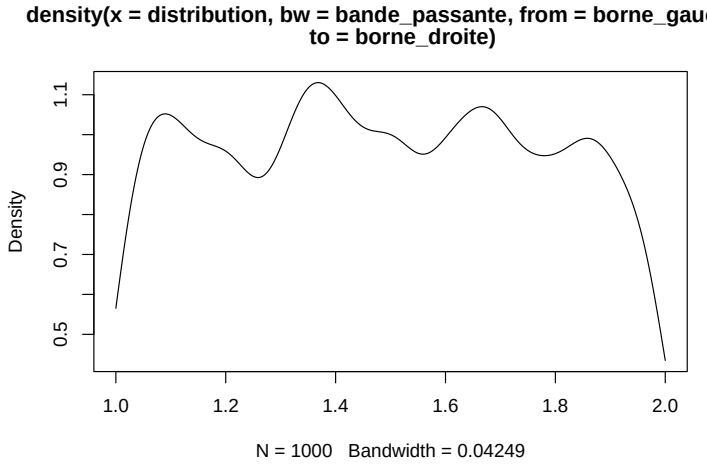
L'estimation n'est pas nulle hors de l'intervalle $[1, 2]$.

2 Solutions possibles

2.1 Censure de l'estimation

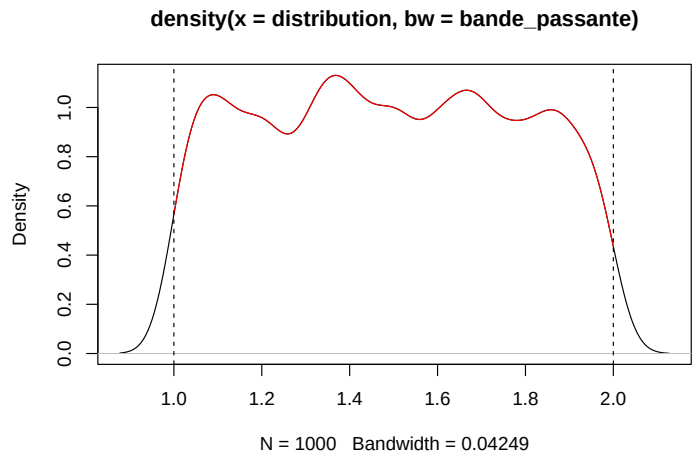
La fonction `density` accepte les arguments `from`et `to` pour censurer l'estimation aux bornes choisies.

```
d_censuree <- density(
  distribution,
  bw = bande_passante,
  from = borne_gauche, to = borne_droite
)
plot(d_censuree)
```



L'effet des arguments `from`et `to` est simplement de supprimer les valeurs estimées hors de l'intervalle.

```
plot(d_reference)
lines(d_censuree, col = "red")
abline(v=borne_gauche, lty = 2)
abline(v=borne_droite, lty = 2)
```



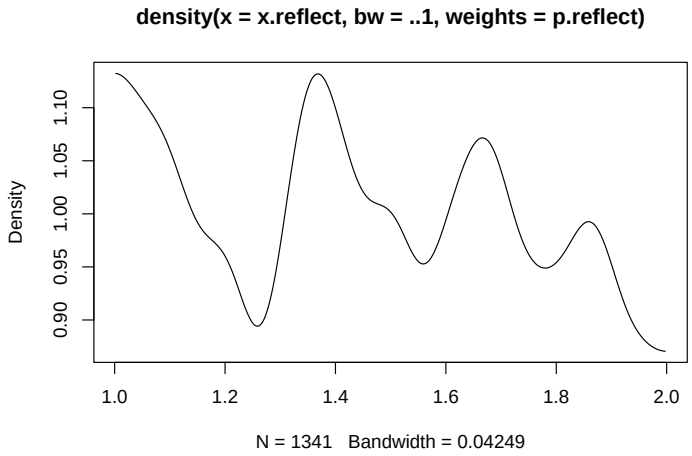
La densité censurée (en rouge) se superpose parfaitement à la densité de référence.

Cette approche n'est pas satisfaisante : les densités sont largement sous-estimées près des bornes, et la masse estimée hors des bornes est perdue ; autrement dit, la densité de probabilité de la distribution ne somme pas à 1.

2.2 Estimation avec réflexion

Le package *GoFKernel* (Pavia, 2015) propose la fonction `density.reflected` pour estimer correctement la densité précédente.

```
library("GoFKernel")
d_GoFKernel <- density.reflected(
  distribution,
  lower = borne_gauche,
  upper = borne_droite,
  bw = bande_passante
)
plot(d_GoFKernel)
```

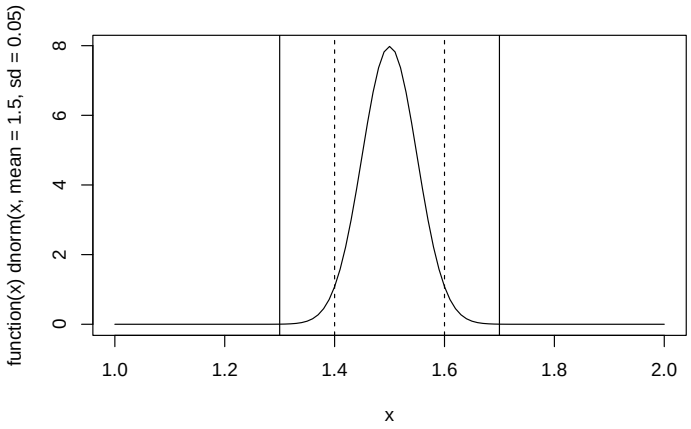


La méthode employée est décrite ci-dessous.

2.2.1 Méthode

Le noyau gaussien utilisé pour l'estimation de la densité donne un poids fort aux points de la distribution proches du point d'estimation. Le poids en fonction de la distance au point suit une loi normale dont l'écart-type est la bande passante choisie.

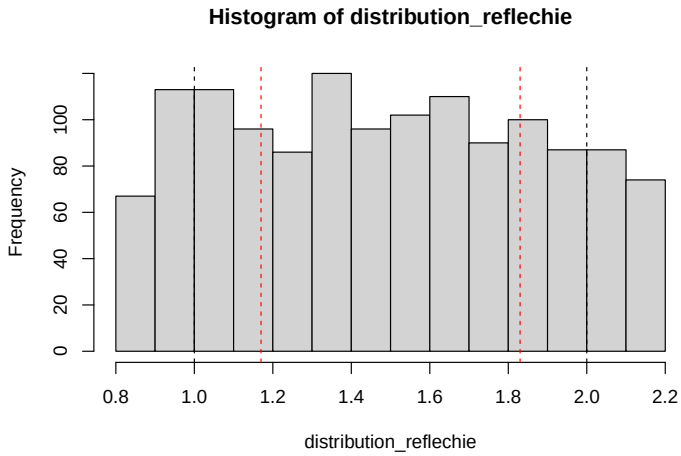
```
# Représentation du noyau
plot(function(x) dnorm(x, mean = 1.5, sd = 0.05), from  = 1, to = 2)
# moyenne +- 2 écart-type
abline(v = 1.4, lty = 2)
abline(v = 1.6, lty = 2)
# moyenne +- 4 écarts-types
abline(v = 1.3)
abline(v = 1.7)
```



La figure représente le poids des points observés entre 1 et 2 pour l'estimation de la densité au point 1,5 dans un noyau gaussien d'écart-type (bande passante) égal à 0,05. Le poids des voisins de 1,5 diminue avec la distance. Au-delà de 4 écart-types, le poids des voisins est négligeable.

La méthode utilisée consiste à reproduire les données observées en miroir par rapport aux bornes (Silverman, 1986). L'estimation près des bornes utilise les données mises en miroir : comme leur influence est négligeable au-delà de 4 bandes passantes, seules les données situées à moins de 4 bandes passantes des bornes sont reproduites.

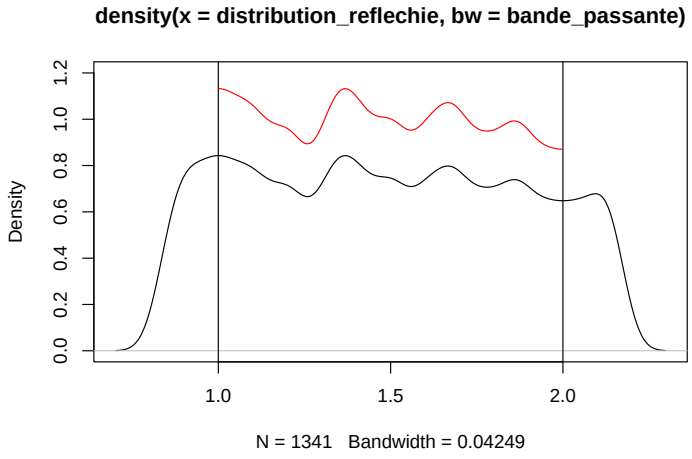
```
# Réflexion
a_reflechir_gauche <- which(distribution < 4*bande_passante + borne_gauche)
a_reflechir_droite <- which(distribution > borne_droite - 4*bande_passante)
distribution_reflechie <- c(
  distribution,
  2 * borne_gauche - distribution[a_reflechir_gauche],
  2 * borne_droite-distribution[a_reflechir_droite]
)
hist(distribution_reflechie)
abline(v = borne_gauche, lty = 2)
abline(v = borne_droite, lty = 2)
abline(v = 4 * bande_passante + borne_gauche, lty = 2, col = "red")
abline(v = borne_droite - 4 * bande_passante, lty = 2, col = "red")
```



L'histogramme des données complétées montre la réplication des données autour des bornes (limites verticales noires), limitées à 4 bandes passantes (limites verticales rouges).

La nouvelle estimation de la densité (en noir) est presque identique à celle de la fonction `density.reflected` (en rouge) mais une partie de la masse se trouve hors des bornes, réduisant forcément la valeur de la densité entre les bornes.

```
# Nouvelle estimation de la densité
d_reflechie <- density(distribution_reflechie, bw=bande_passante)
plot(d_reflechie, ylim = c(0,1.2))
abline(v = borne_gauche)
abline(v = borne_droite)
lines(d_GoFKernel, col = "red")
```



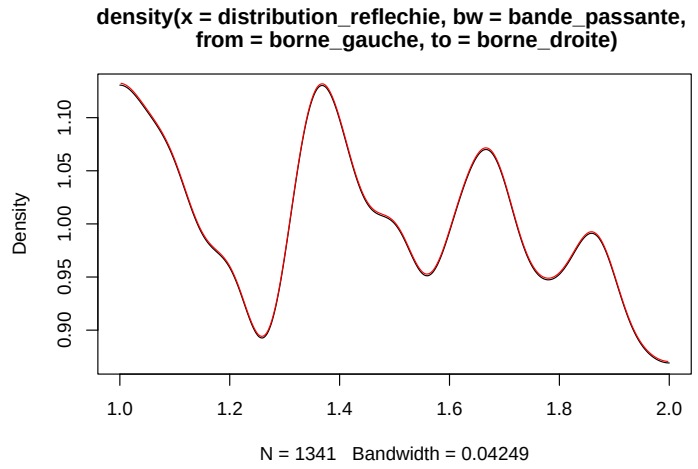
Il reste donc à censurer l'estimation aux bornes et à la renormaliser pour qu'elle somme à 1. L'intégrale de la densité censurée est calculée en multipliant la valeur moyenne entre deux estimations successives de la densité par la largeur de l'intervalle qui les sépare.

```
# Filtrage de la densité dans les bornes
d_reflechie <- density(
  distribution_reflechie,
  bw = bande_passante,
  from = borne_gauche,
  to = borne_droite
)
# Intégrale de la densité censurée
(integrale <- sum(
  (d_reflechie$y[-1] + d_reflechie$y[-length(d_reflechie$y)]) /
  2 * diff(d_reflechie$x)
))
```

```
## [1] 0.7457114
```

La renormalisation consiste simplement à diviser toutes les valeurs estimées par l'intégrale.

```
d_reflechie$y <- d_reflechie$y / integrale
# Vérification
plot(d_reflechie)
lines(d_GoFKernel, col = "red")
```

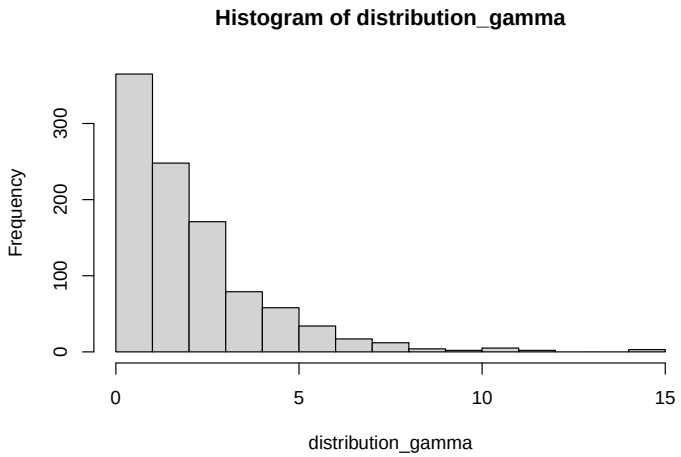


La densité estimée est maintenant celle de `density.reflected`.

2.2.2 Borne unique

L'exemple suivant traite le cas d'une variable bornée d'un seul côté. Les données sont tirées dans une loi γ de forme 1 et échelle 2. La bande passante est fixée à 0,15 pour les comparaisons entre méthodes.

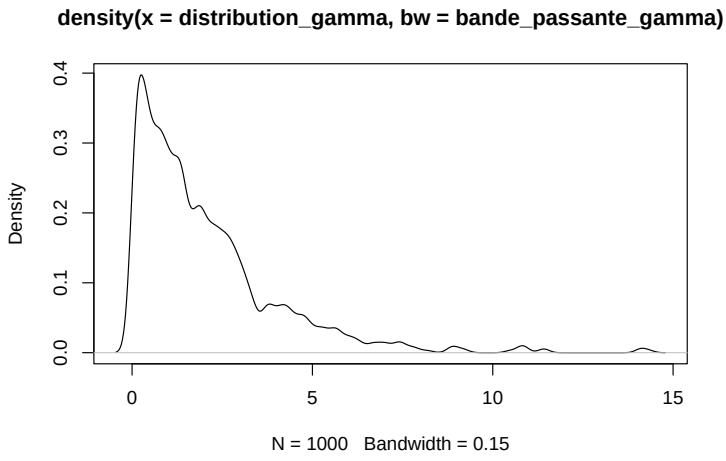
```r
# Tirage de 1000 valeurs
distribution_gamma <- rgamma(1000, shape = 1, scale = 2)
borne_gauche <- 0
# Histogramme de la distribution
hist(distribution_gamma)
```




```r
# Choix d'une bande passante pour l'estimation
bw.SJ(distribution_gamma)
```

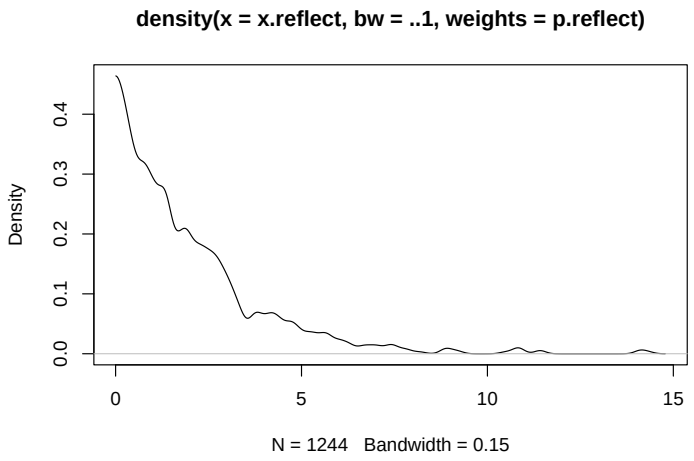
```
## [1] 0.1620609
```

```r
bande_passante_gamma <- 0.15
# Estimation de la densité
d_reference <- density(distribution_gamma, bw = bande_passante_gamma)
plot(d_reference)
```



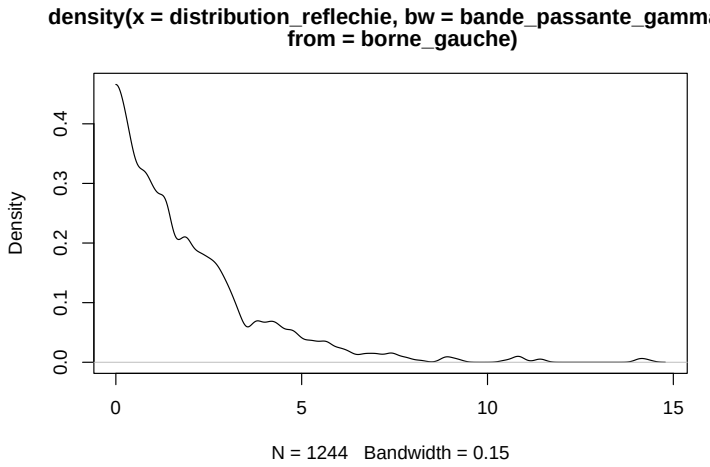
La densité doit être nulle pour les valeurs négatives. Avec le package *GoF-Kernel*, seul le paramètre `lower` doit être précisé.

```
d_GoFKernel <- density.reflected(
  distribution_gamma,
  lower = borne_gauche,
  bw = bande_passante_gamma
)
plot(d_GoFKernel)
```



Le code complet pour obtenir cette estimation est le suivant :

```
# Réflexion
a_reflechir_gauche <- which(
  distribution_gamma < 4 * bande_passante_gamma + borne_gauche
)
distribution_reflechie <- c(
  distribution_gamma,
  2 * borne_gauche - distribution_gamma[a_reflechir_gauche]
)
# Densité
d_reflechie <- density(
  distribution_reflechie,
  bw = bande_passante_gamma,
  from = borne_gauche
)
# Renormalisation
integrale <- sum(
  (d_reflechie$y[-1] + d_reflechie$y[-length(d_reflechie$y)]) /
  2 * diff(d_reflechie$x)
)
d_reflechie$y <- d_reflechie$y / integrale
# Figure
plot(d_reflechie)
```

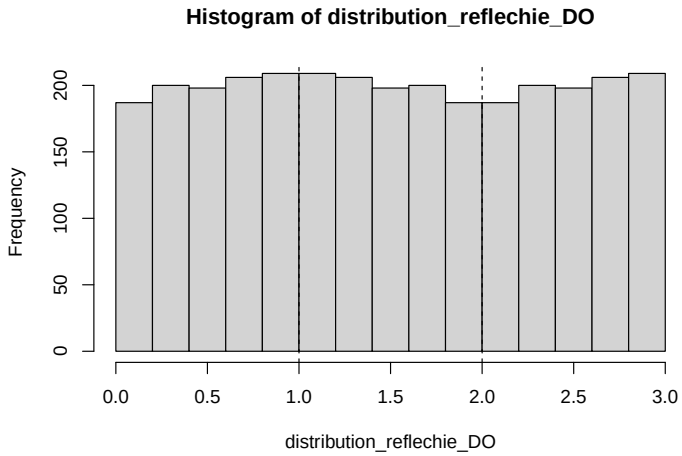


2.2.3 Méthode de Duranton and Overman (2005)

Duranton and Overman (2005) estiment la densité de probabilité de trouver un voisin en fonction de la distance à un point de référence quelconque, à partir d'une matrice de distance entre des points situés dans un espace en deux dimensions. La densité doit être nulle pour les distances négatives. La technique des auteurs est simplement de dupliquer *toutes* les distances en miroir (sans se limiter à 4 bandes passantes) par rapport à 0 avant l'estimation de la densité. L'avantage de la méthode est que le calcul de l'intégrale est inutile pour renormaliser la densité : il suffit de doubler les valeurs puisque la masse des données a été doublée. L'inconvénient est l'ajout de calculs inutiles hors des bornes et la perte de précision de l'estimation dans les bornes : la moitié des 512 points d'estimation sont hors des bornes.

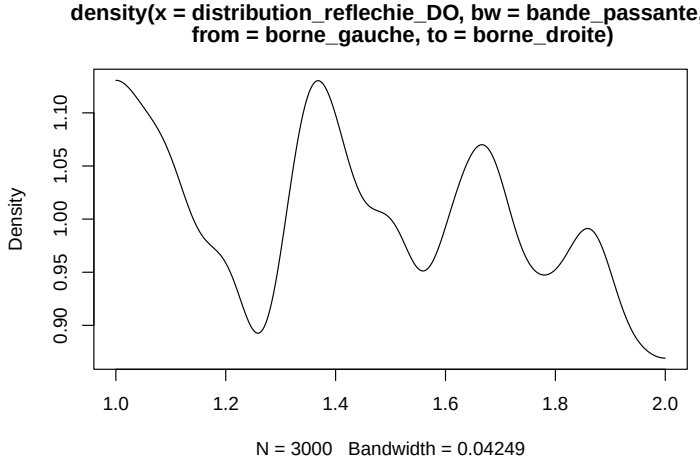
La méthode peut être généralisée ici en dupliquant les données à gauche et à droite :

```
# Valeur des bornes
borne_gauche <- 1
borne_droite <- 2
# Réflexion de toutes les données, à gauche et à droite
distribution_reflechie_DO <- c(
  distribution,
  2 * borne_gauche - distribution,
  2 * borne_droite - distribution
)
# Histogramme
hist(distribution_reflechie_DO)
abline(v = borne_gauche, lty = 2)
abline(v = borne_droite, lty = 2)
```



La densité est calculée correctement, et doit être renormalisée en la multipliant par 3 (la masse des données a été triplée) :

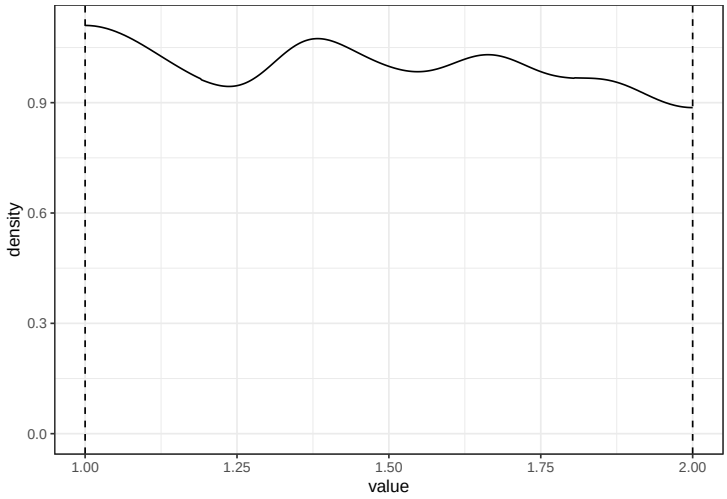
```r
# Densité censurée
d_DO <- density(
  distribution_reflechie_DO,
  bw = bande_passante,
  from = borne_gauche,
  to = borne_droite
)
# Renormalisation
d_DO$y <- d_DO$y * 3
# Vérification
plot(d_DO)
```



2.3 ggplot

Avec *ggplot2*, l'argument `bounds` de `geom_density()` permet de borner l'estimation de la densité.

```
library("tidyverse")
distribution %>%
  as_tibble %>%
  ggplot() +
  geom_density(aes(x = value), bounds = c(borne_gauche, borne_droite)) +
  geom_vline(xintercept = c(borne_gauche, borne_droite), lty = 2)
```



3 Conclusion

En pratique, pour estimer une densité bornée, utiliser le package *GoFKernel* pour sa simplicité. Pour une simple représentation graphique, utiliser *ggplot2*.

Si la dépendance à un package n'est pas souhaitable (par exemple dans un nouveau package), l'algorithme suivant peut être utilisé :

- dupliquer les données en miroir autour de la borne, jusqu'à 4 bandes passantes de la borne ;

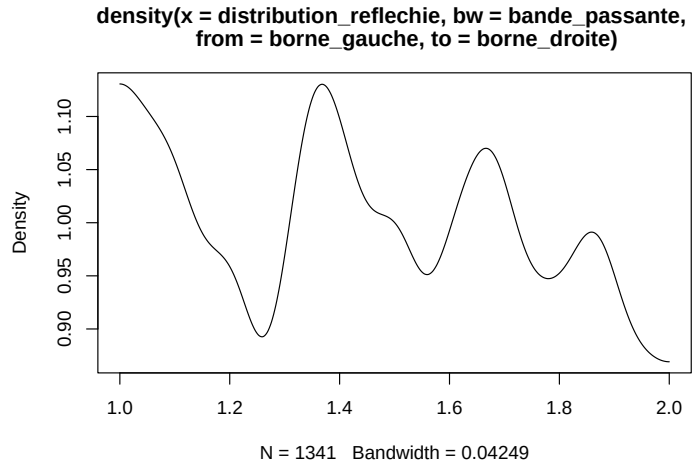
```
# Valeur des bornes
borne_gauche <- 1
borne_droite <- 2
# Choix d'une bande passante pour l'estimation
bande_passante <- bw.SJ(distribution)
# Réflexion
a_reflechir_gauche <- which(distribution < 4 * bande_passante + borne_gauche)
a_reflechir_droite <- which(distribution > borne_droite - 4 * bande_passante)
distribution_reflechie <- c(
  distribution,
  2 * borne_gauche - distribution[a_reflechir_gauche],
  2 * borne_droite - distribution[a_reflechir_droite]
)
```

- estimer la densité avec la fonction `density`, en censurant l'estimation aux bornes ;

```
# Densité. Bande passante par défaut.
d_reflechie <- density(
  distribution_reflechie,
  bw = bande_passante,
  from = borne_gauche,
  to = borne_droite
)
```

- calculer l'intégrale de la densité obtenue et renormaliser les densités estimées par cette intégrale.

```
# Renormalisation
integrale <- sum(
  (d_reflechie$y[-1] + d_reflechie$y[-length(d_reflechie$y)]) /
    2 * diff(d_reflechie$x)
)
d_reflechie$y <- d_reflechie$y / integrale
# Figure
plot(d_reflechie)
```



Références

Duranton, G. and H. G. Overman (2005). Testing for Localisation Using Micro-Geographic Data. *Review of Economic Studies 72*(4), 1077–1106.

Pavia, J. M. (2015). Testing Goodness-of-Fit with the Kernel Density Estimator : GoFKernel. *Journal of Statistical Software 66*(Code Snippet 1).

Sheather, S. J. and M. C. Jones (1991). A Reliable Data-Based Bandwidth Selection Method for Kernel Density Estimation. *Journal of the Royal Statistical Society : Series B (Statistical Methodology) 53*(3), 683–690.

Silverman, B. W. (1986). *Density estimation for statistics and data analysis*. London : Chapman and Hall.