

Normalisation d'une variable quantitative : la transformation de Box et Cox

Pr Jean R. LOBRY

La fonction de BOX et COX (1964) est une généralisation astucieuse de la transformation logarithmique qui permet, dans des situations favorables, de justifier objectivement de l'utilisation d'icelle. Dans des situations moins favorables elle permet *a minima* de proposer une transformation qui permet de mieux appréhender la variabilité des données. L'objectif de cette fiche de TD est de vous faire pratiquer cette transformation sur des données réelles pour vous familiariser avec l'outil.

Table des matières

1	Introduction	3
2	Visualisation de la normalité	4
2.1	Distribution normale	4
2.2	Distribution symétrique à grosses queues	5
2.3	Distribution symétrique à petites queues	5
2.4	Distribution symétrique bimodale	6
2.5	Distribution asymétrique à droite	7
2.6	Distribution asymétrique à gauche	7
3	Exemples concrets de non-normalité	7
3.1	Masse de 62 mammifères	7
3.2	Volume des testicules de 150 chats	8
3.3	Salaire de 828 joueurs de baseball	9
3.4	Salaire de (2165 + 1) contrats de collaborateurs parlementaires .	10
3.5	Dispersion de 70 carnivores	11
3.6	Âge de 236 individus	13
4	La transformation de Box et Cox	14

5	Estimation de λ basée sur les observations originelles	16
5.1	Données simulées	16
5.2	Masse de 62 mammifères	17
5.3	Volume des testicules de 150 chats	18
5.4	Salaire de 828 joueurs de baseball	19
5.5	Salaire de (2165 + 1) contrats de collaborateurs parlementaires .	20
5.6	Dispersion de 70 carnivores	21
5.7	Âge de 236 individus	22
6	Estimation de λ dans le cadre d'un modèle linéaire	23
6.1	Données simulées	23
6.2	Volume des testicules de 150 chats	25
6.3	Salaire de 828 joueurs de baseball	27
6.4	Âge de 236 individus	31
7	La fonction <code>boxcox()</code> du paquet MASS	32
8	Ressources francophones en ligne	37
	Références	37

1 Introduction

On peut envisager les transformations de variables comme n'étant rien de plus qu'une ré-expression des données dans des unités différentes [18]. De nombreuses transformations cachées sont présentes dans la vie de tous les jours, par exemple une température exprimée en degré CELSIUS ($^{\circ}\text{C}$) n'est rien d'autre qu'une température exprimée en kelvin (K) translatée de +273,15 unités. Le pH est une transformation logarithmique de la concentration en ions hydronium :

$$\text{pH} = -\log_{10}[\text{H}_3\text{O}^+]$$

En optique, l'absorbance A d'un milieu est une transformation logarithmique du rapport entre l'intensité énergétique incidente I_0 et de l'intensité énergétique transmise I :

$$A = \log_{10} \frac{I_0}{I}$$

En musique, l'octave E est une unité logarithmique du rapport entre une fréquence f_1 et une fréquence f_0 :

$$E = \log_2 \frac{f_1}{f_0}$$

En épidémiologie, l'« *odd ratio* » OD encore appelé « rapport des cotes » ou « rapport des facilités ¹ » de deux événements de probabilité p_1 et p_2 est souvent utilisé après une transformation logarithmique :

$$\ln \text{OD} = \log_e \frac{p_1/(1-p_1)}{p_2/(1-p_2)}$$

En enzymologie, la transformation dite en « double inverse » de LINEWEAVER et BURK [22] est utilisée pour linéariser le modèle de MICHAELIS et MENTEN [24]. Bien d'autres transformations ont été proposées à cet effet (*cf.* la fiche tdr47²). En biologie comparative, les études allométriques utilisent une transformation dite en « double log » des données (*cf.* la fiche tdr333³).

Les transformations de variables sont souvent utilisées pour induire des propriétés (*e.g.* normalité, homoscedasticité, linéarité) désirables pour la représentation des données, les tests paramétriques ou les procédures d'estimation de paramètres. Ce ne sont pas des « tripatouillages honteux » utilisés pour cacher quoi que ce soit mais bien le rebours des outils très utiles pour analyser et comprendre les données [2]. Pour reprendre *mutatis mutandis* une phrase de Philippe BRENOT [4] : « je l'avoue publiquement et comme un acte expiatoire : oui, j'ai pratiqué la transformation de variables ... et à plusieurs reprises ! ».

On s'intéresse ici à la transformation proposée par BOX et COX en 1964 pour normaliser les données dans un article [3] notoire puisqu'il a été cité plus de 12000 fois dans des articles scientifiques 50 ans après sa publication.

1. Terminologie de Pierre-Simon LAPLACE [21], voir exemple d'utilisation dans <https://pbil.univ-lyon1.fr/R/pdf/qro.pdf>
2. <https://pbil.univ-lyon1.fr/R/pdf/tdr47.pdf>
3. <https://pbil.univ-lyon1.fr/R/pdf/tdr333.pdf>

2 Visualisation de la normalité

Le caractère gaussien ou non-gaussien d'une variable quantitative n'est pas forcément le plus simple à apprécier sur une représentation directe de type histogramme. On définit ici une fonction utilitaire pour nous aider dans cette tâche en représentant côte à côte :

- 1° la distribution des valeurs à l'aide d'un histogramme auquel on superpose un estimateur de la densité locale (soyons résolument modernes comme dans MASS [31]);
- 2° une représentation en coordonnées semi-logarithmiques de l'estimateur de la densité locale auquel on superpose ce qui serait attendu sous une loi normale ainsi que la valeur du coefficient d'asymétrie (voir la fiche⁴ « risques, puissance et robustesse » pour plus de détails sur ce coefficient);
- 3° la droite de HENRY, se référer à la fiche⁵ « graphes quantiles-quantiles » pour plus de détails (soyons résolument rétrogrades).

```
library(e1071) # pour skewness()
myplot <- function(x, mylwd = 3, mycol = rgb(0, 0.5, 0.5), myborder = grey(0.8),
  mycolhist = rgb(0, 1, 1, 0.1), mycolnorm = rgb(0.7, 0.5, 0.7), ...){
  old.par <- par(no.readonly = TRUE)
  on.exit(par(old.par))
  par(mfrow = c(1, 3))
  hx <- hist(x, plot = FALSE)
  dx <- density(x)
  hist(x, freq = FALSE, border = myborder, col = mycolhist, ylim = c(0, max(hx$density, dx$y)), ...)
  lines(dx, lwd = mylwd, col = mycol)
  plot(dx$x, log10(dx$y), type = "l", main = "Coordonnées semi-logarithmiques", xlab = "x",
    ylab = "log10(Density)", las = 1)
  legend("center", legend = round(skewness(x), 4), cex = 2, text.col = mycol, bty = "n",
    adj = 0.5)
  lines(dx$x, log10(dnorm(dx$x, mean(x), sd(x))), col = mycolnorm, lwd = mylwd)
  qqnorm(x, main = "Droite de Henry", xlab = "Quantiles théoriques",
    ylab = "Quantiles observés", las = 1)
  qqline(x)
}
```

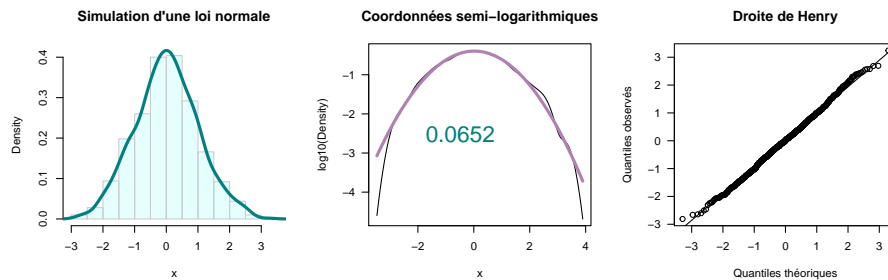
2.1 Distribution normale

Dans le cas d'une distribution normale on s'attend à avoir :

- 1° une courbe « en cloche » ;
- 2° une parabole et un coefficient d'asymétrie proche de 0 ;
- 3° une droite.

```
set.seed(123)
x <- rnorm(10^3)
myplot(x, main = "Simulation d'une loi normale")
```

4. <http://pbil.univ-lyon1.fr/R/pdf/tdr36.pdf>
 5. <http://pbil.univ-lyon1.fr/R/pdf/tdr22.pdf>

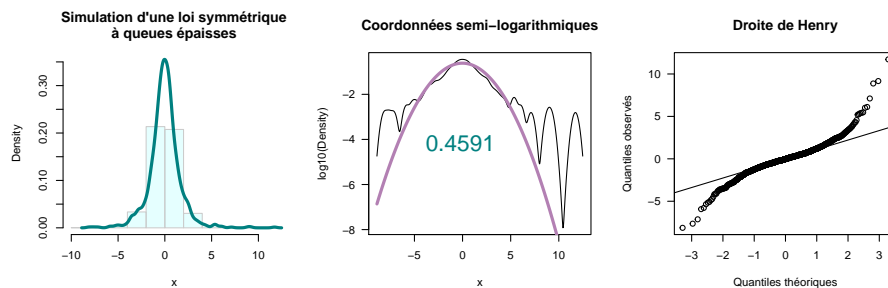


2.2 Distribution symétrique à grosses queues

Dans le cas des distributions symétriques à grosses queues⁶ on s'attend à avoir :

- 1° une courbe symétrique resserrée sur ses valeurs centrales (on peut se représenter la distribution δ de DIRAC comme un cas limite) ;
- 2° une courbe au dessus de la parabole pour les valeurs distales et un coefficient d'asymétrie proche de 0 ;
- 3° une courbe avec un point d'inflexion pour 0 et une concavité tournée vers le bas avant et vers le haut après.

```
x <- rt(10^3, 3)
myplot(x, main = "Simulation d'une loi symétrique à queues épaisses")
```



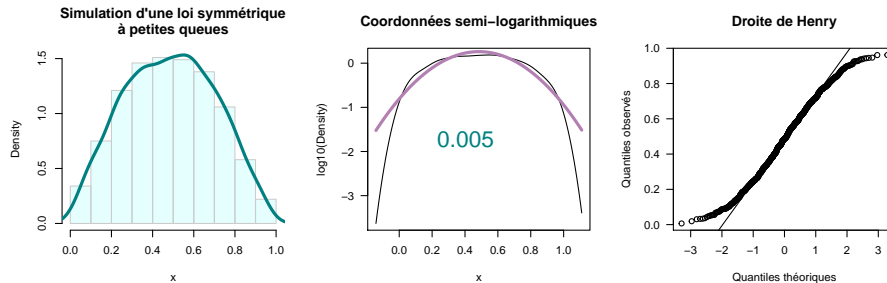
2.3 Distribution symétrique à petites queues

On s'attend à avoir :

- 1° une courbe symétrique peu resserrée sur ses valeurs centrales (on peut se représenter la distribution uniforme comme un cas limite) ;
- 2° une courbe au dessous de la parabole pour les valeurs distales et un coefficient d'asymétrie proche de 0 ;
- 3° une courbe avec un point d'inflexion pour 0 et une concavité tournée vers le haut avant et vers le bas après.

6. Aux âmes chagrines qui oseraient se gausser, dans un contexte gaussien, de notre traduction ambivalente et approximative du terme « *fat tail* » sachez que nous sommes capables de bien pire : <http://pbil.univ-lyon1.fr/R/pdf/expd.pdf>

```
x <- rbeta(10^3, 2, 2)
myplot(x, main = "Simulation d'une loi symétrique\nà petites queues")
```

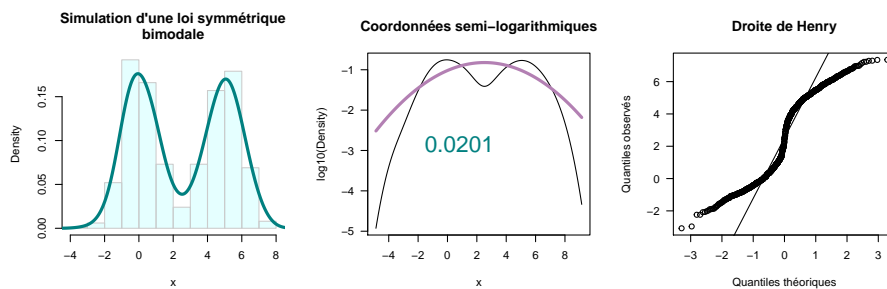


2.4 Distribution symétrique bimodale

On s'attend à avoir :

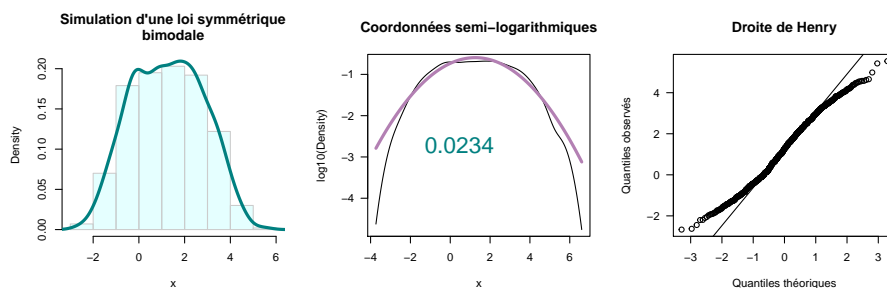
- 1° une distribution bimodale ;
- 2° une distribution bimodale et un coefficient d'asymétrie proche de 0 ;
- 3° une courbe en « S ».

```
x <- c(rnorm(500), rnorm(500, mean = 5))
myplot(x, main = "Simulation d'une loi symétrique\nbimodale")
```



Notez que si les groupes sont mal résolus on retombe sur le cas de la distribution symétrique à petite queues :

```
x <- c(rnorm(500), rnorm(500, mean = 2.5))
myplot(x, main = "Simulation d'une loi symétrique\nbimodale")
```

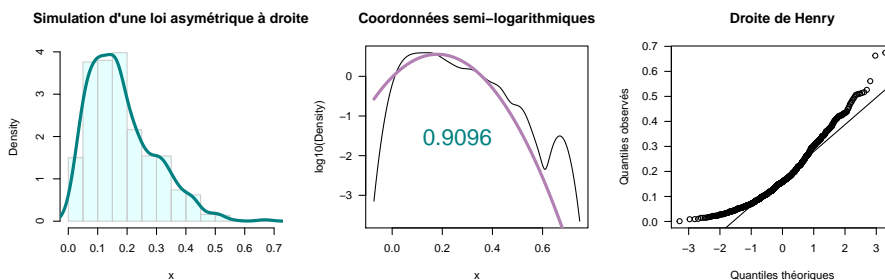


2.5 Distribution asymétrique à droite

On s'attend à avoir :

- 1° une distribution avec une petite queue à gauche et une grosse à droite ;
- 2° une courbe en dessous de la parabole à gauche et au dessus à droite et un coefficient d'asymétrie positif ;
- 3° une courbe avec la concavité tournée vers le haut (la dérivée seconde est du même signe que le coefficient d'asymétrie).

```
x <- rbeta(1000, 2, 9)
myplot(x, main = "Simulation d'une loi asymétrique à droite")
```

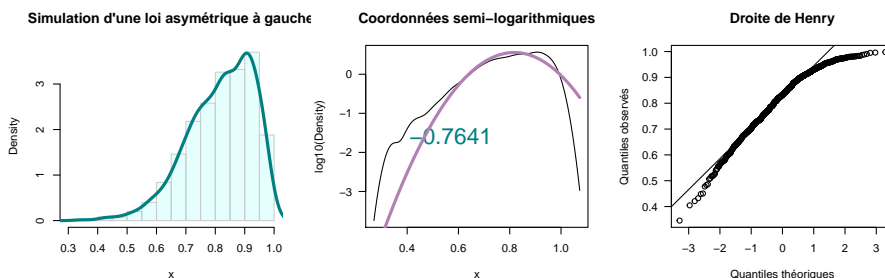


2.6 Distribution asymétrique à gauche

On s'attend à avoir :

- 1° une distribution avec une grosse queue à gauche et petite à droite ;
- 2° une courbe au dessus de la parabole à gauche et en dessous à droite et un coefficient d'asymétrie négatif ;
- 3° une courbe avec la concavité tournée vers le bas (la dérivée seconde est du même signe que le coefficient d'asymétrie).

```
x <- rbeta(1000, 9, 2)
myplot(x, main = "Simulation d'une loi asymétrique à gauche")
```



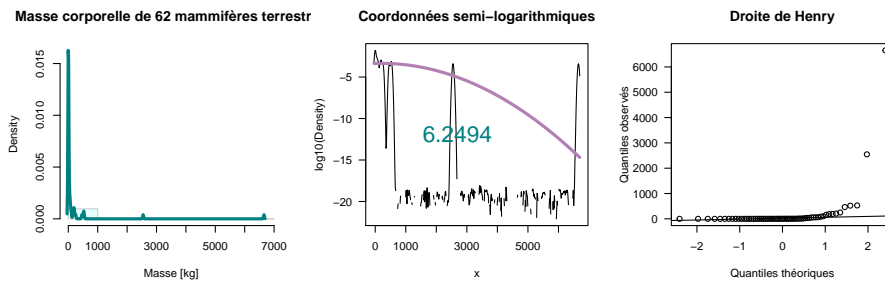
3 Exemples concrets de non-normalité

3.1 Masse de 62 mammifères

Une situation typique est celle des variables morphologiques dans les études inter-spécifiques. Prenons par exemple la distribution de la masse corporelle de

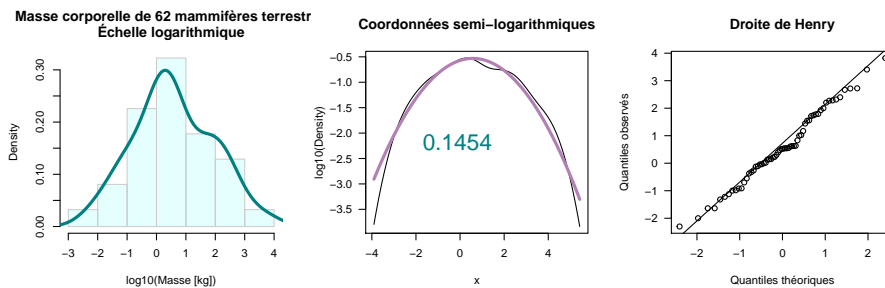
62 mammifères terrestres, données disponibles dans le paquet MASS [31] et issues de [1].

```
library(MASS)
data(mammals)
myplot(mammals$body, xlab = "Masse [kg]",
       main = "Masse corporelle de 62 mammifères terrestres")
```



La distribution est très fortement asymétrique à droite, ne serait-ce qu'à cause de l'éléphant d'Afrique et de l'éléphant d'Asie qui sont beaucoup plus lourds que les autres espèces. Une solution classique est d'utiliser une échelle logarithmique :

```
myplot(log10(mammals$body), xlab = "log10(Masse [kg])",
       main = "Masse corporelle de 62 mammifères terrestres\nÉchelle logarithmique")
```



La transformation est très pertinente ici puisque l'on peut maintenant considérer que la distribution est normale :

```
shapiro.test(log10(mammals$body))
      Shapiro-Wilk normality test
data:  log10(mammals$body)
W = 0.98645, p-value = 0.7272
```

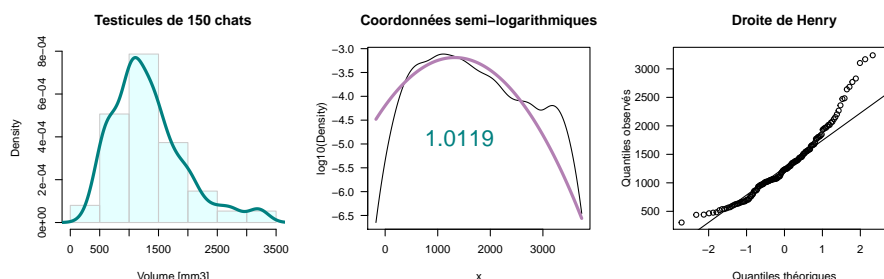
3.2 Volume des testicules de 150 chats

On reprend les données décrites dans la fiche tdr334⁷ « tailles des testicules et systèmes d'appariement » et issue de [28]. On s'intéresse au volume des testicules (en mm³).

```
chasay <- read.table("https://pbil.univ-lyon1.fr/R/donnees/chasay.txt", header = TRUE)
```

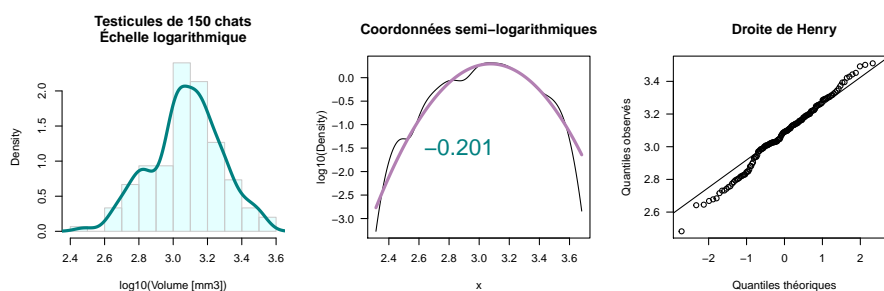
7. <https://pbil.univ-lyon1.fr/R/pdf/tdr334.pdf>


```
myplot(chasay$vol, main = "Testicules de 150 chats", xlab = "Volume [mm3]")
```



Représentez l'effet d'une transformation logarithmique et testez si l'on peut considérer les données redressées comme suivant une loi normale :

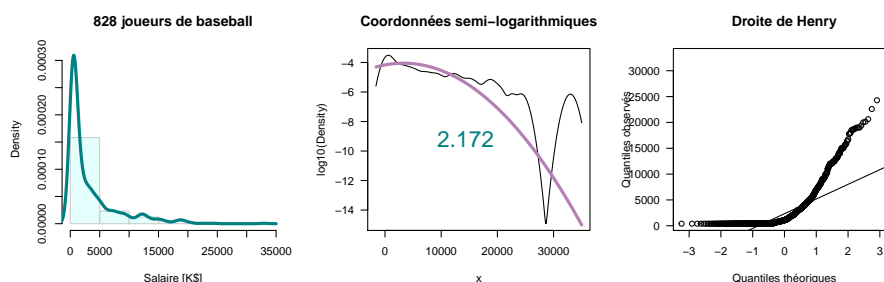
```
Shapiro-Wilk normality test
data: x
W = 0.98987, p-value = 0.3546
```



3.3 Salaire de 828 joueurs de baseball

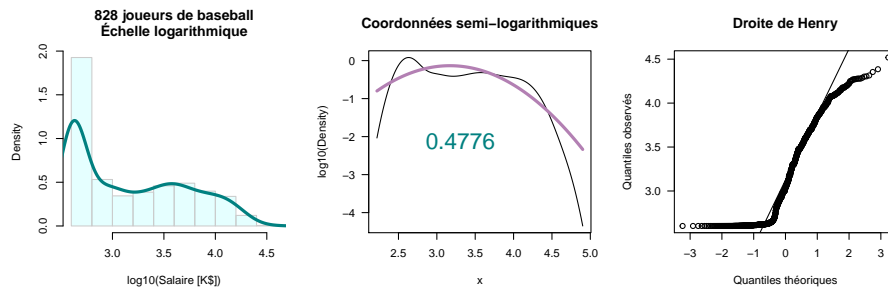
Les données ont été compilées par les auteurs du paquet `openintro` [11].

```
library(openintro)
data(MLB)
myplot(MLB$salary, main = "828 joueurs de baseball",
       xlab = "Salaire [K$]")
```



Représentez l'effet d'une transformation logarithmique et testez si l'on peut considérer les données redressées comme suivant une loi normale :

```
Shapiro-Wilk normality test
data: x
W = 0.87465, p-value < 2.2e-16
```

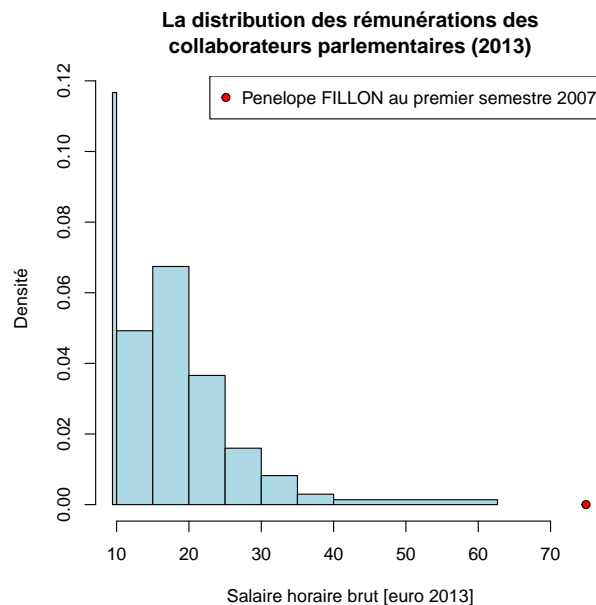


3.4 Salaire de (2165 + 1) contrats de collaborateurs parlementaires

Pour l'origine et la description des données voir la figure 1 page 12. Importer les données sous R et faire une représentation graphique :

```
dta <- read.table("https://pbil.univ-lyon1.fr/R/donnees/SalColParl.txt",
  header = TRUE, dec = ",", sep = "\t")

dta$med <- (dta$min + dta$max)/2
x <- with(dta, rep(med, n))
PFI <- 9.43*(10167/151.67)/8.44
hist(x, breaks = c(dta$min, max(dta$max)), col = "lightblue",
  main = "La distribution des rémunérations des collaborateurs parlementaires (2013)",
  xlim = c(min(x), PFI), xlab = "Salaire horaire brut [euro 2013]",
  ylab = "Densité")
points(PFI, 0, pch = 21, bg = "red")
legend("topright", legend = "Penelope FILLON au premier semestre 2007", pch = 21, pt.bg = "red")
```

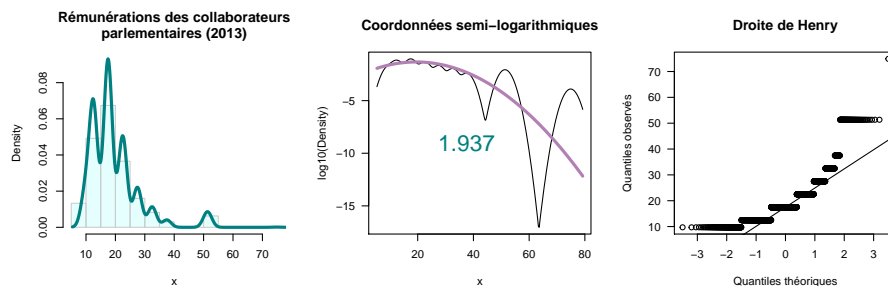


Les rémunérations sont très dispersées avec beaucoup de contrats voisins du SMIC et peu de contrats à plus de 40 € de l'heure. Le salaire de Penelope FILLON est supérieur à tous les autres, mais on aimerait être plus précis et

quantifier le phénomène pour pouvoir répondre à la question « ce salaire est-il anormalement élevé ? ». Pour ce faire, nous avons besoin de modéliser les données avec une fonction de densité de probabilité continue. Commençons par vérifier que la loi normale ne convient pas :

```
x <- c(x, PFI)
myplot(x, main = "Rémunérations des collaborateurs\nparlementaires (2013)")
shapiro.test(x)

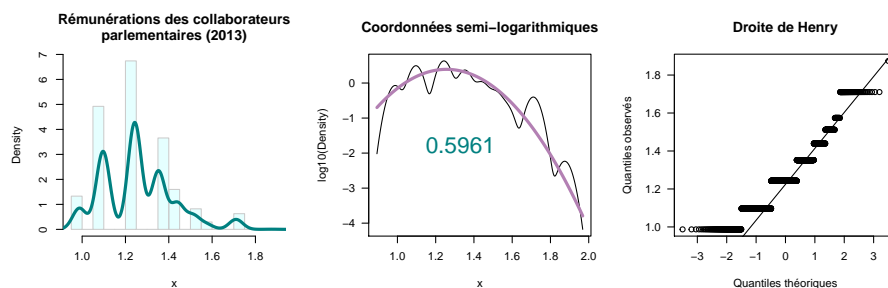
Shapiro-Wilk normality test
data:  x
W = 0.79712, p-value < 2.2e-16
```



On a une distribution fortement asymétrique à droite. Notez l'aspect caractéristique de la droite de Henry avec ses sauts. C'est typique des jeux de données où l'on ne dispose pas comme ici des valeurs brutes mais de celles après regroupement par intervalles, on retrouve graphiquement ce découpage en tranches. Une situation similaire se rencontre également lorsque la précision de la mesure est faible par rapport à l'amplitude des données. Testez la transformation logarithmique :

```
myplot(log10(x), main = "Rémunérations des collaborateurs\nparlementaires (2013)")
shapiro.test(log10(x))

Shapiro-Wilk normality test
data:  log10(x)
W = 0.923, p-value < 2.2e-16
```



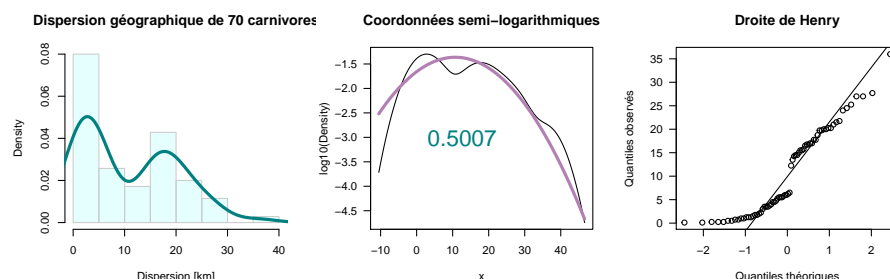
On a amélioré les choses, mais la distribution est toujours asymétrique à droite, on ne pourra pas utiliser la loi normale ici. Il sera intéressant de voir si la transformation de Box et Cox fait mieux.

3.5 Dispersion de 70 carnivores

La dispersion géographique (**range**) de 70 carnivores disponible dans le jeu de données **carni70** du paquet **ade4** [5] et issue de [12]. Représentez la distribution des données :

Tranche de rémunération	Nombre de contrats	Pourcentage des contrats concernés sur l'ensemble
de 9,43 à 9,99 €	144	6,65 %
de 10 à 14,99 €	533	24,62 %
de 15 à 19,99 €	730	33,72 %
de 20 à 24,99 €	396	18,29 %
de 25 à 29,99 €	173	7,99 %
de 30 à 34,99 €	89	4,11 %
de 35 à 39,99 €	32	1,48 %
de 40 à 62,66 €	68	3,14 %

FIGURE 1 – Le tableau des données. Copie d'écran d'une partie de la page 6 du document intitulé « Réunion du 10 juillet 2013 avec les représentants des associations et syndicats de collaborateurs parlementaires. Éléments statistiques et de bilan (juin 2013) ». Les rémunérations sont exprimées en euros bruts de l'heure. En janvier 2013, le SMIC horaire brut était de 9,43 euros. Il y a un total de 2165 contrats actifs qui est supérieur au nombre de collaborateurs (2090) du fait du cumul de plusieurs contrats pour certaines personnes. Dans son édition du 1^{er} février 2017, le *Canard enchaîné* rapporte que le salaire brut mensuel de Penelope FILLON était de 10167 euros entre le 1^{er} janvier et le 31 août 2007. En supposant un contrat de travail à plein temps de 151,67 heures mensuelles, cela correspond à un taux horaire de 67 euros de 2007. En 2007, le SMIC horaire brut était de 8,44 euros. En euros de 2013, le salaire horaire brut de Penelope FILLON est donc de 74,9. Source : <http://blogs.lexpress.fr/cuisines-assemblee/wp-content/blogs.dir/669/files/2013/07/Elements-rapport-dactivite-juillet-2013.pdf>, dernière consultation le 2017-02-01.

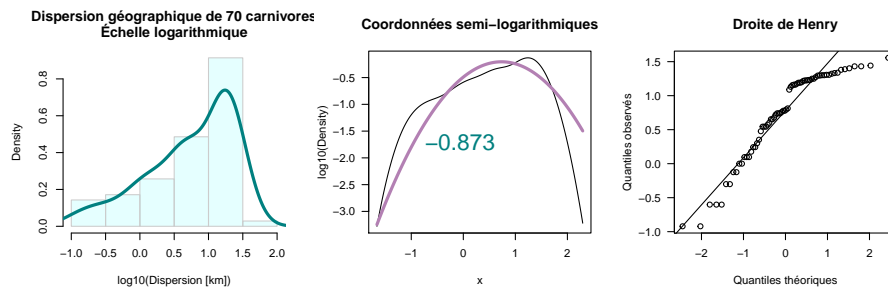


Représentez l'effet d'une transformation logarithmique et testez si l'on peut considérer les données redressées comme suivant une loi normale :

```

Shapiro-Wilk normality test
data: x
W = 0.89017, p-value = 1.62e-05

```



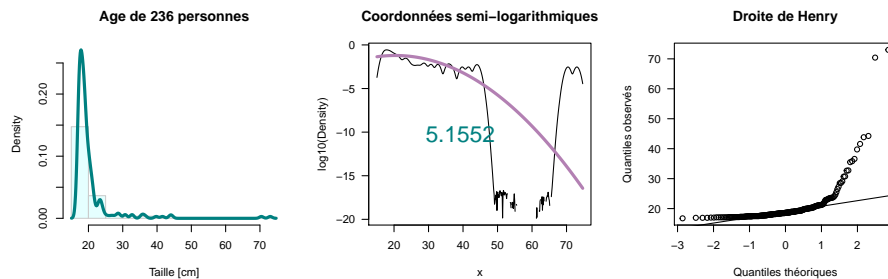
3.6 Âge de 236 individus

On extrait du jeu de données `survey` du paquet `MASS` [31] tous les individus dont on connaît le sexe et l'âge :

```

library(MASS)
data(survey)
scc <- survey[!is.na(survey$Sex) & !is.na(survey$Age), c("Sex", "Age")]
myplot(scc$Age, main = "Age de 236 personnes", xlab = "Taille [cm]")
shapiro.test(scc$Age)
Shapiro-Wilk normality test
data: scc$Age
W = 0.45562, p-value < 2.2e-16

```

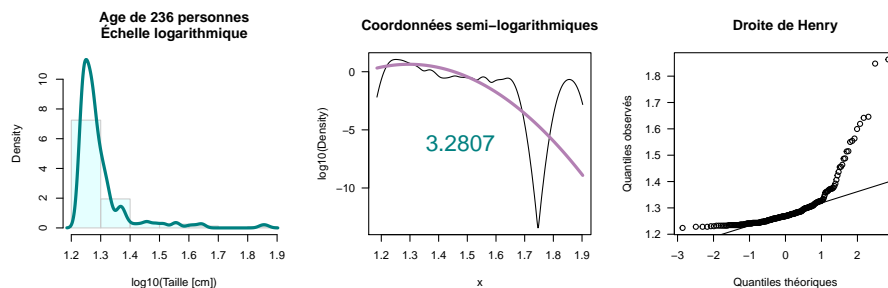


Représentez l'effet d'une transformation logarithmique et testez si l'on peut considérer les données redressées comme suivant une loi normale :

```

Shapiro-Wilk normality test
data: x
W = 0.62864, p-value < 2.2e-16

```



Voici donc encore un exemple où la transformation logarithmique échoue complètement.

4 La transformation de Box et Cox

C'est une transformation qui a été proposée par Box et Cox [3] :

$$\{y \in \mathbb{R}_+^*, \lambda \in \mathbb{R}\} : y^* = f(y, \lambda) = \begin{cases} \frac{y^\lambda - 1}{\lambda} & (\lambda \neq 0) \\ \log y & (\lambda = 0) \end{cases} \quad (1)$$

On retrouve donc dans le cas $\lambda = 0$ notre transformation logarithmique précédente. Notez que dans le cas où $\lambda = 1$ cela revient à ne pas faire de transformation et donc conserver la variable d'origine à une translation près. Pour un échantillon de n observations (y_1, y_2, \dots, y_n) , la procédure consiste à appliquer la *même* transformation à toutes les valeurs :

$$\{y_i \in \mathbb{R}_+^*, \lambda \in \mathbb{R}\} : y_i^* = f(y_i, \lambda) = \begin{cases} \frac{y_i^\lambda - 1}{\lambda} & (\lambda \neq 0) \\ \log y_i & (\lambda = 0) \end{cases} \quad (i = 1, 2, \dots, n) \quad (2)$$

La transformation de BOX et COX étant toujours monotone, l'ordre initial des données est conservé après transformation. Les tests statistiques basés sur les rangs ne seront donc pas affectés par cette transformation.

La transformation (1) n'est pas définie pour les valeurs négatives de la variable. Dans ce cas de figure, BOX et COX proposent [3] de translater toutes les valeurs :

$$\{y \in \mathbb{R}_+^*, \lambda_1 \in \mathbb{R}, \lambda_2 \in \mathbb{R}\} : y^* = f(y, \lambda_1, \lambda_2) = \begin{cases} \frac{(y + \lambda_2)^{\lambda_1} - 1}{\lambda_1} & (\lambda_1 \neq 0) \\ \log(y + \lambda_2) & (\lambda_1 = 0) \end{cases} \quad (3)$$

Deux stratégies peuvent être utilisées pour fixer λ_2 . On peut ajouter à toutes les valeurs la valeur absolue de la plus petite valeur plus un ϵ_{\min} pour que la transformation soit toujours définie. C'est par exemple ce qui est fait par la fonction `boxcoxR()` du paquet `fifer` [15] avec une valeur par défaut de $\epsilon_{\min} = 0,01$. On peut aussi comme dans l'article de BOX et COX [3] estimer simultanément λ_1 et λ_2 . Cette approche est implémentée dans la fonction `boxcoxfit()` du paquet `geoR` [27].

Pour augmenter la précision du calcul de la transformation de BOX et COX, on utilise en pratique une version mise à l'échelle, y^g , en divisant toutes les valeurs par la moyenne géométrique, \hat{y} :

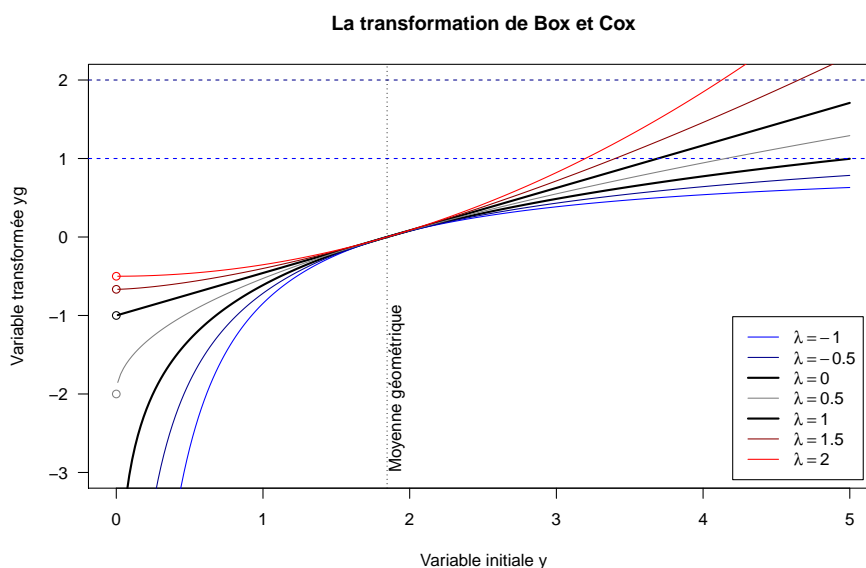
$$\{y \in \mathbb{R}_+^*, \lambda_1 \in \mathbb{R}\} : y^g = g(y, \lambda) = \begin{cases} \frac{(y/\hat{y})^\lambda - 1}{\lambda} & (\lambda \neq 0) \\ \log y - \log \hat{y} & (\lambda = 0) \end{cases} \quad \text{où } \hat{y} = e^{\frac{1}{n} \sum_{i=1}^n \log y_i} \quad (4)$$

Ceci permet d'avoir une moyenne géométrique de 1 pour y/\hat{y} et donc de concentrer les valeurs dans la région où la transformation de BOX et COX se comporte comme $y = x - 1$ puisque quelle que soit la valeur de λ on a :

$$\lim_{x \rightarrow 1} \frac{df}{dx} = 1$$

Représentons graphiquement cette fonction pour quelques valeurs du paramètre λ :

```
fboxcox <- function(y, lambda){
  stopifnot(all(y > 0))
  y <- y/exp(mean(log(y))) # Mise à l'échelle
  if(isTRUE(all.equal(lambda, 0))){ # On peut faire mieux, cf. MASS
    log(y)
  } else {
    (y^lambda - 1)/lambda
  }
}
y <- seq(from = 0.01, to = 5, le = 255)
plot(y, fboxcox(y, 1), type = "l", ylim = c(-3, 2), las = 1,
     main = "La transformation de Box et Cox", lwd = 2,
     xlab = "Variable initiale y", ylab = "Variable transformée yg") ; points(0, -1)
lines(y, fboxcox(y, 0), lwd = 2)
lines(y, fboxcox(y, -1), col = "blue") ; abline(h = 1, col = "blue", lty = 2)
lines(y, fboxcox(y, -0.5), col = "darkblue") ; abline(h = 2, col = "darkblue", lty = 2)
lines(y, fboxcox(y, 0.5), col = grey(0.5)) ; points(0, -2, col = grey(0.5))
lines(y, fboxcox(y, 1.5), col = "darkred") ; points(0, -1/1.5, col = "darkred")
lines(y, fboxcox(y, 2), col = "red") ; points(0, -0.5, col = "red")
abline(v = ydot <- exp(mean(log(y))), lty = 3)
text(x = ydot, y = -3, "Moyenne géométrique", srt = 90, pos = 4)
legend("bottomright", inset = 0.02, legend = c(expression(lambda == -1),
expression(lambda == -0.5), expression(lambda == 0), expression(lambda == 0.5),
expression(lambda == 1), expression(lambda == 1.5), expression(lambda == 2)),
      col = c("blue", "darkblue", "black", "grey(0.5)", "black", "darkred", "red"), lty = 1,
      lwd = c(1, 1, 2, 1, 2, 1, 1))
```




Le paramètre λ apporte donc une souplesse par rapport à la transformation logarithmique. Les valeurs négatives de λ conduisent à plus compresser les fortes valeurs que la transformation logarithmique et *a contrario* les valeurs positives à être moins stringente que la fonction logarithmique. On peut pressentir un petit souci ici. Pour les valeurs de λ strictement positives les valeurs de la variable transformée ne peuvent être inférieures à $-1/\lambda$ (mis en évidence par un point dans le graphique). Pour les valeurs de λ strictement négatives les valeurs de la variable transformée ne peuvent être supérieures à $-1/\lambda$ (mis en évidence par l'asymptote horizontale en pointillé sur le graphique). C'est un peu embêtant pour une variable normale supposée pouvoir gambader librement entre $-\infty$ et

$+\infty$. On jettera un voile pudique sur ce problème⁸. Notez que la transformation logarithmique ($\lambda = 0$) ne souffre pas de ce problème, c'est un avantage théorique indéniable pour la favoriser si on n'a pas de forte indication de son inadéquation. Un autre avantage, plus pratique, est que l'échelle logarithmique est familière dans le domaine scientifique.

5 Estimation de λ basée sur les observations originales

Fonction	Paquet	Référence	Remarques
<code>boxcox()</code>	MASS	[31]	<i>recommended by the R core team</i>
<code>BoxCoxTrans()</code>	caret	[20]	Appelle <code>MASS::boxcox()</code>
<code>boxcoxR()</code>	fifer	[15]	Utilise <code>MASS::boxcox()</code>
<code>boxCox()</code>	car	[16]	Généralisation de <code>MASS::boxcox()</code>
<code>boxcoxfit()</code>	geoR	[27]	Estime aussi λ_2
<code>boxcoxnc()</code>	AID	[7, 8, 9]	Utilise 7 tests de normalité
<code>BoxCox()</code>	FitAR	[23]	Méthodes pour les séries temporelles
<code>BoxCoxLambda()</code>	DescTools	[30]	Implémente la méthode de GUERRERO [17]
<code>find_lambda()</code>	rust	[26]	
<code>boxcox()</code>	EnvStats	[25]	Fonction très bien documentée
<code>modelBoxCox()</code>	lmSupport	[6]	Appelle <code>car::boxCox()</code>

TABLE 1 – Exemples de fonctions  permettant d'estimer le paramètre λ de la transformation de Box et Cox [3].

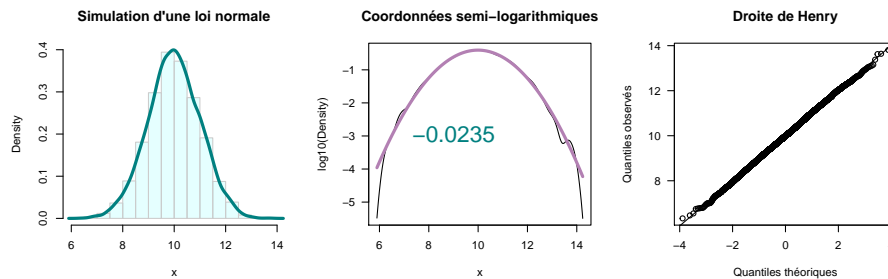
Il existe de nombreux paquets  qui proposent des fonctions pour estimer le paramètre λ de la transformation de Box et Cox (*cf.* table 1). Nous allons utiliser ici la fonction `boxcox()` du paquet MASS [31] par simplicité et parce que ce dernier est « *recommended by the R core team* (**r-recommended**) ». Le critère utilisé pour estimer λ est comme dans l'article de Box et Cox [3] celui du maximum de vraisemblance. Ce n'est pas le seul critère utilisable. Par exemple dans la fonction `boxcox()` du paquet EnvStats [25] permet en plus d'utiliser un critère basé sur le coefficient de corrélation linéaire dans le graphe quantile-quantile de la droite de HENRY ou bien sur la statistique du test de normalité de SHAPIRO et WILK [29]. La fonction `boxcoxnc()` du paquet AID [7, 8, 9] permet d'utiliser pas moins de 7 critères différents. On se contentera ici d'utiliser le critère du maximum de vraisemblance de la fonction `boxcox()` du paquet MASS [31].

5.1 Données simulées

Commençons par un jeu de données simulé en tirant dans une loi normale. On s'attend à trouver $\lambda = 1$ puisqu'aucune transformation ne s'impose dans ce cas de figure.

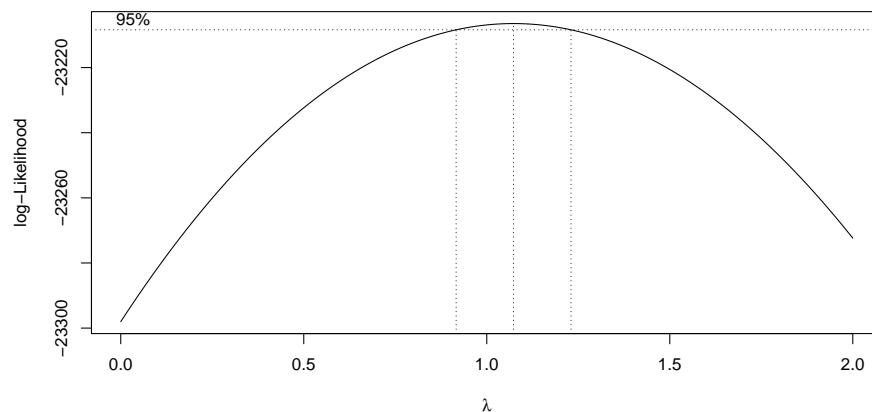
⁸. Une autre solution est d'appliquer le très puissant « algorithme de l'autruche » pour reprendre une expression de la traduction française de [10] disponible à <http://russell.vcharite.univ-mrs.fr/EIE/fchap14.pdf>


```
set.seed(1)
x <- rnorm(10000, mean = 10)
myplot(x, main = "Simulation d'une loi normale")
```



Estimons la valeur du paramètre λ :

```
res <- boxcox(x~1, lambda = seq(0, 2, length = 1000))
res$x[which.max(res$y)]
[1] 1.073073
```

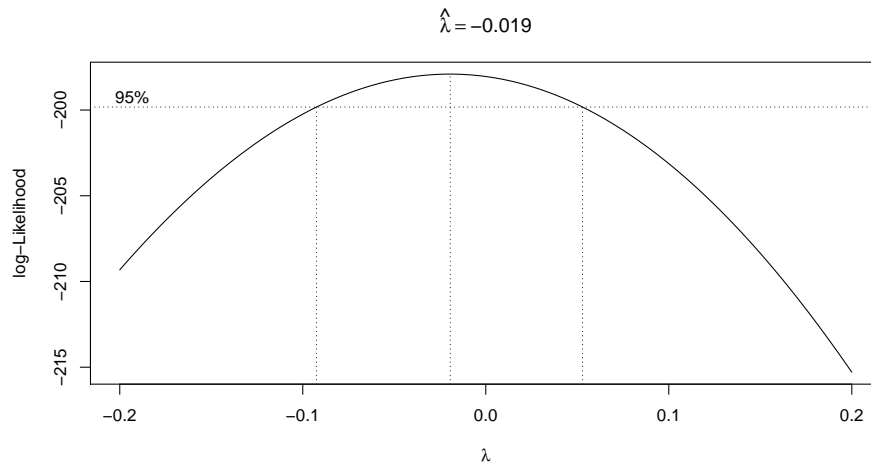


Donc effectivement, on retrouve bien une valeur de λ proche de 1 indiquant qu'il n'y a pas lieu de faire de transformation ici.

5.2 Masse de 62 mammifères

On repart maintenant de l'exemple sur la masse corporelle de 62 mammifères terrestres dont on a vu précédemment que la transformation logarithmique donnait de bon résultats. On s'attend donc à trouver un λ proche de zéro.

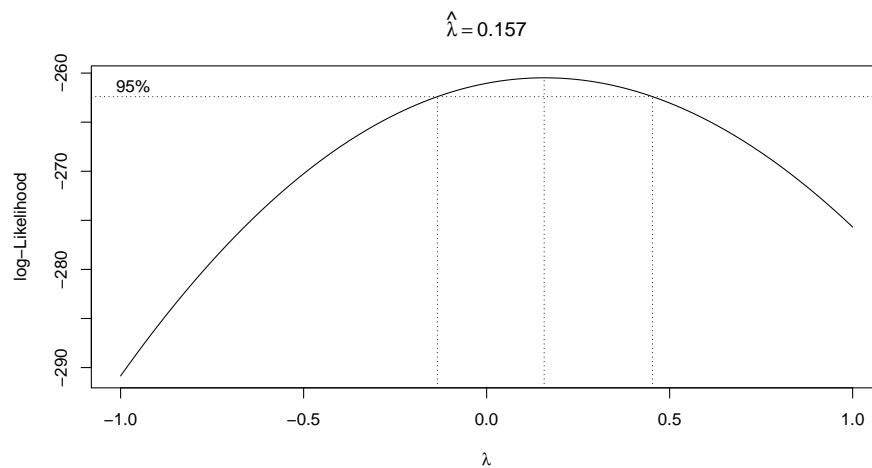
```
x <- mammals$body
res <- boxcox(x~1, plotit = TRUE, lambda = seq(-0.2, 0.2, length = 1000))
(lest <- res$x[which.max(res$y)])
[1] -0.01941942
title(main = bquote(hat(lambda) == .(round(lest,3))))
shapiro.test(fboxcox(x, lest))
      Shapiro-Wilk normality test
data:  fboxcox(x, lest)
W = 0.98796, p-value = 0.805
```



On a donc $\lambda \approx -0.02$ et la valeur de 0 est comprise dans l'intervalle de confiance pour la valeur de λ . On retrouve ici le fait que la transformation logarithmique est tout à fait appropriée pour normaliser les données. La transformation de Box et Cox améliore les chose d'un pouième : on est passé d'une valeur p de 0,7272 à 0,805 et l'intervalle de confiance pour λ nous indique bien que l'on ne peut pas rejeter la transformation logarithmique comme optimale. Donc autant faire simple.

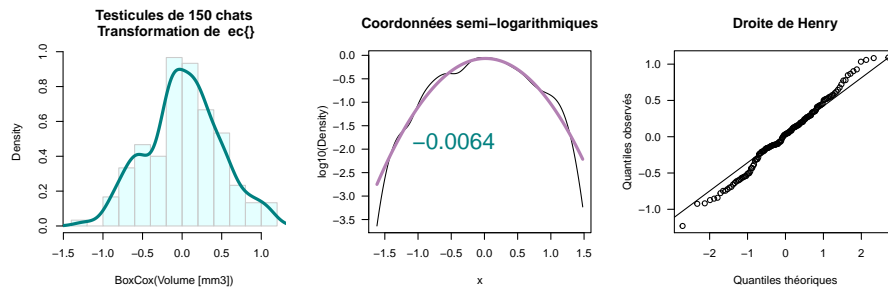
5.3 Volume des testicules de 150 chats

Reprenez le jeu de données `chasay` et estimez le paramètre λ de la distribution de Box et Cox :



Étudiez et testez la normalité de la variable après transformation de Box et Cox :

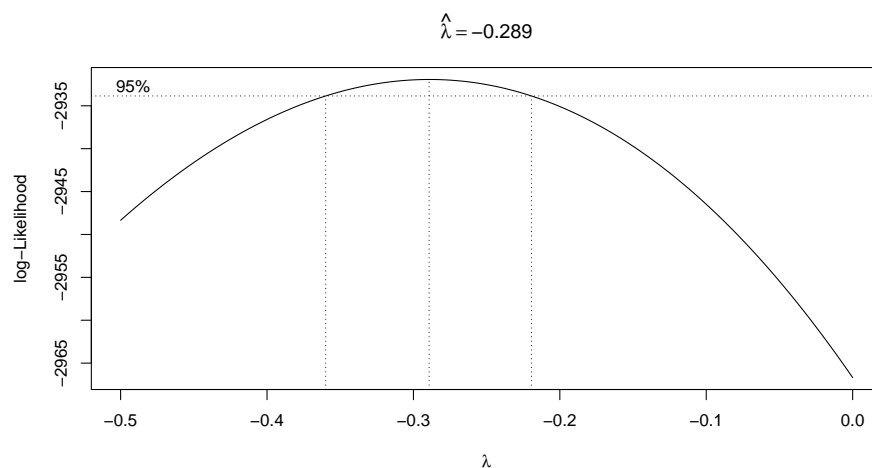
```
Shapiro-Wilk normality test
data: xbc
W = 0.99214, p-value = 0.5805
```



Quelle transformation suggèreriez-vous d'utiliser dans ce cas de figure ?

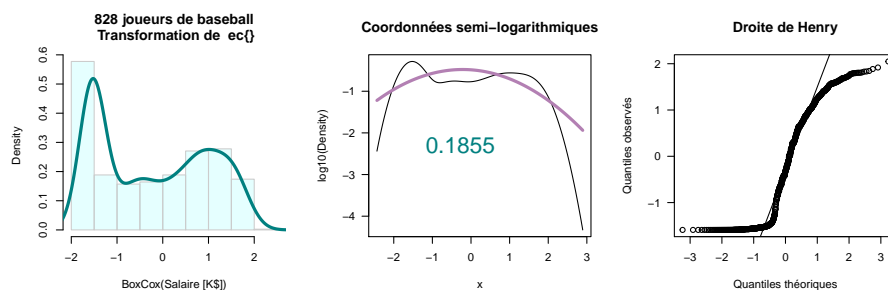
5.4 Salaire de 828 joueurs de baseball

C'est un jeu de données où la transformation logarithmique n'avait pas réussi à normaliser les données. La transformation de BOX et COX pourra-t-elle y arriver ? Reprenez le jeu de données MLB et estimez le paramètre λ :



Étudiez et testez la normalité de la variable après transformation de BOX et COX :

```
Shapiro-Wilk normality test
data: xbc
W = 0.86925, p-value < 2.2e-16
```

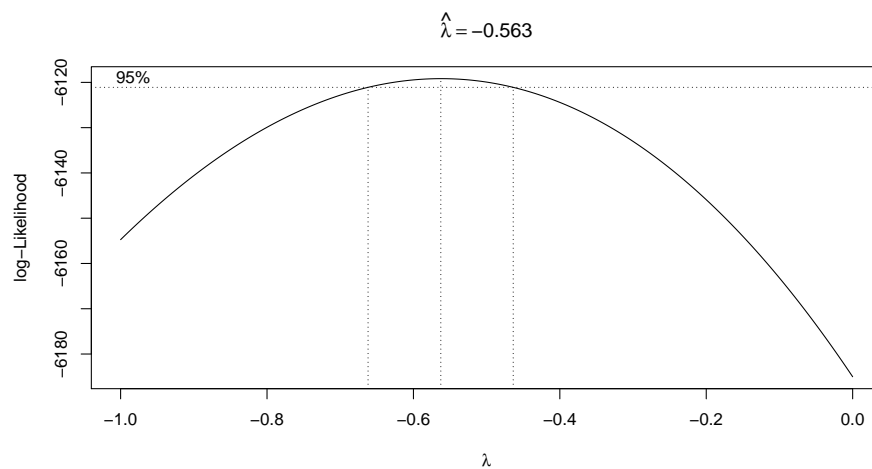


Si on s'en tient à la valeur p du test de normalité on pourrait dire que la transformation de BOX et COX n'apporte rien ici puisque l'on reste dans des

valeurs epsilonlesques. Mais on pourrait en disconvenir courtoisement en faisant remarquer que la transformation de BOX et COX met mieux en évidence le caractère multimodal de la distribution.

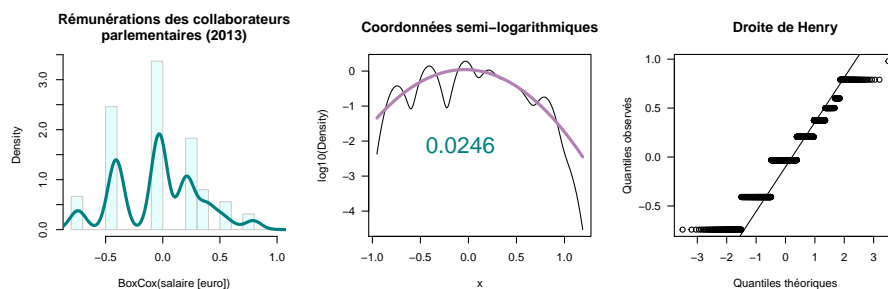
5.5 Salaire de (2165 + 1) contrats de collaborateurs parlementaires

On avait vu pour ce jeu de données (*cf.* figure 1 page 12) que si la transformation logarithmique améliorait les choses, on était encore loin d'une distribution symétrique. Estimez le paramètre λ de la transformation de Box et Cox :



C'est un cas intéressant puisque l'on a un λ clairement différent de zéro, et donc de la transformation logarithmique habituelle. Représentez les données normalisées :

```
Shapiro-Wilk normality test
data:  xbc
W = 0.93637, p-value < 2.2e-16
```



C'est un cas de figure où la transformation de BOX et COX est utile puisqu'elle nous permet de mieux symétriser les données que la transformation logarithmique. Le test de normalité n'est toujours pas bon (avec des données discrétisées, rien d'étonnant) mais on peut maintenant mieux répondre à la question « ce salaire est-il anormalement élevé ? »

```
1 - pnorm(xbc[length(xbc)], mean(xbc), sd(xbc))
[1] 0.002499631
```

On prend effectivement très peu de risque à affirmer, statistiquement parlant, que le salaire de celle qui déclarait le 18 mai 2007 à Kim WILLISHER du *Daily Telegraph* : « *but I've never been actually his assistant or anything like that. I don't deal with his communication* » était anormalement élevé. Avec le test du χ^2 pour les valeurs extrêmes [13] implémenté dans la fonction `chisq.out.test()` du paquet `outliers` [19] on peut facilement tester cette hypothèse :

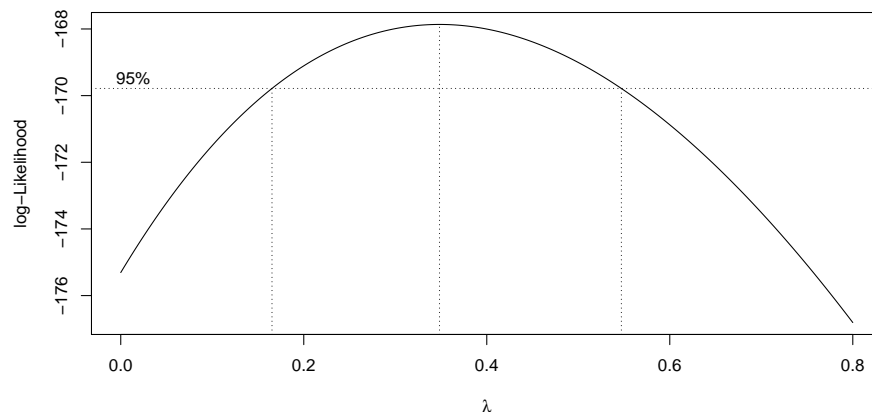
```
xbc[length(xbc)] # PFI
[1] 0.9792501
library(outliers)
chisq.out.test(xbc)
      chi-squared test for outlier
data:  xbc
X-squared = 7.8797, p-value = 0.004999
alternative hypothesis: highest value 0.979250115444296 is an outlier
```

Avec un risque de première espèce de 5 %, les données expérimentales sont en contradiction avec l'hypothèse nulle que le salaire de Penelope FILLON ne soit pas extrême. On rejette donc l'hypothèse nulle pour son alternative : ce salaire est extrême (*outlier*). Quant à l'a-moralité, c'est une notion éminemment subjective, il n'y a pas de test statistique pour cela, libre à chacun de se forger sa propre opinion.

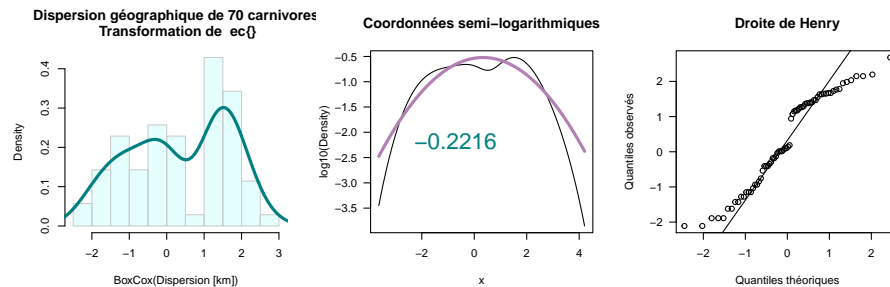
5.6 Dispersion de 70 carnivores

Estimez la valeur du paramètre λ pour le jeu de données des 70 carnivores :

```
[1] 0.3483483
```



Représentez les données redressées :



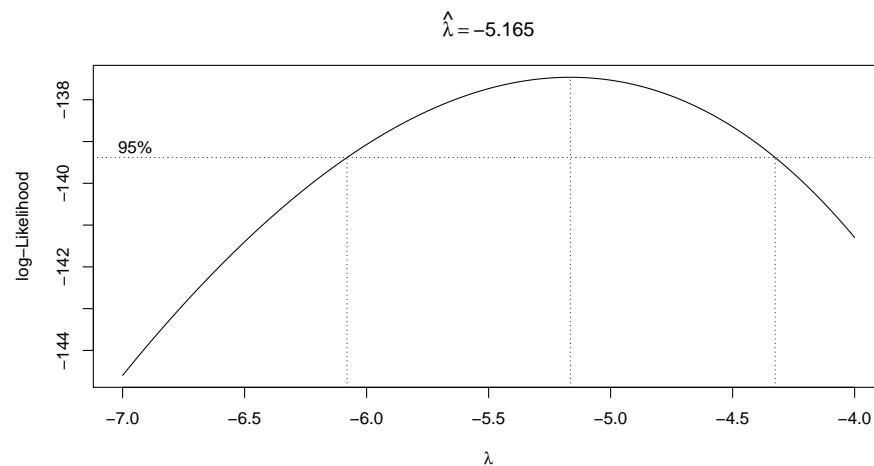
Testez la normalité :

```
Shapiro-Wilk normality test
data: fboxcox(x, lest)
W = 0.93205, p-value = 0.0009301
```

On a amélioré les choses par rapport à la transformation logarithmique mais on ne peut toujours pas considérer que la distribution soit normale. La distribution est clairement bimodale et aucune transformation simple ne pourra effacer cet aspect.

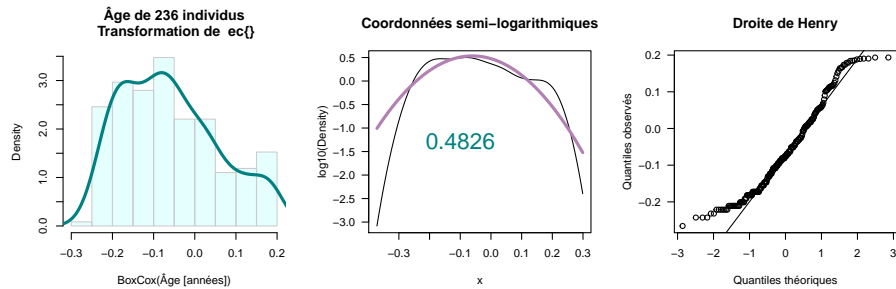
5.7 Âge de 236 individus

C'est encore un jeu de données où la transformation logarithmique n'avait pas réussi à normaliser les données. La transformation de BOX et COX pourra-t-elle y arriver ? Reprenez le jeu de données `scc` et estimez le paramètre λ :



Étudiez et testez la normalité de la variable après transformation de BOX et COX :

```
Shapiro-Wilk normality test
data: xbc
W = 0.95512, p-value = 1.034e-06
```



Quelle transformation suggèreriez-vous d'utiliser dans ce cas de figure ?

6 Estimation de λ dans le cadre d'un modèle linéaire

Dans le cas d'un modèle linéaire standard portant sur n observations et comportant p prédicteurs on a :

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip} + \epsilon_i \quad (5)$$

Par exemple, dans la section sur la transformation de BOX et COX sur les données originelles, on a utilisé implicitement un modèle sans prédicteurs qui consiste simplement à modéliser les données par leur moyenne plus un terme d'erreur gaussien de moyenne nulle ϵ_i :

$$y_i = \beta_0 + \epsilon_i \quad (6)$$

La fonction objective utilisée pour estimer λ se base sur ces résidus ϵ_i pour caractériser la normalité de la distribution. Pour un modèle linéaire un peu plus général, on utilise les variables de réponse modifiées y_i^g par la transformation de BOX et COX (4) au lieu des variables originelles y_i dans le modèle :

$$y_i^g = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip} + \epsilon_i \quad (7)$$

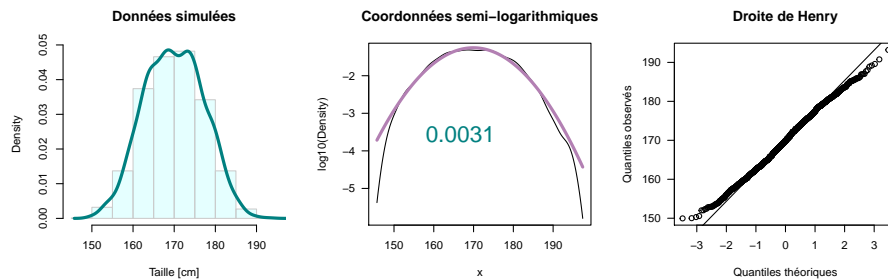
La fonction objective utilisée pour estimer λ se base sur ces résidus ϵ_i pour caractériser la normalité de la distribution. C'est donc bien à la normalité de ces résidus, et non celle de la variable transformée, qu'il faudra s'intéresser.

6.1 Données simulées

On simule un jeu de données pour la taille de 1000 individus mâles et de 1000 individus femelles distribuée normalement au sein de chaque groupe.

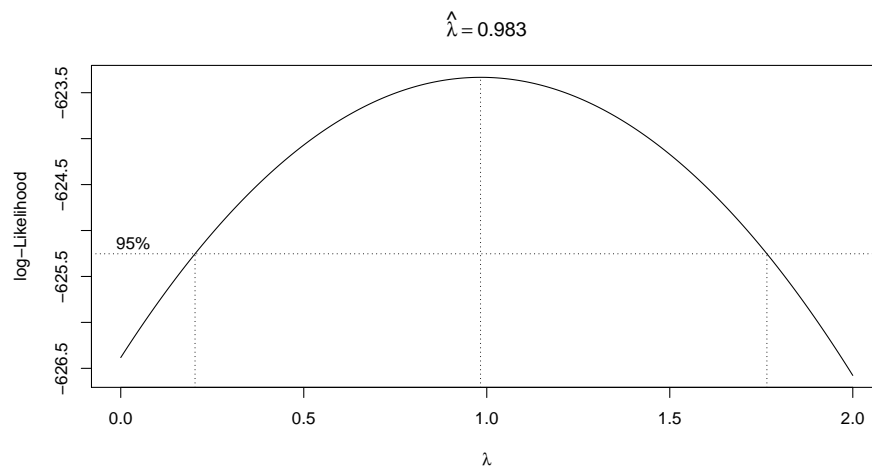
```
set.seed(1)
x <- c(rnorm(1000, 165, 5), rnorm(1000, 175, 5))
sex <- gl(2, 1000, labels = c("F", "M"))
myplot(x, main = "Données simulées",
       xlab = "Taille [cm]")
shapiro.test(xbc)
```

```
Shapiro-Wilk normality test
data:  xbc
W = 0.95512, p-value = 1.034e-06
```



On s'attend donc à avoir un λ proche de 1 puisqu'aucune transformation n'est nécessaire dans ce cas idéal.

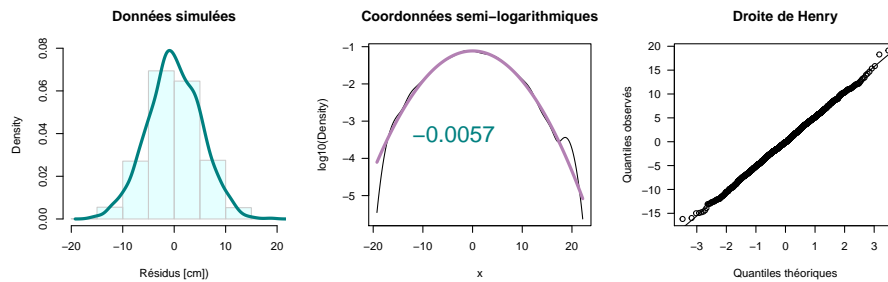
```
rbc <- boxcox(x~sex, lambda = seq(0, 2, le=1000))
lest <- rbc$x[which.max(rbc$y)]
title(main = bquote(hat(lambda) == .(round(lest,3))))
```



On décide donc de ne pas transformer les données et on étudie la distribution des résidus :

```
xbc <- resid(lm(x~sex))
myplot(xbc, main = "Données simulées",
       xlab = "Résidus [cm]")
shapiro.test(xbc)

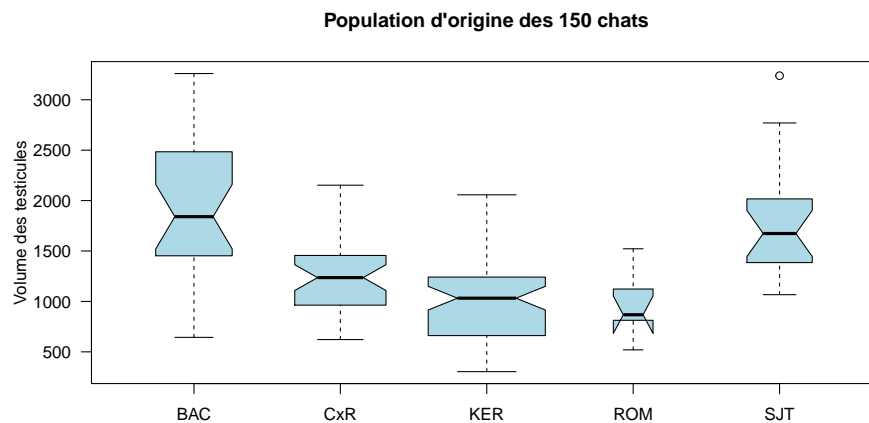
Shapiro-Wilk normality test
data:  xbc
W = 0.99941, p-value = 0.8175
```

6.2 Volume des testicules de 150 chats

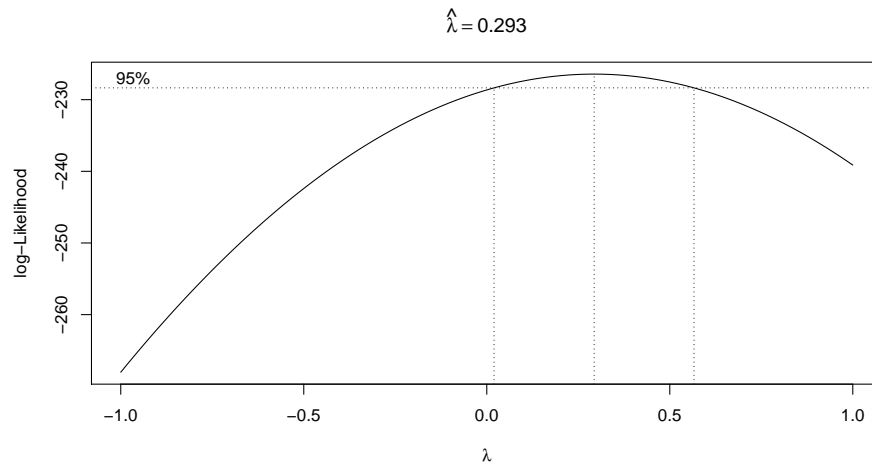
On dispose dans le jeu de données `chasay` de la population d'origine (`pop`) des chats.

```
boxplot(chasay$vol~chasay$pop, varwidth = T, las = 1, notch = T, col = "lightblue",
        main = "Population d'origine des 150 chats", ylab = "Volume des testicules")
```



Il y a un effet de la population d'origine, il serait idiot de ne pas en tenir compte.

```
rbc <- boxcox(vol~pop, data = chasay, lambda = seq(-1, 1, length = 1000))
lest <- rbc$x[which.max(rbc$y)]
title(main = bquote(hat(lambda) == .(round(lest,3))))
```

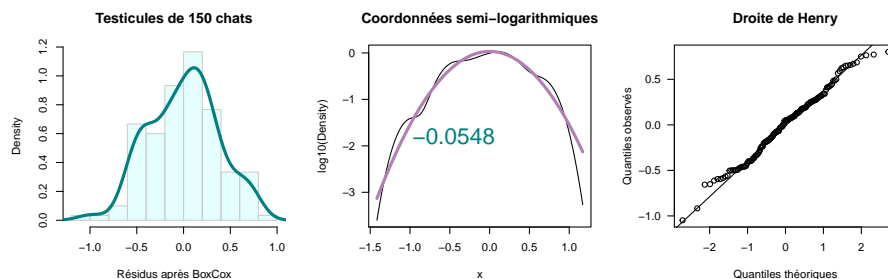


La transformation optimale est intermédiaire entre la transformation logarithmique et l'absence de transformation. Étudions la distribution des résidus après la transformation de Box et Cox.

```

xbc <- resid(lm(fboxcox(vol, lest)-pop, data = chasay))
myplot(xbc, main = "Testicules de 150 chats",
       xlab = "Résidus après BoxCox")
shapiro.test(xbc)
      Shapiro-Wilk normality test
data:  xbc
W = 0.98894, p-value = 0.284

```

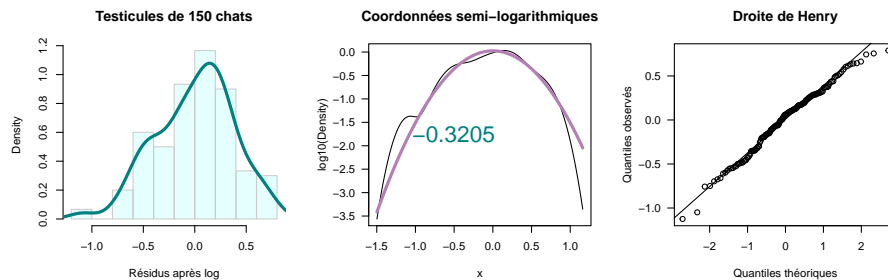


Ce n'est pas mal du tout, mais le gain est-il vraiment important par rapport à une simple transformation logarithmique ?

```

xlog <- resid(lm(log(vol)-pop, data = chasay))
myplot(xlog, main = "Testicules de 150 chats",
       xlab = "Résidus après log")
shapiro.test(xlog)
      Shapiro-Wilk normality test
data:  xlog
W = 0.98631, p-value = 0.1457

```

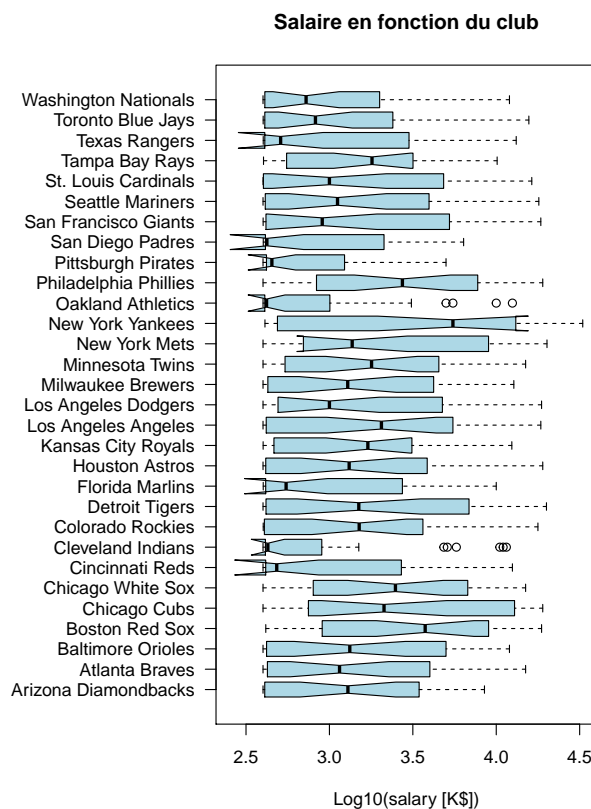


Il n'est pas certain que le jeu en vaille la chandelle ici.

6.3 Salaire de 828 joueurs de baseball

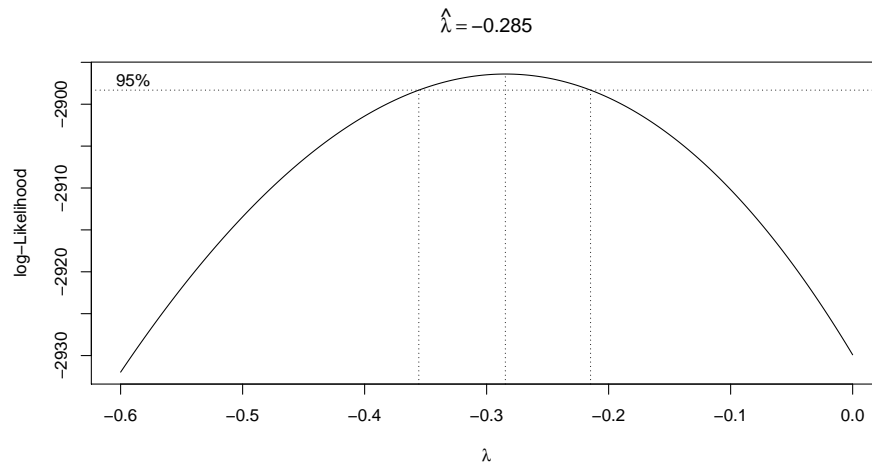
On dispose dans le jeu de données MLB du club d'origine des joueurs. Il est de notoriété publique que les clubs sportifs sont plus ou moins riches et que cela risque d'influencer le salaire des joueurs. Voyons si c'est le cas ici.

```
par(mar = c(5, 10, 4, 2) + 0.1)
boxplot(log10(salary)~team, data = MLB, horizontal = TRUE, las = 1, varwidth = T,
  col = "lightblue", notch = T, xlab = "Log10(salary [K$])",
  main = "Salaire en fonction du club")
```



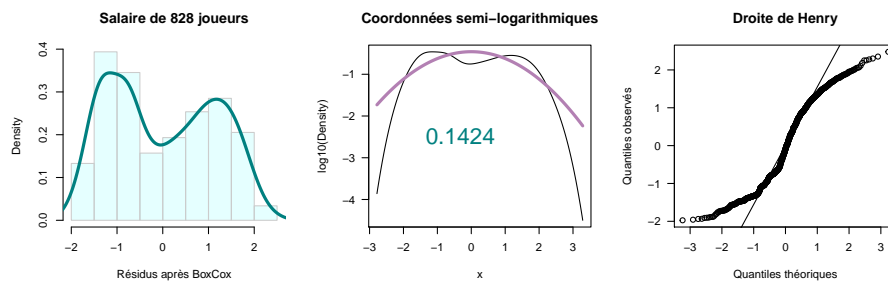
Il y bien un fort effet du club d'origine. On voit par exemple que le salaire médian des « *Oakland Athletics* » est un ordre de grandeur inférieur à celui

des « *New York Yankees* ». On note deux clubs qui pratiquent une politique salariale particulièrement inégalitaire puisqu'ils arrivent à avoir des *outliers* avec une échelle logarithmique. Estimez le paramètre λ de la distribution de Box et Cox :



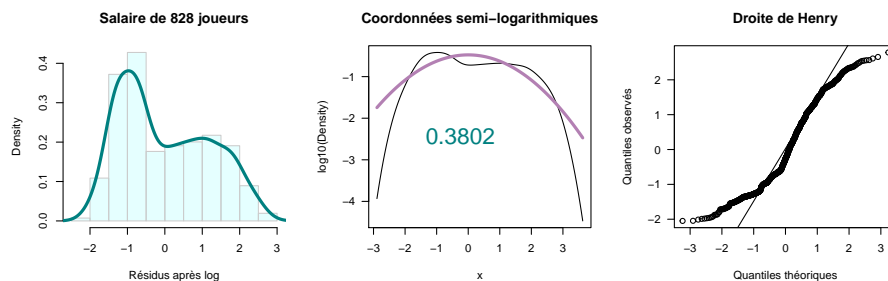
Étudiez la distribution des résidus après la transformation de Box et Cox.

```
Shapiro-Wilk normality test
data: xbc
W = 0.93868, p-value < 2.2e-16
```



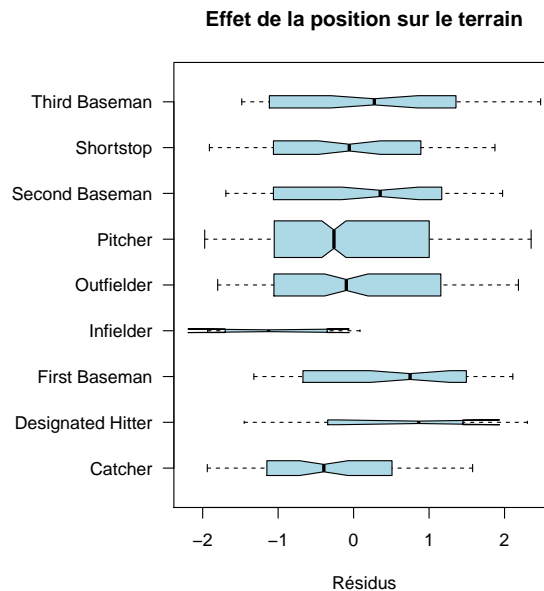
Comparez avec la transformation logarithmique :

```
Shapiro-Wilk normality test
data: xlog
W = 0.94227, p-value < 2.2e-16
```



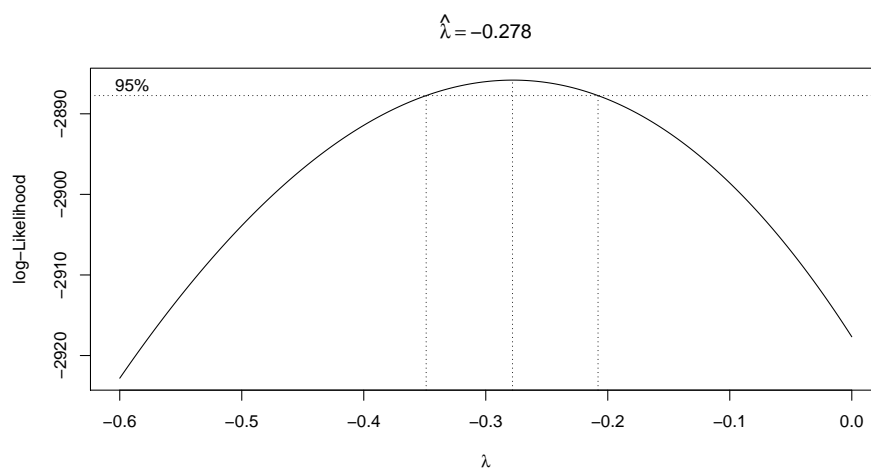
La transformation de Box et Cox permet ici de mieux mettre en évidence le caractère bimodal de la distribution. C'est le signe qu'il nous manque une

variable explicative. Le jeu de données MLB comporte également la position des joueurs sur le terrain. Serait-ce notre variable explicative manquante ? Représentez la distribution des résidus après la transformation de Box et Cox et prise en compte de l'effet club en fonction de la position sur le terrain.



Estimez la valeur du paramètre λ quand on tient compte simultanément de ces deux facteurs explicatifs. On commence par un modèle simple sans interaction :

```
rbc <- boxcox(salary~team+position, data = MLB, lambda = seq(-0.6, 0, le=1000))
lest <- rbc$x[which.max(rbc$y)]
title(main = bquote(hat(lambda) == .(round(lest,3))))
```



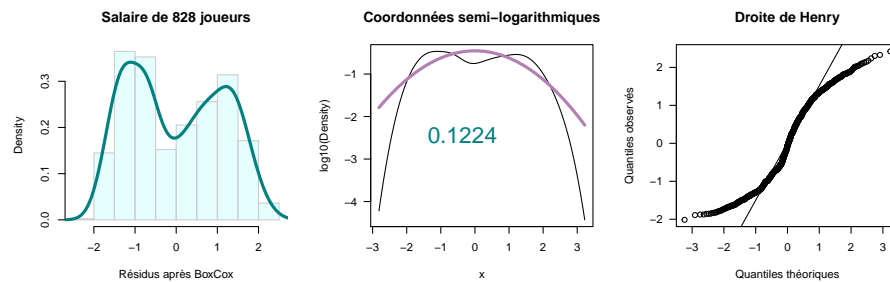
Comment se comportent les résidus ?

```

xbc <- resid(lm(fboxcox(salary, lest)~team+position, data = MLB))
myplot(xbc, main = "Salaire de 828 joueurs", xlab = "Résidus après BoxCox")
shapiro.test(xbc)

Shapiro-Wilk normality test
data:  xbc
W = 0.94121, p-value < 2.2e-16

```

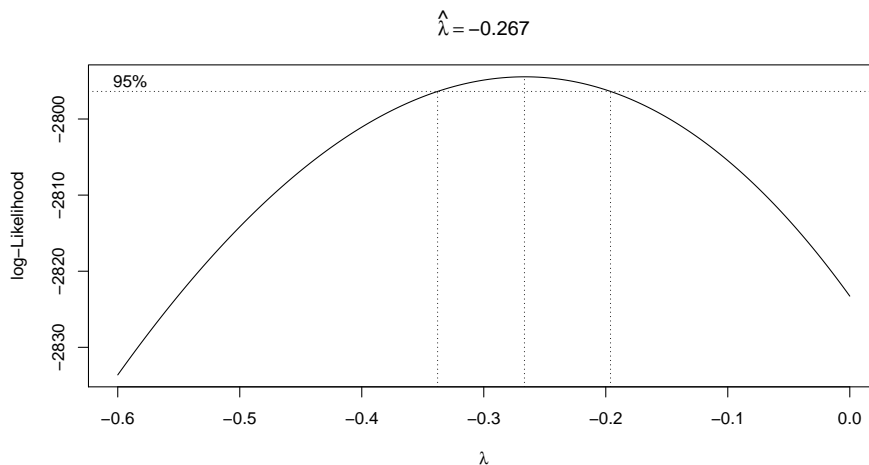


On a quasiment rien gagné ici. Il y a interaction entre deux facteurs sur une variable lorsque l'effet d'un facteur sur la variable est modifié par la valeur de l'autre facteur. Il n'est pas impensable d'imaginer que ce soit le cas ici, la politique salariale est peut-être différente d'un club à l'autre, par exemple un club peut décider d'investir plus sur l'attaque et un autre plus sur la défense. Essayons donc un modèle avec interaction :

```

rbc <- boxcox(salary~team*position, data = MLB, lambda = seq(-0.6, 0, le=1000))
lest <- rbc$x[which.max(rbc$y)]
title(main = bquote(hat(lambda) == .(round(lest,3))))

```



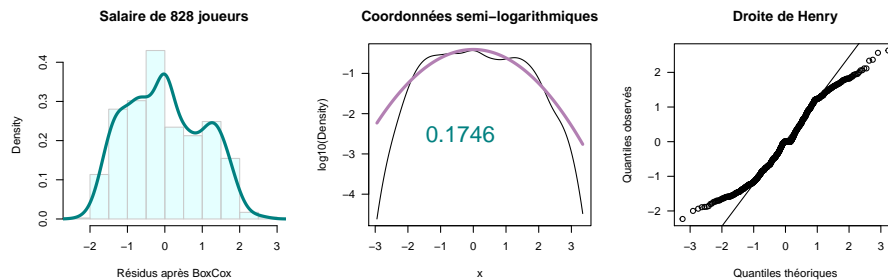
Comment se comportent les résidus ?

```

xbc <- resid(lm(fboxcox(salary, lest)~team*position, data = MLB))
myplot(xbc, main = "Salaire de 828 joueurs", xlab = "Résidus après BoxCox")
shapiro.test(xbc)

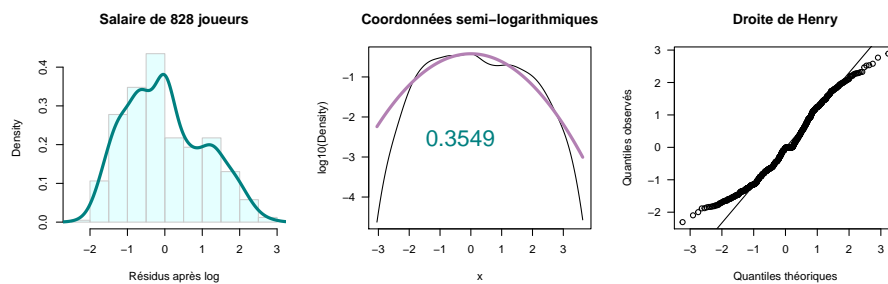
Shapiro-Wilk normality test
data:  xbc
W = 0.97347, p-value = 3.993e-11

```



On a atténué le caractère bimodal de la distribution des résidus mais on est encore loin de la normalité. Il doit nous manquer un facteur explicatif. Comparons avec la transformation logarithmique :

```
xlog <- resid(lm(log(salary)~team*position, data = MLB))
myplot(xlog, main = "Salaire de 828 joueurs", xlab = "Résidus après log")
shapiro.test(xlog)
# Shapiro-Wilk normality test
# data:  xlog
# W = 0.97419, p-value = 6.27e-11
```

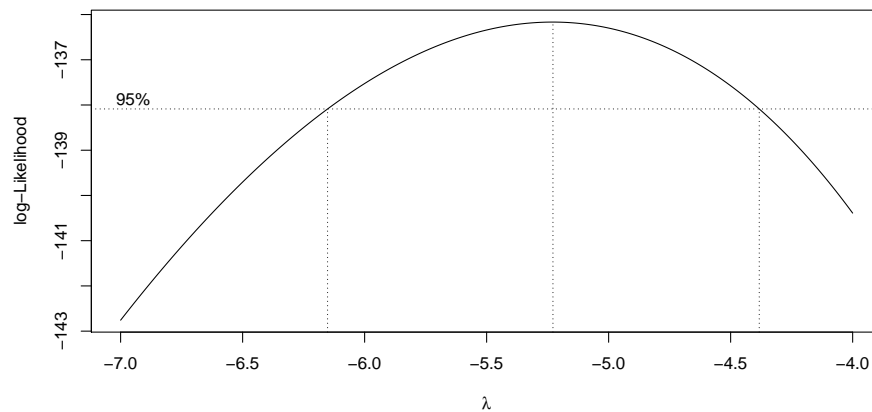


La différence n'est pas flagrante.

6.4 Âge de 236 individus

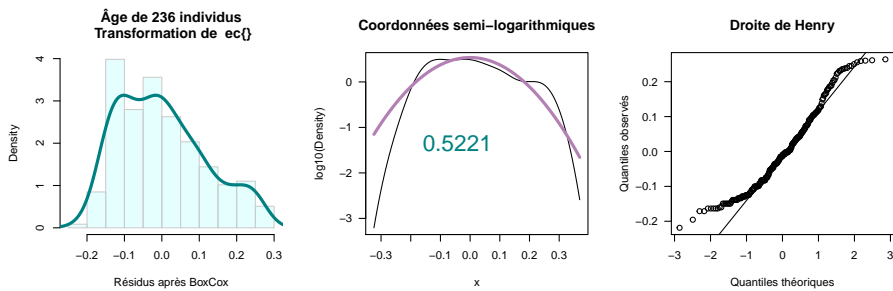
Reprenez le jeu de données `scc` et estimez le paramètre λ en tenant compte du sexe des individus :

$$\hat{\lambda} = -5.228$$



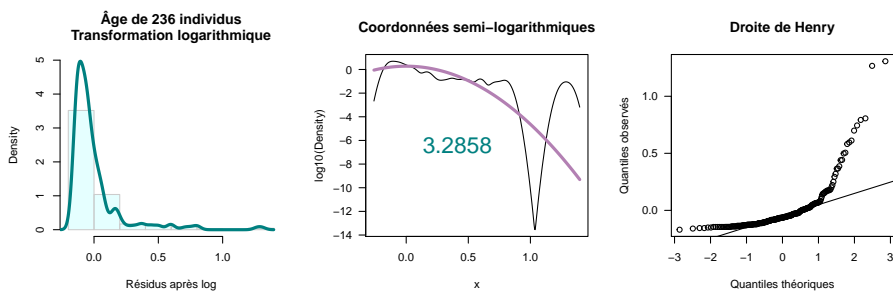
Étudiez et testez la normalité de la variable après transformation de Box et Cox :

```
Shapiro-Wilk normality test
data:  xbc
W = 0.95499, p-value = 9.988e-07
```



Comparez avec la transformation logarithmique :

```
Shapiro-Wilk normality test
data:  xlog
W = 0.62698, p-value < 2.2e-16
```



7 La fonction boxcox() du paquet MASS

Cette section détaille ce que fait exactement la fonction `boxcox()` du paquet MASS [31] et peut donc être ignorée en première lecture. L'avantage d'un logiciel libre comme R est que le code source est disponible, on se réfère ici au fichier MASS/R/boxcox.R de la version 7.3-45 du 2015-11-10. La fonction est définie ainsi :

```
boxcox <- function(object, ...) UseMethod("boxcox")
```

C'est donc une fonction générique qui va s'adapter automatiquement à la classe de l'objet qui lui est donné en argument. Les constructions du type `Age~Sex` sont de la classe `formula` :

```
with(scc, class(Age~Sex))
[1] "formula"
```

C'est donc la fonction `boxcox.formula()` qui va être invoquée dans notre cas :


```
boxcox.formula <-
function(object, lambda = seq(-2, 2, 1/10), plotit = TRUE,
      interp = (plotit && (m < 100)), eps = 1/50,
      xlab = expression(lambda), ylab = "log-Likelihood", ...)
```

La fonction commence par définir les valeurs par défaut de quelques paramètres. Le paramètre `lambda` est un vecteur donnant les valeurs de λ pour lesquelles on va calculer le logarithme de la vraisemblance. La valeur par défaut consiste à utiliser les 41 valeurs comprises entre -2 et +2 avec un pas de 0,1. C'est un intervalle centré sur $\lambda = 0$ correspondant à la transformation logarithmique. Le paramètre logique `plotit` est vrai par défaut et signifie donc de représenter graphiquement la fonction de vraisemblance. Le paramètre logique `interp` contrôle si on veut utiliser une interpolation avec des splines pour le tracé de la fonction de vraisemblance. Il sera vrai par défaut puisque `m`, la longueur du vecteur `lambda`, vaut 41 par défaut. Le paramètre `eps` contrôle la tolérance pour considérer que l'on a $\lambda \approx 0$ et vaut 0,02 par défaut. On est ici bien au dessus du « epsilon machine » :

```
.Machine$double.eps
[1] 2.220446e-16
```

On verra plus loin qu'il y a une bonne raison pour qu'il en soit ainsi. Enfin les paramètres `xlab` et `ylab` donnent les annotations des axes qui par défaut sont « λ » pour l'axe des abscisses et « log-Likelihood » pour l'axe des ordonnées. L'argument point-point-point « ... » absorbera tous les paramètres supplémentaires. Voyons maintenant le corps de la fonction :

```
{
  m <- length(lambda)
  object <- lm(object, y = TRUE, qr = TRUE, ...)
  result <- NextMethod()
  if(plotit) invisible(result)
  else result
}
```

On commence par calculer `m`, la longueur du vecteur `lambda`, puis on invoque la fonction `lm()` qui est l'acronyme de « *linear models* » pour ajuster le modèle linéaire décrit par la formule. À noter que l'on force les paramètres `y` et `qr` à être vrais alors que par défaut seul `qr` est vrai. Le paramètre `y` demande de calculer la réponse du modèle et le paramètre `qr` la décomposition QR. On s'assure donc ici de bien avoir à disposition les valeurs de la variable à expliquer :

```
with(scc, all.equal(as.vector(lm(Age~1, y = TRUE)$y), Age))
[1] TRUE
with(scc, all.equal(as.vector(lm(Age~Sex, y = TRUE)$y), Age))
[1] TRUE
```

On a maintenant un objet de la classe `lm` c'est donc la fonction `boxcox.lm()` qui va être invoquée :

```
boxcox.lm <-
function(object, lambda = seq(-2, 2, 1/10), plotit = TRUE,
      interp = (plotit && (m < 100)), eps = 1/50,
```

```

        xlab = expression(lambda), ylab = "log-Likelihood", ...)
{
  m <- length(lambda)
  if(is.null(object$y) || is.null(object$qr))
    object <- update(object, y = TRUE, qr = TRUE, ...)
  result <- NextMethod()
  if(plotit) invisible(result)
  else result
}

```

Elle s'assure de mettre à jour le calcul de l'ajustement du modèle linéaire avec `y = TRUE` pour être certain d'avoir à disposition les valeurs de la variable à expliquer. Dans notre cas c'est déjà fait mais il faut s'en assurer si `boxcox()` est invoquée avec un objet de la classe `lm` directement. Voyons maintenant la fonction par défaut qui fait le travail.

```

boxcox.default <-
function(object, lambda = seq(-2, 2, 1/10), plotit = TRUE,
        interp = (plotit && (m < 100)), eps = 1/
        50, xlab = expression(lambda), ylab = "log-Likelihood", ...)
{
  if(is.null(y <- object$y) || is.null(xqr <- object$qr))
    stop(gettextf("%s does not have both 'qr' and 'y' components",
        sQuote(deparse(substitute(object)))), domain = NA)
  if(any(y <= 0))
    stop("response variable must be positive")

```

On commence par vérifier que l'objet passé en argument possède bien les composants `y` et `qr` requis. On vérifie également que toutes les valeurs de la variable à expliquer sont strictement positives pour que la transformation de Box et Cox soit définie.

```

  n <- length(y)

```

La variable `n` contient donc le nombre total de valeurs dans le vecteur `y` de la variable à expliquer.

```

  ## scale y[] {for accuracy in y^la - 1 }:
  y <- y / exp(mean(log(y)))
  logy <- log(y) # now ydot = exp(mean(log(y))) == 1

```

On fait ici une mise à l'échelle en divisant toutes les valeurs par la moyenne géométrique des valeurs de la variable à expliquer. La transformation utilisée ici est donc la version mise à l'échelle de la transformation de Box et Cox (*cf.* équation 4). La suite du programme contient la boucle centrale calculant le logarithme de la fonction de vraisemblance :

```

  xl <- loglik <- as.vector(lambda)
  m <- length(xl)
  for(i in 1L:m) {
    if(abs(la <- xl[i]) > eps)
      yt <- (y^la - 1)/la
    else yt <- logy * (1 + (la * logy)/2 *
        (1 + (la * logy)/3 * (1 + (la * logy)/4)))
    loglik[i] <- - n/2 * log(sum(qr.resid(xqr, yt)^2))
  }

```

Le vecteur `x1` qui contient les valeurs explorées de λ et le vecteur `loglik` qui contient la valeur du logarithme de la fonction de vraisemblance sont initialisés avec le vecteur `lambda`. Les valeurs de `x1` ne seront pas changées mais celles de `loglik` oui. La variable `m` est le nombre total de valeurs explorées pour λ . On teste ensuite la valeur courante de λ par rapport à la variable `eps` qui vaut 0.02 par défaut. Si $|\lambda| > \epsilon$ on n'utilise pas la transformation logarithmique. Le vecteur `yt` contient les valeurs après transformation. Dans le cas où $|\lambda| > \epsilon$, on retrouve bien la transformation de BOX et COX mais dans le cas où $|\lambda| \leq \epsilon$ où on s'attendait à simplement trouver la transformation logarithmique, on a une expression bien plus compliquée de la forme :

$$y^* = f(y, \lambda) = \log y \left(1 + \frac{\lambda \log y}{2 \left(1 + \frac{\lambda \log y}{3 \left(1 + \frac{\lambda \log y}{4} \right)} \right)} \right)$$

Tout d'abord, on note que quand $\lambda = 0$ on retrouve bien $\log y$ comme attendu. Pourquoi cette expression compliquée ? La raison est que quand λ tend vers 0, la transformation de BOX et COX tend vers une forme indéterminée de type 0/0, ce qui est une source d'imprécision numérique (NaN). Pour les petites valeurs de λ on préfère utiliser une formule approchée qui a l'avantage d'être numériquement stable. Ceci nous explique pourquoi le paramètre `eps` est bien supérieur en valeur absolue au « epsilon machine ».

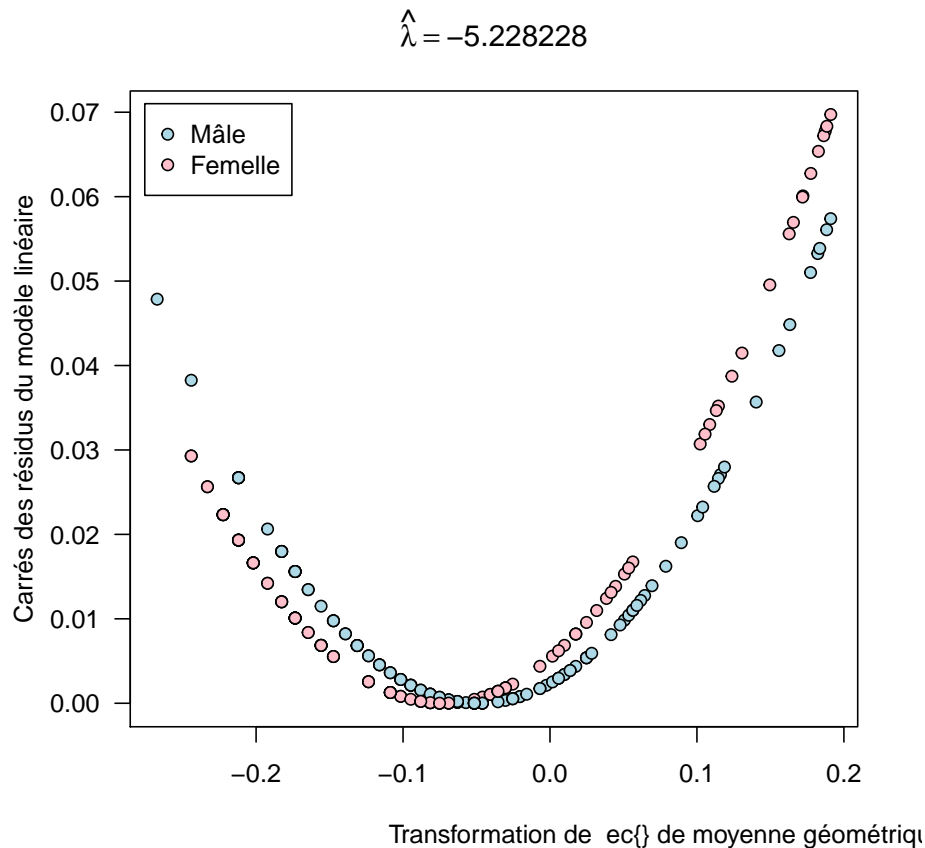
La dernière ligne de code sert à calculer le logarithme de la fonction de vraisemblance de façon particulièrement compacte. La fonction `qr.resid(xqr, yt)` renvoie la valeur des résidus obtenue en ajustant `yt` à la matrice de décomposition QR `xqr`. C'est à ce niveau que va intervenir le modèle :

```
head(lm(Age~1, data = scc)$qr$qr)
      (Intercept)
1 -15.36229150
2  0.06509446
3  0.06509446
4  0.06509446
5  0.06509446
6  0.06509446

head(lm(Age~Sex, data = scc)$qr$qr)
      (Intercept)      SexMale
1 -15.36229150 -7.68114575
2  0.06509446 -7.68114575
3  0.06509446  0.06907278
4  0.06509446  0.06907278
5  0.06509446  0.06907278
6  0.06509446 -0.06111613
```

Faisons les calculs à la main avec $\hat{\lambda}$ pour voir à quoi cela ressemble :

```
(la <- lest)
[1] -5.228228
obj <- lm(Age~Sex, y = TRUE, data = scc)
y <- obj$y
y <- y/exp(mean(log(y)))
logy <- log(y)
yt <- (y^la - 1)/la
resi <- qr.resid(obj$qr, yt)
plot(yt, resi^2, pch = 21, bg = ifelse(scc$Sex == "Male", "lightblue", "pink"), las = 1,
     xlab = "Transformation de \bec{} de moyenne géométrique unité",
     ylab = "Carrés des résidus du modèle linéaire",
     main = bquote(hat(lambda) == .(lest)))
legend("topleft", inset = 0.02, legend = c("Mâle", "Femelle"), pch = 21,
     pt.bg = c("lightblue", "pink"))
```



Le modèle résume les données par la moyenne intra-groupe, il est donc logique que les individus de tendance centrale aient de plus faibles carrés des résidus que les individus de tendance distale. On retrouve ici le fait que l'âge moyen des femelles est légèrement plus faible que celui des mâles. Vérifions que l'on retrouve la bonne valeur du log de la vraisemblance :

```
n <- length(y)
- n/2 * log(sum(resi^2))
[1] -136.1667
```

C'est cohérent. Le reste du code de la fonction a trait aux aspects graphiques et ne sera pas commenté ici.

```
if(interp) {
  sp <- spline(xl, loglik, n = 100)
  xl <- sp$x
  loglik <- sp$y
  m <- length(xl)
}
if(plotit) {
  mx <- (1L:m)[loglik == max(loglik)][1L]
  Lmax <- loglik[mx]
  lim <- Lmax - qchisq(19/20, 1)/2
  dev.hold(); on.exit(dev.flush())
  plot(xl, loglik, xlab = xlab, ylab = ylab, type
       = "l", ylim = range(loglik, lim))
  plims <- par("usr")
}
```


```

abline(h = lim, lty = 3)
y0 <- plims[3L]
scal <- (1/10 * (plims[4L] - y0))/par("pin")[2L]
scx <- (1/10 * (plims[2L] - plims[1L]))/par("pin")[1L]
text(xl[1L] + scx, lim + scal, " 95%", xpd = TRUE)
la <- xl[mx]
if(mx > 1 && mx < m)
  segments(la, y0, la, Lmax, lty = 3)
ind <- range((1L:m)[loglik > lim])
if(loglik[1L] < lim) {
  i <- ind[1L]
  x <- xl[i - 1] + ((lim - loglik[i - 1]) *
    (xl[i] - xl[i - 1]))/(loglik[i] - loglik[i - 1])
  segments(x, y0, x, lim, lty = 3)
}
if(loglik[m] < lim) {
  i <- ind[2L] + 1
  x <- xl[i - 1] + ((lim - loglik[i - 1]) *
    (xl[i] - xl[i - 1]))/(loglik[i] - loglik[i - 1])
  segments(x, y0, x, lim, lty = 3)
}
}
list(x = xl, y = loglik)
}

```

8 Ressources francophones en ligne

Ces ressources sont rangées dans un ordre, approximatif, de spécialisation croissante.

1. La section 4 du cours⁹ « tests de normalité » de Ricco RAKOTOMALALA de l'université Lyon 2 est entièrement consacré à transformation de BOX et COX.
2. Un TP¹⁰ sous  d'économétrie faisant pratiquer la transformation de BOX et COX du master MAEF « Math-Économie-Finance » de l'université Panthéon-Sorbonne.
3. La traduction française du chapitre 14 [10] « transformation de la variable dépendante »¹¹. Ce document ne doit pas être diffusé librement : il est destiné uniquement aux étudiants de la faculté des sciences économiques et de gestion de l'université de la Méditerranée, et aux étudiants des formations doctorales du GREQAM « Groupement de Recherche en Économie Quantitative d'Aix-Marseille ».
4. Un article¹² [14] discutant en détail de la transformation de BOX et COX.
5. Le « guide¹³ de bonnes pratiques pour la mise en œuvre de la méthode des prix hédoniques » de Sébastien TERRA.
6. L'article « méthodes Box-Cox, algorithmes de TRIO et demande de transport : trois consignes occamiennes pour faire rendre sens aux coefficients de régression de modèles simples et logistiques discrets ou agrégés » de Marc GAUDRY¹⁴.

9. http://eric.univ-lyon2.fr/~ricco/cours/cours/Test_Normalite.pdf

10. https://samm.univ-paris1.fr/IMG/pdf/TP_MCG_2014-2.pdf

11. <http://russell.vcharite.univ-mrs.fr/EIE/fchap14.pdf>

12. <http://www.cjrs-rcsr.org/V34/2/CJRS-RCSR-34-2-04.pdf>

13. <http://iwllearn.net/abt-iwllearn/events/workshops/ouagadougou/readingfiles/france-methode-hedoniques.pdf>

14. https://www.researchgate.net/profile/Marc_Gaudry/publication/297700780_Methodes_Box-Cox_algorithmes_de_TRIO_et_demande_de_transport_trois_consignes_

Références

- [1] T. Allison and D. V. Cicchetti. Sleep in mammals : ecological and constitutional correlates. *Science*, 194 :732–734, 1976.
- [2] P.M. Berthouex and L.C. Brown. *Statistics for Environmental Engineers*. Lewis, Boca Raton, FL., USA, 2002.
- [3] G.E.P. Box and D.R. Cox. An analysis of transformations. *Journal of the Royal Statistical Society, B*, 26 :211–252, 1964.
- [4] P. Brenot. *Éloge de la masturbation*. Zulma, 18, rue du Dragon, 75006 Paris, France, 2002.
- [5] D. Chessel, A.-B. Dufour, and J. Thioulouse. The ade4 package-I- One-table methods. *R News*, 4 :5–10, 2004.
- [6] J. Curtin. *lmSupport : Support for Linear Models*, 2017. R package version 2.9.8.
- [7] O. Dag, O. Asar, and O. Ilk. A methodology to implement Box-Cox transformation when no covariate is available. *Communications in Statistics - Simulation and Computation*, 43 :1740–1759, 2014.
- [8] O. Dag, O. Asar, and O. Ilk. Estimating Box-Cox power transformation parameter via goodness-of-fit tests. *Communications in Statistics - Simulation and Computation*, 46 :91–105, 2017.
- [9] O. Dag and O. Ilk. An algorithm for estimating Box-Cox transformation parameter in ANOVA. *Communications in Statistics - Simulation and Computation*, Accepted (June 16, 2016).
- [10] R. Davidson and J.G. MacKinnon. *Econometric Theory and Methods*. Oxford University Press, New York, USA, 2004.
- [11] D.M. Diez, C.D. Barr, and M. Cetinkaya-Rundel. *openintro : OpenIntro data sets and supplemental functions*, 2012. R package version 1.4.
- [12] J. A. F. Diniz-Filho and N. M. Tôrres. Phylogenetic comparative methods and the geographic range size -body size relationship in new world terrestrial carnivora. *Evolutionary Ecology*, 16 :351–367, 2002.
- [13] W.J. Dixon. Analysis of extreme values. *Ann. Math. Stat.*, 4 :488–506, 1950.
- [14] J. Dubé, F. Des Rosiers, and M. Thériault. Le choix de la forme fonctionnelle dans la théorie hédonique : retour sur un vieux débat. *Canadian Journal of Regional Science/Revue canadienne des sciences régionales*, 34 :81–90, 2011.
- [15] D. Fife. *fifer : A collection of miscellaneous functions.*, 2014. R package version 1.0.

occamiennes_pour_faire_rendre_sens_aux_coefficients_de_regression_de_modeles_simples_et_logistiques_discrets_ou_agreges/links/56e060c108ae979addf0f115.pdf?origin=publication_list

- [16] J. Fox and S. Weisberg. *An R Companion to Applied Regression*. Sage, Thousand Oaks CA, second edition, 2011.
- [17] V.M. Guerrero. Time-series analysis supported by power transformations. *Journal of Forecasting*, 12 :37–48, 1993.
- [18] R.A. Johnson and D.W. Wichern. *Applied Multivariate Statistical Analysis*. Pearson Prentice Hall, USA, 2007.
- [19] Lukasz Komsta. *outliers : Tests for outliers*, 2011. R package version 0.14.
- [20] M. Kuhn, J. Wing, S. Weston, A. Williams, C. Keefer, A. Engelhardt, T. Cooper, Z. Mayer, B. Kenkel, M. Benesty, R. Lescarbeau, A. Ziem, L. Scrucca, Y. Tang, C. Candan, and T. Hunt. *caret : Classification and Regression Training*, 2016. R package version 6.0-73.
- [21] P.-S. Laplace. *Théorie analytique des probabilités*. Veuve Courcier, Paris, France, 1812.
- [22] H. Lineweaver and D. Burk. The determination of enzyme dissociation constants. *Journal of the American Chemical Society*, 56 :658–666, 1934.
- [23] A.I. McLeod and Y. Zhang. Improved subset autoregression : With R package. *Journal of Statistical Software*, 28(2), 2008.
- [24] L. Michaelis and M.L. Menten. Die Kinetik der Invertinwirkung. *Biochemische Zeitschrift*, 49 :333–369, 1913.
- [25] S.P. Millard. *EnvStats : An R Package for Environmental Statistics*. Springer, New York, 2013.
- [26] P.J. Northrop. *rust : Ratio-of-Uniforms Simulation with Transformation*, 2016. R package version 1.1.0.
- [27] P.J. Ribeiro Jr and P.J. Diggle. *geoR : Analysis of Geostatistical Data*, 2016. R package version 1.7-5.2.
- [28] L. Say and D. Pontier. What determines testis size in the domestic cat (*Felis catus* L.)? *Biological Letters*, 43 :41–49, 2006.
- [29] S. S. Shapiro and M. B. Wilk. An analysis of variance test for normality (complete samples). *Biometrika*, (52) :591–611, 1965.
- [30] A. Signorell, L. Chhay, and R.J. Hyndman. *DescTools : Tools for Descriptive Statistics*, 2016. R package version 0.99.18.
- [31] W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth edition, 2002.