# Comparaison des fonctions graphiques de base de R à ggplot2 pour produire un Boxplot

Camille Girard-Bock et Danye Marois

2015-04-13

## Table des matières



---

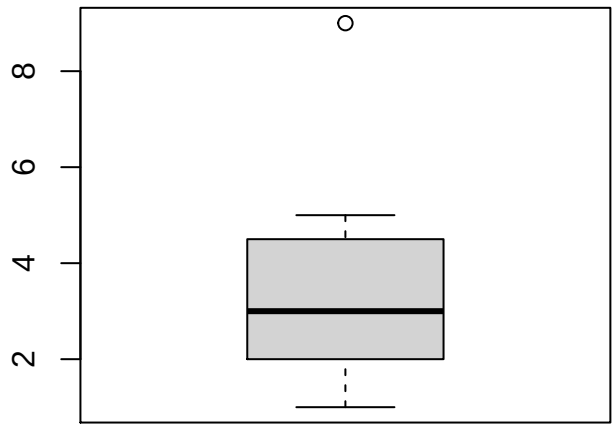
## 1 Introduction

Les graphiques de type boxplot sont utilisés afin de représenter de façon simple un ensemble de données. Le boxplot permet de visualiser rapidement le jeu de données à travers une représentation graphique des quartiles.

Les quartiles sont au nombre de trois et divisent les échantillons en quatre groupes contenant chacun le quart des échantillons : la médiane, qui correspond au second quartile (**Q2**), sépare le jeu de données en deux parts égales alors que le premier(**Q1**) et troisième(**Q3**) quartile redivisent en deux parts égales ces parties inférieures et supérieures du jeu de données, respectivement.

Le graphique de type boxplot représente des données au moyen d'une boîte dont les bords horizontaux supérieurs et inférieurs correspondent respectivement au troisième et premier quartile, tandis que le second quartile est représenté d'un trait horizontal traversant la boîte. La longueur de la boîte correspond alors à l'écart interquartile (**IQR**, InterQuartile Range), une mesure de la dispersion représentée par la différence entre le troisième et premier quartile. Prenons un exemple simple :

```r
boxplot(c(1,2,2,3,4,5,9))
```




Dans cet exemple, **Q1** = 2, **Q2**, la médiane = 3, **Q3** = 4,5 et l'**IQR** = 4,5 - 2 = 2,5. Les barres de part et d'autre de la boîte indiquent la plus petite donnée supérieure ou égale à **Q1** - (1,5 x IQR) = 2 - (1,5 x 2,5) = -1, 75, et la plus grande donnée inférieure ou égale à **Q3** + (1,5 x **IQR**) = 4,5 + (1,5 x 2,5) = 8,25 ; dans cet exemple, la ligne inférieure représente la donnée 1, alors que la ligne supérieure représente la donnée 5. Les données au-delà de ces 2 lignes sont représentées par un point (ici la donnée 9).

## 2 Jeu de données

À titre de démonstration, nous utiliserons le jeu de données *day.csv* provenant de la base de données *Bike Sharing Dataset* téléchargeable sur le *UCI Machine Learning Repository* à l'adresse suivante : https://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset.

Créons tout d'abord le jeu de données qui servira à la démonstration.

```r
bike <- read.csv("day.csv")
```

```r
str(bike)
```

```
## 'data.frame':    731 obs. of  16 variables:
##  $ instant   : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ dteday    : chr  "2011-01-01" "2011-01-02" "2011-01-03" "2011-01-04" ...
##  $ season    : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ yr        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ mnth      : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ holiday   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ weekday   : int  6 0 1 2 3 4 5 6 0 1 ...
##  $ workingday: int  0 0 1 1 1 1 1 0 0 1 ...
```

```
##  $ weathersit: int  2 2 1 1 1 1 2 2 1 1 ...
##  $ temp      : num  0.344 0.363 0.196 0.2 0.227 ...
##  $ atemp     : num  0.364 0.354 0.189 0.212 0.229 ...
##  $ hum       : num  0.806 0.696 0.437 0.59 0.437 ...
##  $ windspeed : num  0.16 0.249 0.248 0.16 0.187 ...
##  $ casual    : int  331 131 120 108 82 88 148 68 54 41 ...
##  $ registered: int  654 670 1229 1454 1518 1518 1362 891 768 1280 ...
##  $ cnt       : int  985 801 1349 1562 1600 1606 1510 959 822 1321 ...
```

La fonction `str(bike)` nous permet de voir que le data.frame `bike` contient 731 observations et 16 variables. Dans le cadre de notre démonstration, nous nous intéressons aux variables suivantes : i) `cnt` = nombre total de locations par jour ; ii) `yr` = année, où 0 = 2011, et 1 = 2012, iii) `mnth` = mois de l'année, iv) `workingday` = variable dichotomique prenant la valeur 0 si la journée correspond à un jour férié ou de fin de semaine, ou 1 sinon.

# 3 Boxplot simple avec les fonctions de base de R

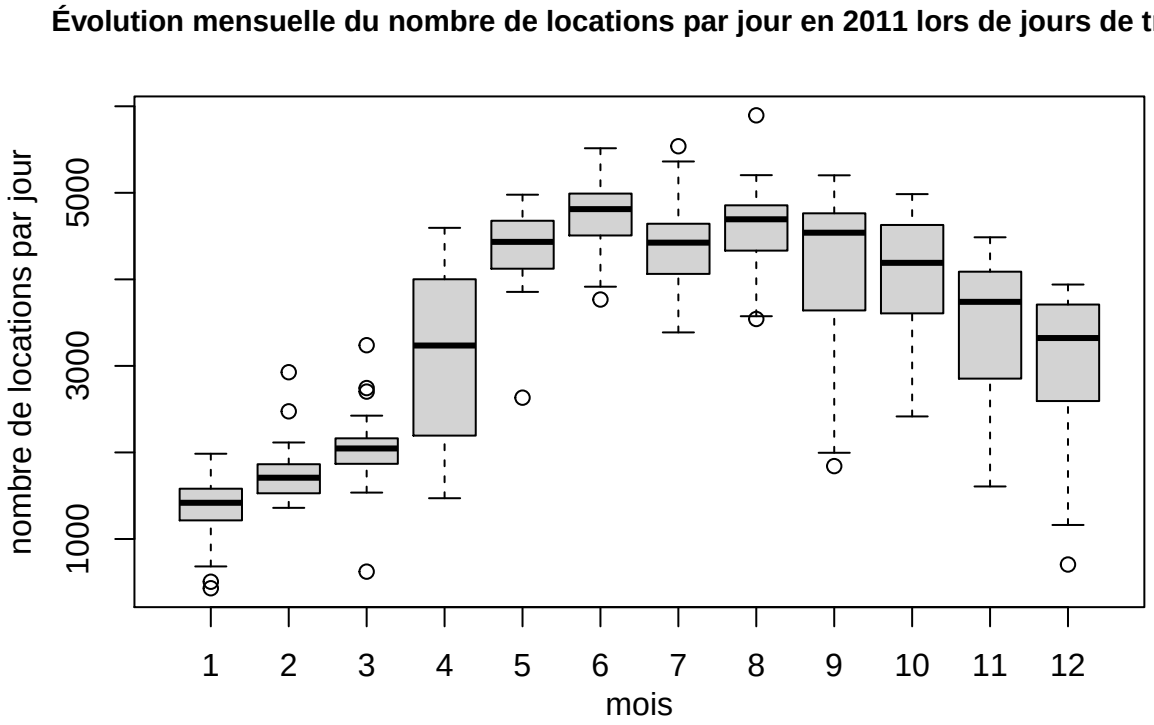
Dans le cadre de l'exemple, nous utiliserons un sous-ensemble du data.frame `bike`, soient les observations correspondant à l'année 2011 et aux jours de travail seulement.

```
bike2011 <- bike[bike$yr == 0 & bike$workingday == 1 , ]
```

La fonction de base dans R est `boxplot`. Il faut fournir en argument :

- la formule de type `y ~ grp`, où `y` représente un vecteur de données numériques qui seront divisées en différents groupes (`grp`, qui est normalement un facteur). Dans notre exemple, `y=cnt`, et `grp=mnth`.
- le jeu de données (`data=`).
- un sous-ensemble du jeu de données (`subset=`), s'il y a lieu. Cet argument ne sera pas utilisé dans notre exemple, étant donné que le jeu de données `bike2011` représente déjà un sous-ensemble du jeu de données.
- on peut également indiquer quoi faire avec les données manquantes à l'aide de `na.action` ; par défaut, `na.action=NULL`, i.e que les valeurs manquantes sont ignorées.

```
boxplot(cnt ~ mnth, data = bike2011, ann = FALSE)
# ajout du titre et identifications des 2 axes
mtext("Évolution mensuelle du nombre de locations par jour en 2011 lors de jours de travail",
      side = 3, cex = 0.95, font = 2, line = 1.5)
mtext("mois", side = 1, cex = 1, line=2)
mtext("nombre de locations par jour", side = 2, cex = 1, line = 2.5)
# Note : Ces ajouts auraient aussi pu être effectués grâce aux arguments
# main, xlab et ylab directement dans l'appel à la fonction boxplot.
```




*Note* : dans notre exemple, `mnth` est de type *integer*. Bien que la fiche d'aide de `boxplot` indique que le groupe est normalement de type *facteur*, R a tout de même produit adéquatement le graphique.

# 4 Boxplot simple avec `ggplot2`

`ggplot2` est un package crée par Hadley Wickham et qui permet de créer facilement différents types de graphiques pour différents types de données numériques ou catégoriques, univariées ou multivariées (*Réf. : http://www.statmethods.net/advgraphs/ggplot2.html* )

L'utilisation de `ggplot2` nécessite l'installation du package `ggplot2` de la façon suivante :

```
install.packages("ggplot2")
```

Il faut ensuite charger le package dans la session de travail. Ceci se fait à l'aide de la commande suivante :

```
library(ggplot2)
```

L'énoncé de base pour produire un **boxplot** avec `ggplot` est le suivant :

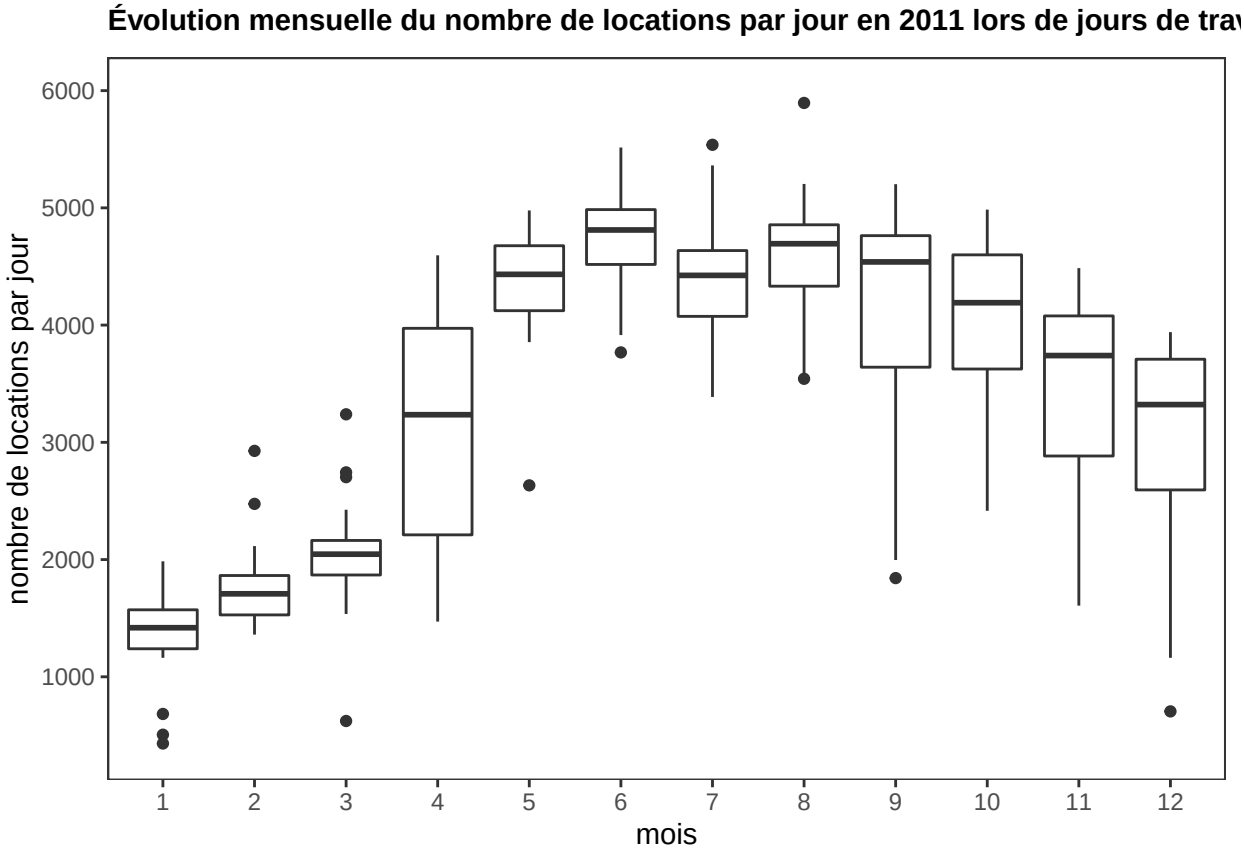
```
bplot <- ggplot(bike2011, aes(x=factor(mnth), y=cnt)) + geom_boxplot()
```

où `bike2001` est le jeu de données à utiliser, `aes` sert à définir les variables `x` et `y` et `geom_boxplot` indique que le graphique est de type *boxplot*. À noter, que contrairement au R de base, `ggplot` exige que la variable `x` soit de type facteur (d'où l'utilisation de `x=factor(mnth)`).

L'apparence du graphique peut être modifiée en rajoutant des fonctions au graphique de base :

- la fonction `labs` permet d'ajouter un nom aux 2 axes, ainsi qu'un titre au graphique,
- la fonction `scale_y_continuous` permet de déterminer les limites ainsi que la graduation de l'axe y,
- la fonction `theme_bw` fait en sorte que le fond du graphique soit blanc,
- la fonction `theme` permet de modifier l'apparence du graphique et de ses composantes (par exemple, le titre). L'argument `panel.grid` permet d'éliminer les quadrillages pour l'axe x et l'axe y. Finalement, `plot.title` permet de modifier l'apparence du titre en utilisant des caractères gras, et en le déplaçant horizontalement vers le haut.

```
bplot +
  labs(x = "mois", y = "nombre de locations par jour",
       title = "Évolution mensuelle du nombre de locations par jour en 2011 lors de jours de travail") +
  scale_y_continuous(limits=c(400,6000), breaks=0:6000*1000) +
  theme_bw() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        plot.title = element_text(face=quote(bold), vjust = 2, size = 11))
```




# 5 Boxplot complexe avec ggplot2

Pour cette partie, nous utiliserons le jeu de données original, `bike`, contenant donc aussi des données pour l'année 2012 et pour les jours de congé.

**Ajouter une distinction par couleur selon des sous-groupes**

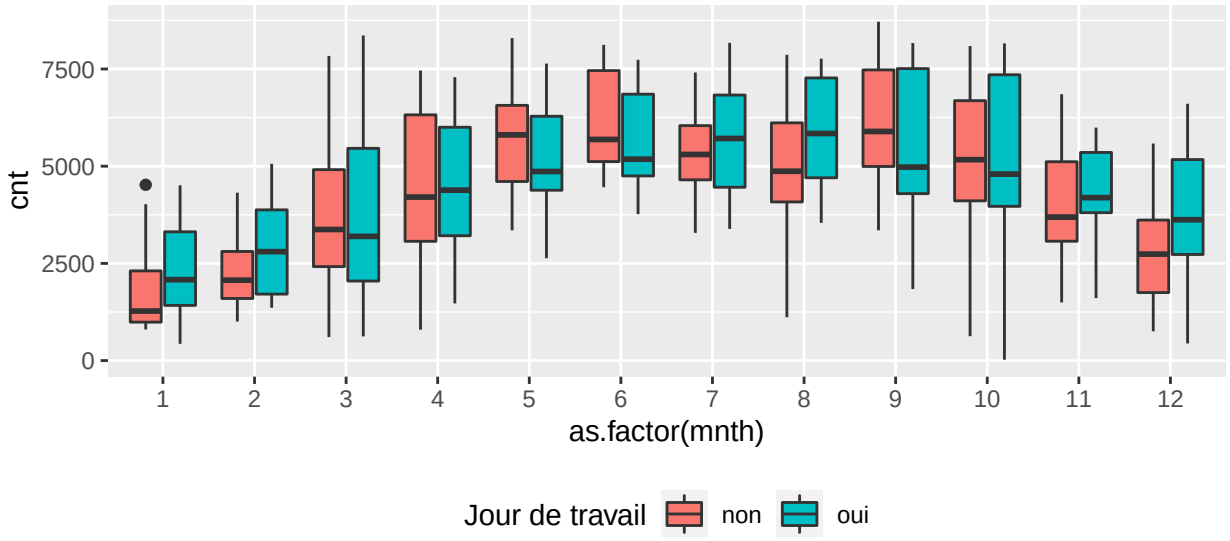
On souhaite représenter dans des boîtes de couleurs différentes les données pour les jours de travail et celles

pour les jours de fin de semaine et jours fériés. Dans le jeu de données, la colonne `workingday` contient un 1 si c'est un jour de travail et un 0 si c'est un jour de fin de semaine ou un jour férié. Ci-dessous, nous créons une nouvelle colonne nommée `travail` qui contiendra plutôt un `oui` pour les jours de travail et un `non` les autres jours.

```
bike$travail[bike$workingday==0] <- "non"
bike$travail[bike$workingday==1] <- "oui"
```

L'argument `fill` dans la fonction `aes` (qui est elle même un argument de la fonction `ggplot`) nous permet de spécifier que c'est selon la valeur retrouvée dans `bike$travail` que nous souhaitons colorer les boîtes de notre boxplot. L'argument `legend.position` sous la fonction `theme` (qui nous permettait plus haut de changer l'aspect du titre) nous permet de spécifier l'emplacement de la légende ainsi créée tandis que la fonction `labs` nous permet de spécifier le nom associé à cette légende :

```
ggplot(bike,aes(x = as.factor(mnth),  y=cnt, fill= travail)) +
  geom_boxplot() +
  theme(legend.position="bottom") +
  labs(fill = "Jour de travail")
```




**Ajouter un second graphique sous un autre onglet**

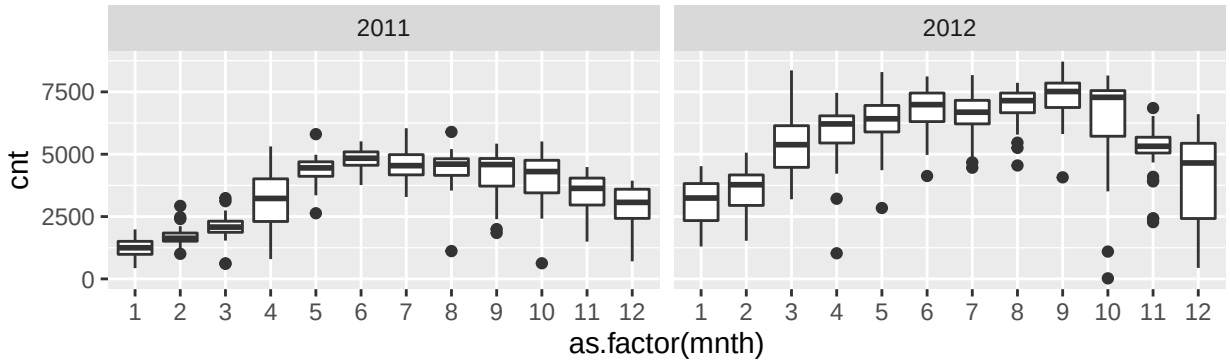
On souhaite différencier les données de 2011 et 2012 dans notre graphique. Pour ce faire, on veut diviser le graphique en deux parties pour chacune des deux années pour lesquelles nous possédons des données.

Pour ce faire, nous créons une colonne `year` qui ne contiendra que l'année (et non la date complète comme dans la colonne `date`). Puisque la colonne `yr` nous indique l'âge du programme en année et que le programme a commencé en 2011, nous savons qu'une valeur de 0 pour la colonne `yr` correspond à 2011 alors qu'une valeur de 1 correspond à 2012 :

```
bike$year[bike$yr==0] <- "2011"
bike$year[bike$yr==1] <- "2012"
```

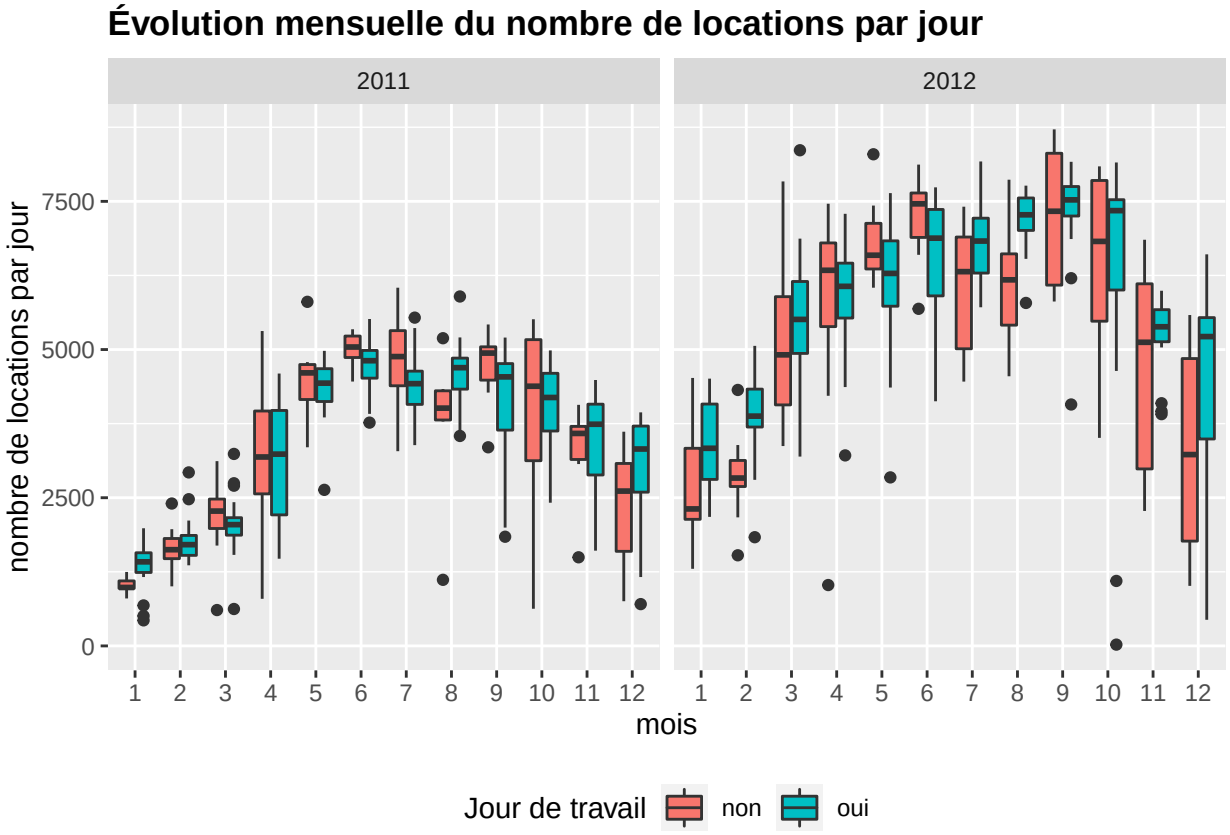
C'est la fonction `facet_grid` qui nous permet de créer cette distinction selon les années, en produisant un graphique pour chacune d'entre elles :

```
ggplot(bike,aes(x = as.factor(mnth),  y=cnt)) +
  geom_boxplot() +
  facet_grid(~ year)
```




**Graphique complet :** En combinant l'usage de l'argument `fill` présenté plus haut à celui de la fonction `facet_grid` et aux fonctions permettant d'afficher le titre principal (`ggtitle`) et ceux des axes (`xlab` et `ylab`) et de les modifier (argument `plot.title` dans `theme`) nous obtenons un graphique de type boxplot complet :

```
ggplot(bike,aes(x = as.factor(mnth),  y=cnt, fill= travail)) +
  geom_boxplot() +
  facet_grid(~ year) +
  ggtitle("Évolution mensuelle du nombre de locations par jour") +
  theme(plot.title = element_text(lineheight=.8, face="bold"), legend.position="bottom") +
  xlab("mois") +
  ylab("nombre de locations par jour") +
  labs(fill = "Jour de travail")
```




## Bibliographie


- Définition des quartiles : Weisstein, Eric W. “Quartile.” From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/Quartile.html
- Introduction à ggplot : http://blog.echen.me/2012/01/17/quick-introduction-to-ggplot2/
- Boxplot de base avec ggplot : http://docs.ggplot2.org/current/geom_boxplot.html et http://www.cookbook-r.com/Graphs/Plotting_distributions_(ggplot2)/
- Informations sur la légende du boxplot : http://www.sthda.com/english/wiki/ggplot2-box-plot-quick-start-guide-r-software-and-data-visualization#change-box-plot-fill-colors
- Informations sur les titres : http://www.cookbook-r.com/Graphs/Titles_(ggplot2)/ et http://www.sthda.com/english/wiki/ggplot2-title-main-axis-and-legend-titles