REGRESSION LINEAIRE SIMPLE avec R - exemple du cours

DU ETD / 2016

IUT Vannes / Dpt STID

2016

Problématique : Etude du montant du loyer d'un appartement en fonction de sa superficie.

On a relevé en avril 2016 sur le site *seloger.com* dans les petites annonces les superficies (en m2) et les montant des loyers (en Euros, charges comprises) de 104 appartements de 2 pièces à louer à Vannes. On veut apprécier le rôle de la superficie sur le prix de la location d'un appartement. Les données du tp sont dans le fichier *APPART.txt*.

Démarche

1. Importation des données.
2. Représentation graphique et analyse du nuage de point.
3. Mise en oeuvre de la régression linéaire simple avec R : **fonction lm()**

- Proposer un modèle statistique permettant d'étudier le lien entre les variables pour répondre à la problématique
- Estimation des paramètres du modèle
- Indicateur de la qualité de l'ajustement

4. Tracer la droite de régression sur le nuage de points
5. Analyse des résidus
6. Prévision d'une nouvelle valeur

Mise en place de la session de travail

- Dans le répertoire *H:/Mes Documents*, créer un répertoire *modeleLineaire*, puis un répetoire *TD-R* puis un répertoire *data*
- Sur l'espace de travail du DU sur l'ENT, récupérer le fichier *APPART.txt* et l'enregistrer dans le répertoire *data*

Ouvrir R-studio

- Définir le répertoire de travail: *Session > Set Working Directory > Choose Directory* et sélectionner le répertoire *modeleLineaire*. La commande R est générée automatiquement :

```
setwd(dir = "C:/Businessdecision/Enseignement/STID - LP/Enseignement/ModeleLineaire-DU-ETD/")
```

- Importer les données: la fenêtre en haut à droite dans R-studio dispose d'un menu permettant d'accéder à un assistant pour l'importation : *Environnement > Import Dataset > from text file*. On sélectionne alors le fichier *APPART.txt* et on précise certaines informations : nom du *data frame* dans lequel les données seront stockées, noms des variables sur la 1ere ligne du fichier ? (headings Yes/no); Séparateurs

des champs (tabulation, espace, etc); Séparateur des décimales (point, virgule, etc); Séparateur texte (guillemets, etc)

Les données sont stockées en R dans un objet *data frame* (individus en lignes et variables en colonne, possibilité de types différents pour les variables). La encore, le code R est généré automatiquement.

```
donnees = read.delim2("data/APPART.txt")
```

Statistiques descriptives simples

Le but est de vérifier de l'importation des données. On peut visualiser le tableau de données importé en cliquant sur le nom de celui-ci dans la liste des objets existants dans la session R, dans l'onglet *environnement* de la fenêtre en haut à droite. A coté du nom de l'objet, on a également des informations sur son type et ses dimensions. Les objects créés dans la session sont classés par type (data, values, functions, etc).

On peut aussi utiliser les lignes de commandes R suivantes pour un rapide coup d'oeil sur les données : vérification des dimensions du tableau de données, connaitre les noms des variables et individus, etc ...

```
dim(donnees)
```

```
## [1] 104   3
```

```
names(donnees)
```

```
## [1] "LOYERCC"    "SUPERFICIE" "VILLE"
```

On rappelle que R est un langage orienté objet et certaines fonctions de bases s'appliquent sur plusieurs types d'objets en s'adaptant, on en utilisera trois dans le tp : **summary()**, **plot()** et **predict()**. Ici, appliquée à un objet de type **data.frame**, la fonction **summary()** retourne les statistiques descriptives des différentes variables : moyenne, quantiles, min et max pour les variables quantitatives, et fréquence de chaque modalité pour les variables qualitatives.

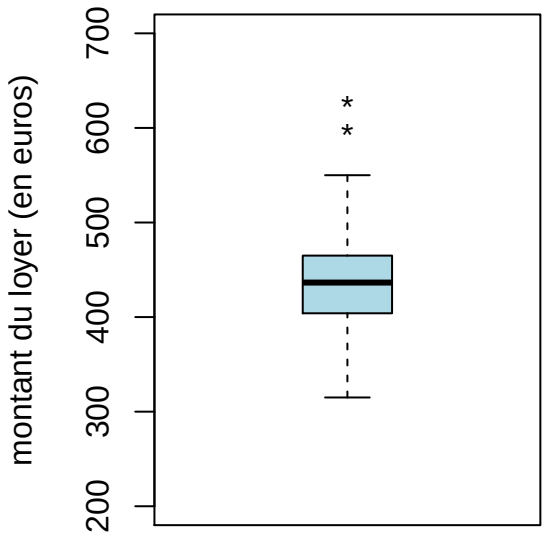
```
summary(donnees)
```

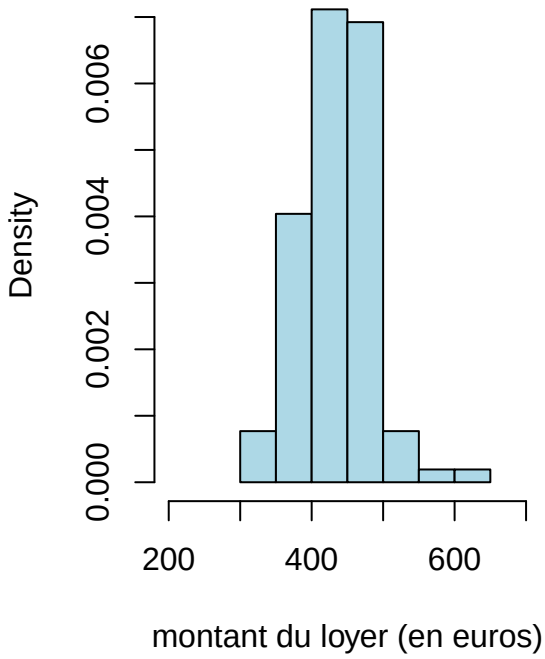
```
##     LOYERCC        SUPERFICIE       VILLE
##  Min.   :315.0   Min.   :25.00   VANNES:104
##  1st Qu.:404.5   1st Qu.:37.95
##  Median :436.5   Median :42.00
##  Mean   :437.5   Mean   :42.03
##  3rd Qu.:465.0   3rd Qu.:45.31
##  Max.   :630.0   Max.   :92.00
```

Représentation graphique du nuage de points

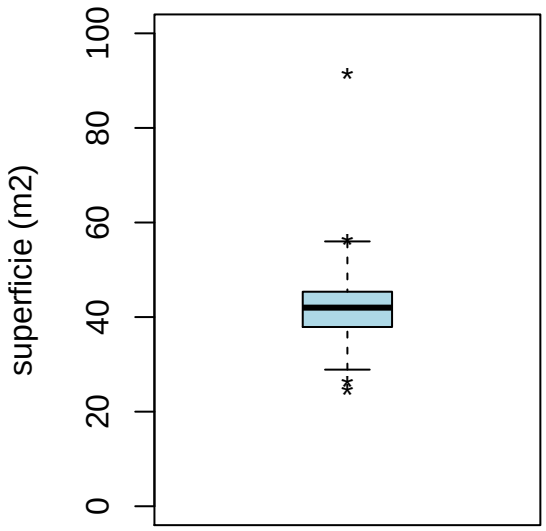
La fonction **plot** permet de réaliser un nuage de point. La fonction **boxplot** permet de réaliser une boite de dispersion ("boites à moustaches"). On peut personnaliser le graphique en ajoutant des arguments optionnels (séparés par des virgules) en précisant par exemple le symbole représentant chaque point (pch), sa taille (cex), sa couleur (col), les limites des axes (xlim et ylim) et leurs labels (ylab et ylab), et ajouter un titre (main) :

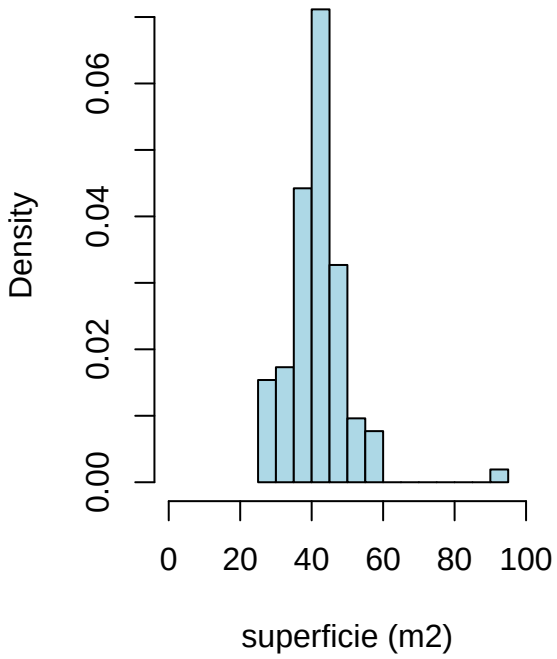
```
# Variable LOYERCC
boxplot(donnees$LOYERCC, col="lightblue", boxwex=0.5, ylim=c(200,700), pch="*", ylab="montant du loyer
hist(donnees$LOYERCC, col="lightblue", breaks=10, xlim=c(200,700), xlab="montant du loyer (en euros)", m
```



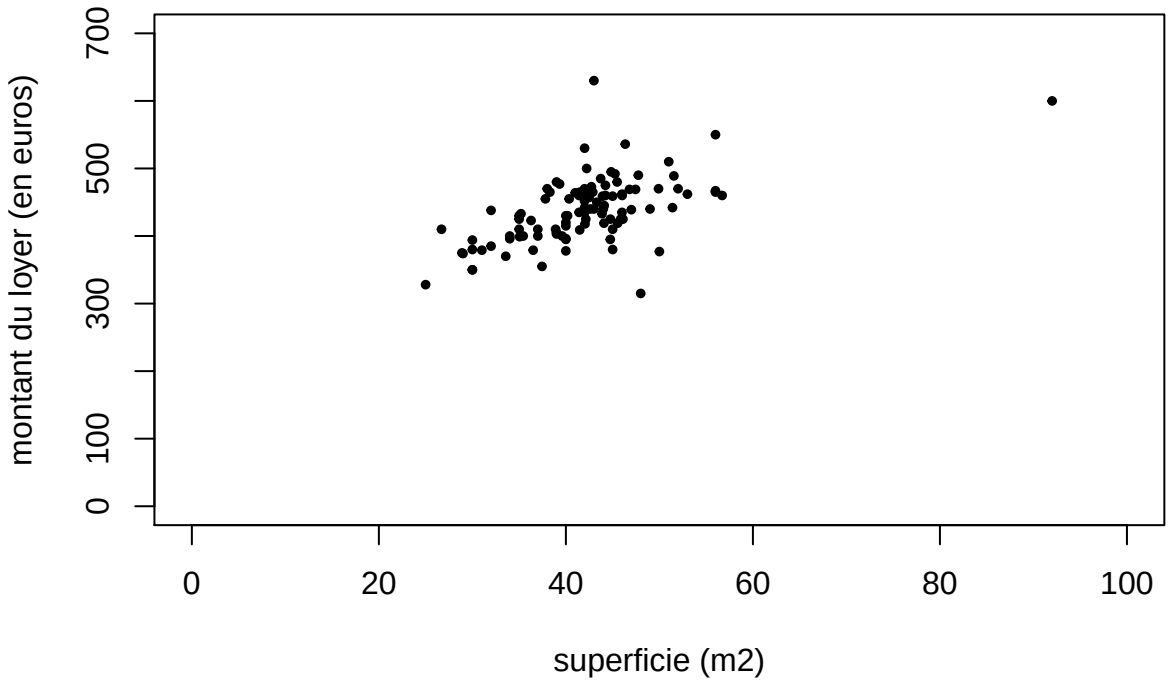



```
# Variable SUPERFICIE
boxplot(donnees$SUPERFICIE, col="lightblue", boxwex=0.5, ylim=c(0,100), pch="*", ylab="superficie (m2)"
hist(donnees$SUPERFICIE, col="lightblue", breaks=10, xlim=c(0,100), xlab="superficie (m2)", main="", fr
```






```
# Nuage de point avec la superficie en abscisse et le montant du loyer en ordonnée
plot(x=donnees$SUPERFICIE, y=donnees$LOYERCC, pch=20, cex=0.8, col=1,xlim=c(0,100), ylim=c(0,700), xlab=
```



La fonction **cor(x,y)** permet de calculer le coefficient de corrélation entre les variables x et y.

```
cor(donnees$SUPERFICIE,donnees$LOYERCC)
```

```
## [1] 0.6044104
```

Régression linéaire simple : fonction lm() de R

Aide sur la fonction : ?lm ou help("lm") ou rechercher la fonction dans le menu *help*

- Écrire le modèle (formula) : variable à expliquer ~ variable explicative

```
lm(formula=LOYERCC~SUPERFICIE,data=donnees)
```

```
##
## Call:
## lm(formula = LOYERCC ~ SUPERFICIE, data = donnees)
##
## Coefficients:
## (Intercept)   SUPERFICIE
##     282.831        3.679
```

```
# Si on veut un modèle sans constante : ajouter -1 dans la formule :
# lm(formula=prix~-1+superficie,data=appartRennes)
```

Les résultats ne sont pas stockés seuls les principaux renseignements sont présentés dans la sortie de R. Pour stocker le résultat :

```
res.lm = lm(formula=LOYERCC~SUPERFICIE,data=donnees)
res.lm
```

```
##
## Call:
## lm(formula = LOYERCC ~ SUPERFICIE, data = donnees)
##
## Coefficients:
## (Intercept)   SUPERFICIE
##     282.831        3.679
```

Cela permet d'afficher les estimations des paramètres (coefficients). L'ensemble des informations contenues dans l'objet *res.lm* est listé à l'aide de la commande **names()**. L'application de la fonction **summary()** sur cet objet permet d'obtenir d'autres informations comme l'écart-type résiduel (estimation de s_E) et le R^2 entre autre :

```
names(res.lm)
```

```
##  [1] "coefficients"  "residuals"     "effects"       "rank"
##  [5] "fitted.values" "assign"        "qr"            "df.residual"
##  [9] "xlevels"       "call"          "terms"         "model"
```

```
summary(res.lm)
```

```
##
## Call:
## lm(formula = LOYERCC ~ SUPERFICIE, data = donnees)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -144.401  -21.993   -1.955   22.938  188.992
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 282.8307    20.5519  13.762  < 2e-16 ***
## SUPERFICIE    3.6786     0.4801   7.662 1.09e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.7 on 102 degrees of freedom
## Multiple R-squared:  0.3653, Adjusted R-squared:  0.3591
## F-statistic: 58.71 on 1 and 102 DF,  p-value: 1.094e-11
```

On peut accéder en particulier aux estimations des coefficients :

```
res.lm$coefficients
```

```
## (Intercept)  SUPERFICIE
##  282.830721    3.678552
```

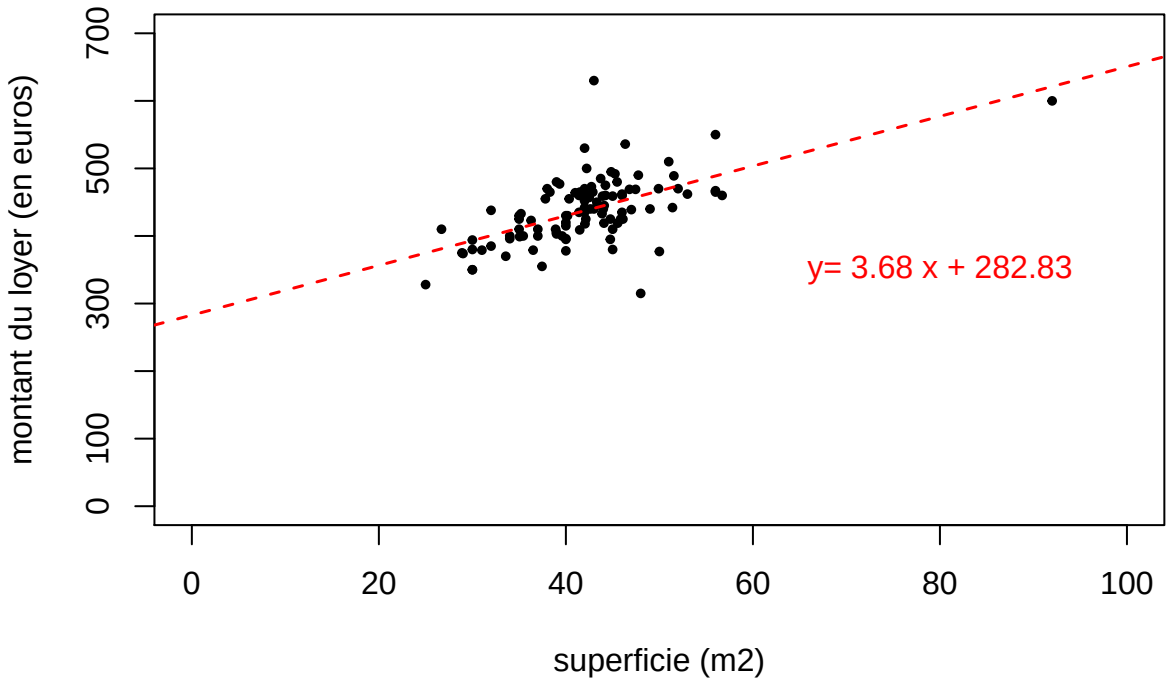
```
res.lm$coefficients[2]
```

```
## SUPERFICIE
##   3.678552
```

Représentation graphique (suite) : tracer la droite de régression sur le nuage de points

La fonction **abline(a,b,..)** permet d'ajouter sur un graphique existant une droite d'équation $y = a + bx$. On peut également ajouter des arguments comme par exemple la couleur de la droite (col), le type de ligne (pointillée/continue..) (lty), ou son épaisseur (lwd).

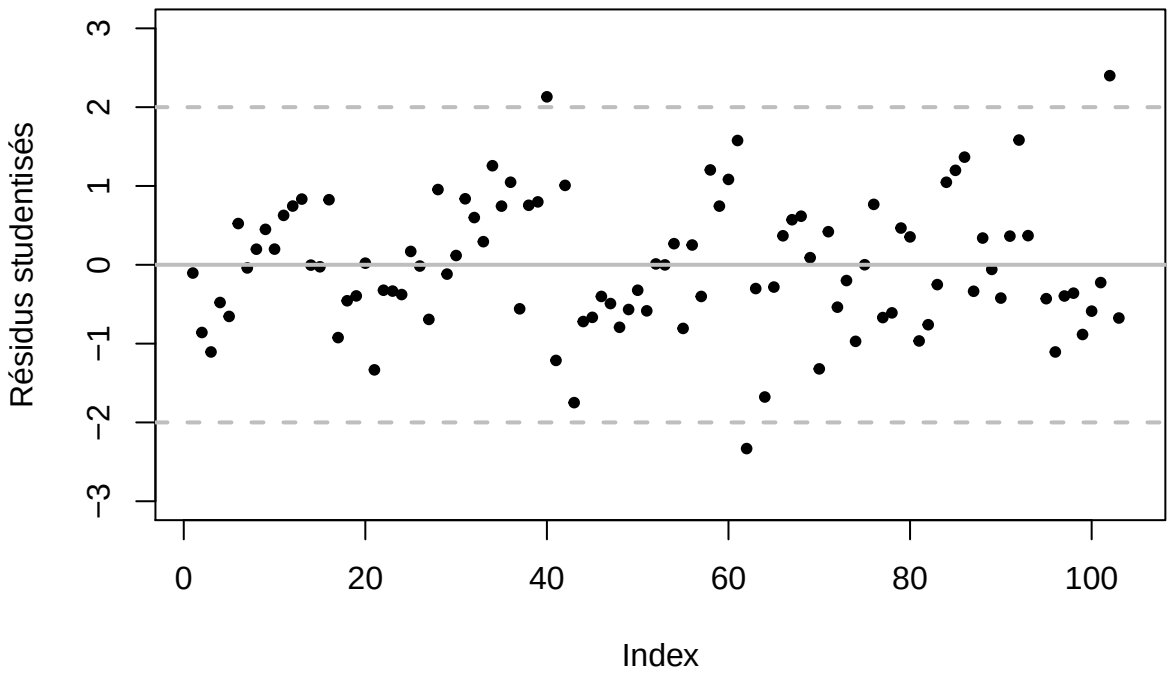
```
# Nuage de points avec la superficie en abscisse et le montant du loyer en ordonnée

plot(x=donnees$SUPERFICIE, y=donnees$LOYERCC, pch=20, cex=0.8, col=1,xlim=c(0,100), ylim=c(0,700), xlab=

# ajout de la doite de régression, en rouge (col=2) et en pointillés (lty=2)
abline(a=res.lm$coefficients[1], b= res.lm$coefficients[2],col=2, lty=2, lwd=1.5)
# pour écrire l'équation sur le graphique
text(x = 80, y= 350, labels = paste("y=",round(res.lm$coefficients[2],2),"x +",round(res.lm$coefficients
```



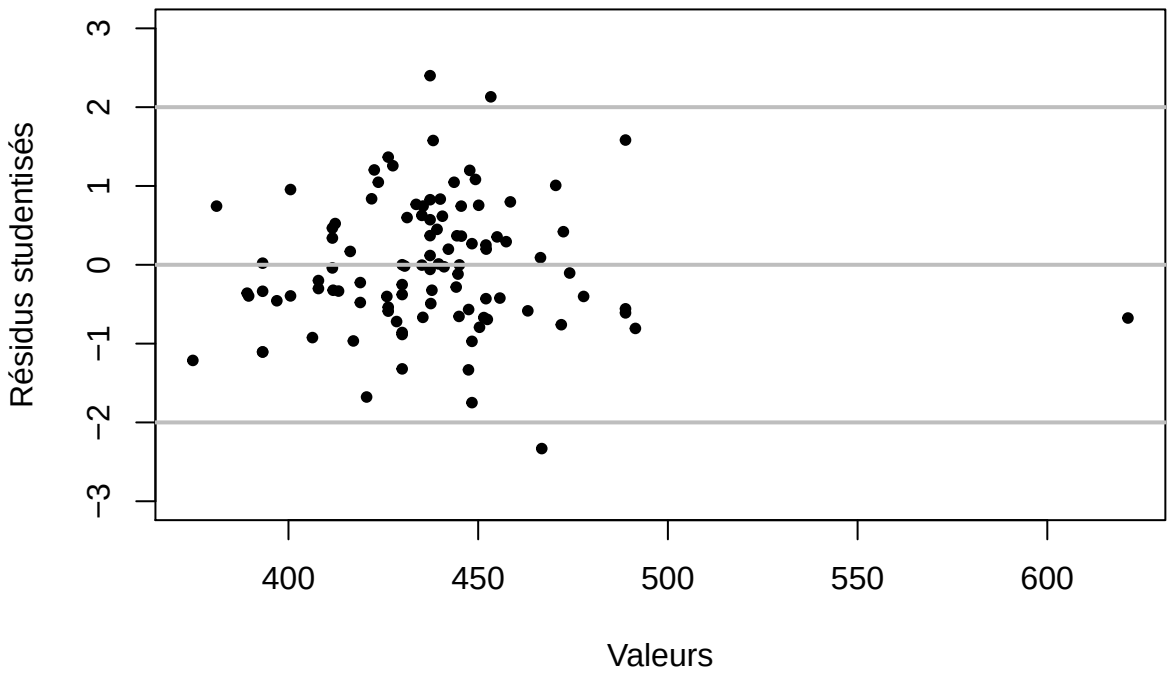
Validation du modèle : analyse des résidus

La fonction **rstudent()** appliquée au modèle permet d'obtenir les résidus studentisés :

```
rstud = rstudent(res.lm)
# graphique des résidus studentisés
plot(rstud,pch=20,ylab="Résidus studentisés",ylim=c(-3,3))
abline(h=c(0), col="grey",lty=1,lwd=2)
abline(h=c(-2,2), col="grey",lty=2,lwd=2)
```

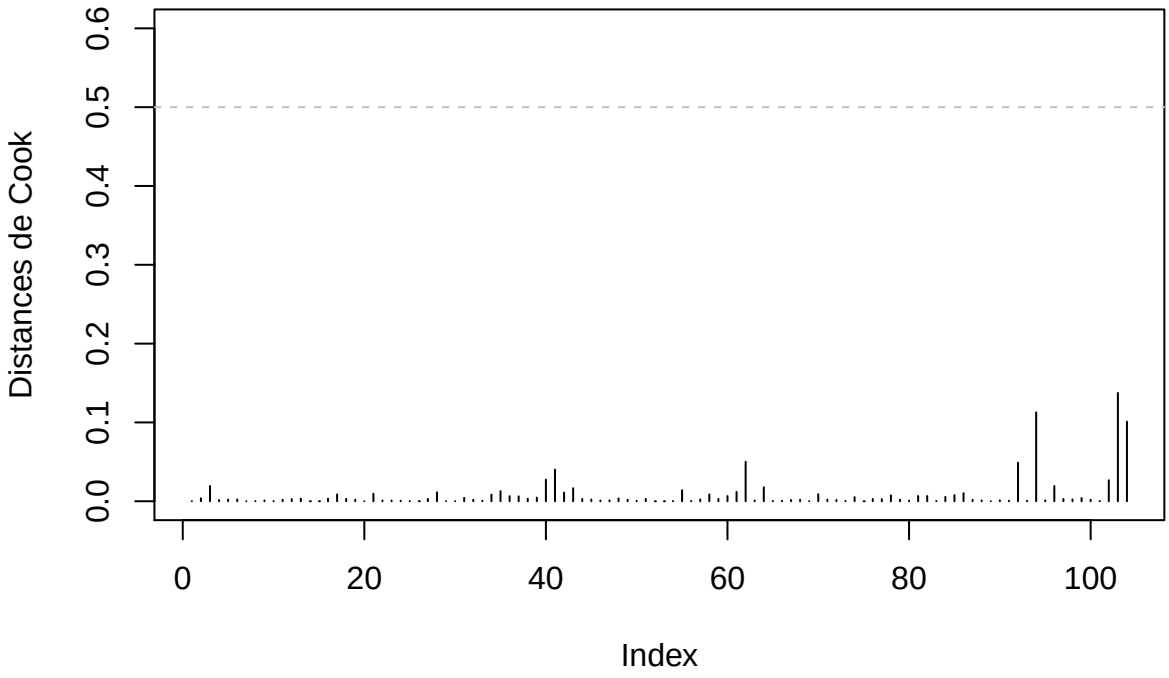



```r
#Tracer les résidus studentisés en fonction des valeurs ajustées :
plot(res.lm$fitted.values,rstud,pch=20,ylab="Résidus studentisés",ylim=c(-3,3),xlab="Valeurs")
abline(h=c(0,-2,2), col="grey",lty=1,lwd=2)
```



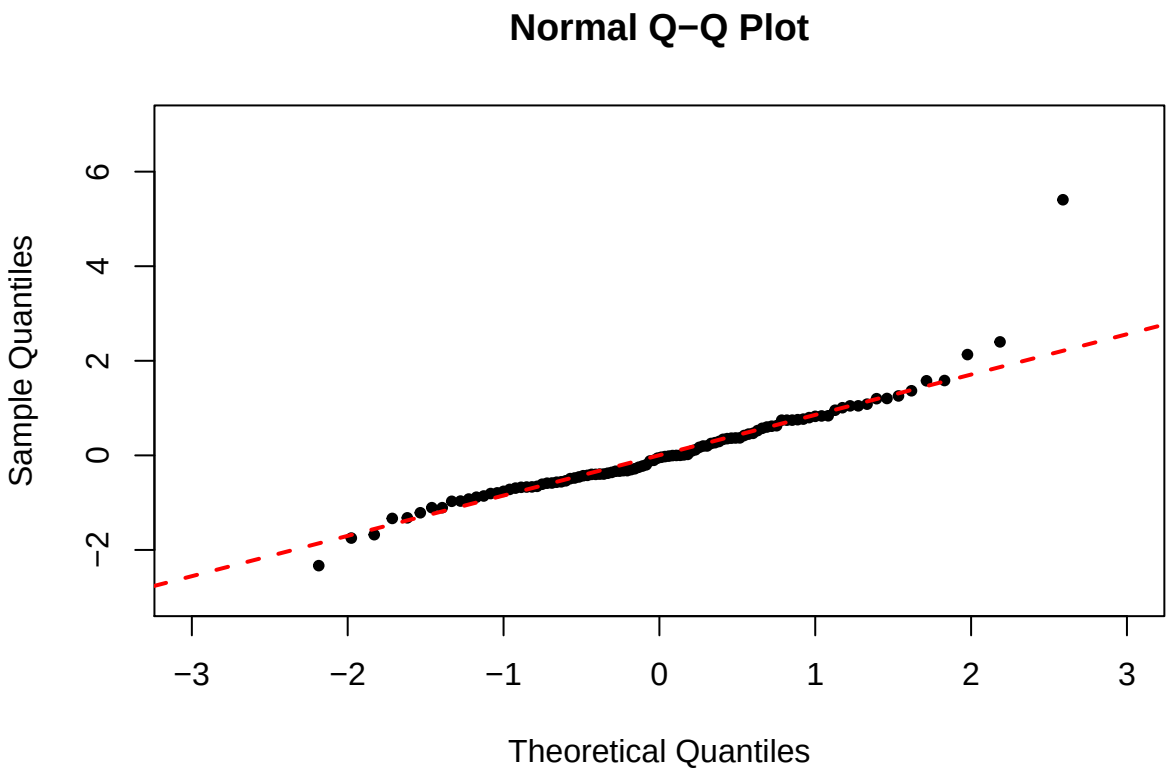
La fonction **cooks.distance()** appliquée au modèle permet d'obtenir les distances de Cook :

```
res.cook=cooks.distance(model=res.lm)
plot(res.cook, type="h",ylab="Distances de Cook", ylim=c(0,0.6))
abline(h=0.5,col="gray",lty=2)
```



Pour tracer le graphique des quantiles et ajouter la bissectrice :

```
res.qq=qqnorm(rstud, pch=20, ylim=c(-3,7),xlim=c(-3,3))
qqline(rstud, lty=2, lwd=2, col=2)
```



La fonction **identify()** permet d'identifier un point sur le graphique (ne pas fermer le graphique avant de lancer la commande et faire *ECHAP* pour quitter après avoir cliqué sur le(s) point(s) à identifier !)

```
identify(res.qq)
```

Pour créer un autre tableau de données *ne contenant pas* l'individus 103 du tableau de données initial :

```
donnees2 = donnees[-103,]
dim(donnees2)
```

```
## [1] 103   3
```

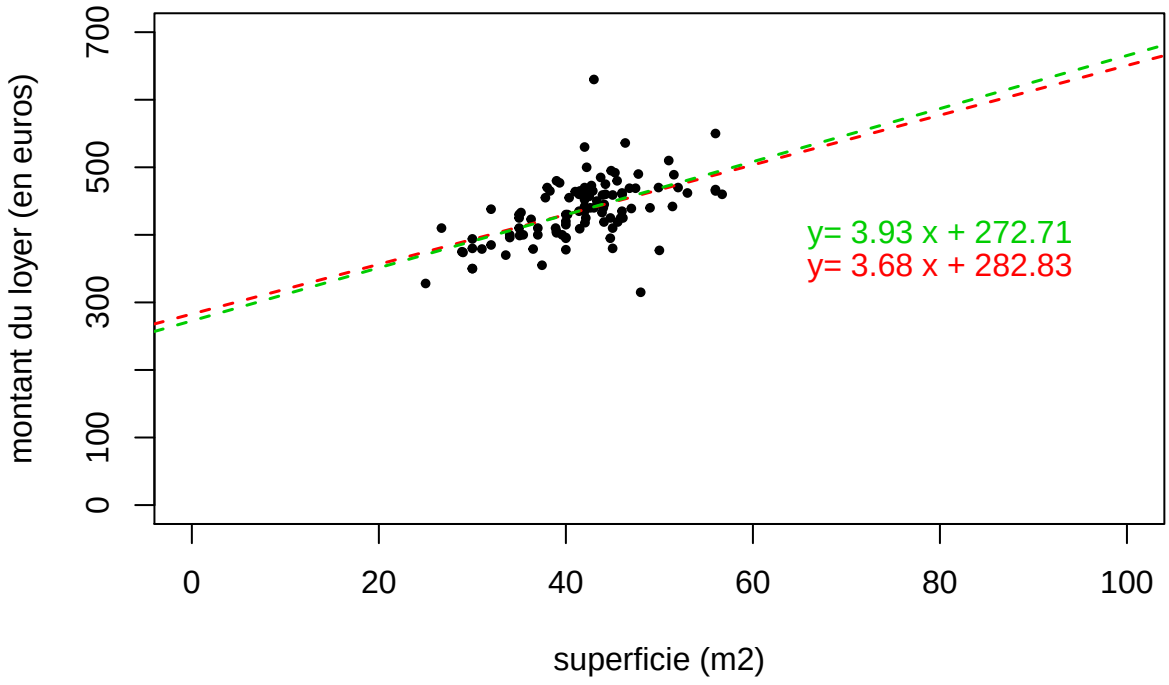
Modèle de régression linéaire simple sur ce jeu de données et représentation graphique:

```
res.lm2 = lm(formula=LOYERCC~SUPERFICIE,data=donnees2)
summary(res.lm2)
```

```
##
## Call:
## lm(formula = LOYERCC ~ SUPERFICIE, data = donnees2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -146.210  -23.177   -1.576   23.091  188.424
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 272.7150    25.4753  10.705  < 2e-16 ***
## SUPERFICIE    3.9270     0.6058   6.482 3.36e-09 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.81 on 101 degrees of freedom
## Multiple R-squared:  0.2938, Adjusted R-squared:  0.2868
## F-statistic: 42.01 on 1 and 101 DF,  p-value: 3.362e-09
```

```r
# Nuage de points avec la superficie en abscisse et le montant du loyer en ordonnée

plot(x=donnees2$SUPERFICIE, y=donnees2$LOYERCC, pch=20, cex=0.8, col=1,xlim=c(0,100), ylim=c(0,700), xla

# ajout de la doite de régression du modèle 1, en rouge (col=2) et en pointillés (lty=2)
abline(a=res.lm$coefficients[1], b= res.lm$coefficients[2],col=2, lty=2, lwd=1.5)
# pour écrire l'équation sur le graphique
text(x = 80, y= 350, labels=paste("y=",round(res.lm$coefficients[2],2),"x +",round(res.lm$coefficients[

# ajout de la doite de régression du modèle 2, en vert (col=3) et en pointillés (lty=2)
abline(a=res.lm2$coefficients[1], b= res.lm2$coefficients[2],col=3, lty=2, lwd=1.5)
# pour écrire l'équation sur le graphique
text(x = 80, y=400, labels=paste("y=",round(res.lm2$coefficients[2],2),"x +",round(res.lm2$coefficients
```

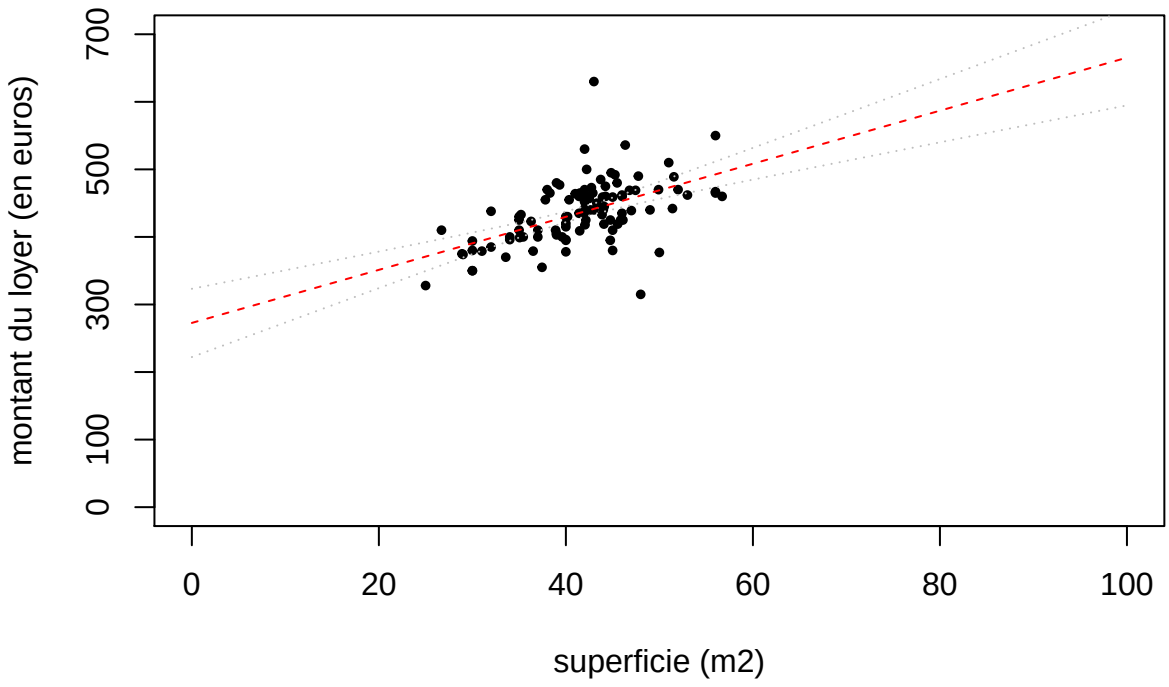



```r
# une grille de calcul pour l'IC
g = seq(from = 0,to = 100,by = 1)
grille=data.frame(SUPERFICIE=g)
# calcul de l'IC pour tous les points de la grille
# NB: la grille doit etre un data.frame, et la colonne doit avoir le meme nom que la variable dans les
ICdte = predict(res.lm2,new=grille,interval = "confidence",level=0.95)
```

```
ICdte[1:10,]
```

```
##         fit      lwr      upr
## 1  272.7150 222.1789 323.2511
## 2  276.6420 227.2930 325.9909
## 3  280.5690 232.4064 328.7315
## 4  284.4959 237.5191 331.4728
## 5  288.4229 242.6308 334.2150
## 6  292.3499 247.7417 336.9581
## 7  296.2769 252.8516 339.7022
## 8  300.2039 257.9604 342.4474
## 9  304.1309 263.0681 345.1937
## 10 308.0579 268.1745 347.9413
```

```
# Nuage de points avec la superficie en abscisse et le montant du loyer en ordonnée
plot(x=donnees2$SUPERFICIE, y=donnees2$LOYERCC, pch=20, cex=0.8, col=1,xlim=c(0,100), ylim=c(0,700), xla
# ajout de la droite de régression en rouge et en pointillés (lty=2)
# et de l'IC en gris et en pointillés (lty=3)
lines(grille$SUPERFICIE,ICdte[,"fit"],lty=2,col="red")
lines(grille$SUPERFICIE,ICdte[,"lwr"],lty=3,col="grey")
lines(grille$SUPERFICIE,ICdte[,"upr"],lty=3,col="grey")
```



Prédiction

A partir d'une nouvelle observation *x0*, on utilise les estimations pour prévoir la valeur de la variable réponse correspondante.

```r
x0=50
x0 = as.data.frame(x0)
colnames(x0) <- "SUPERFICIE"
predict(object=res.lm2, newdata=x0 )
```

```
##        1
## 469.0645
```

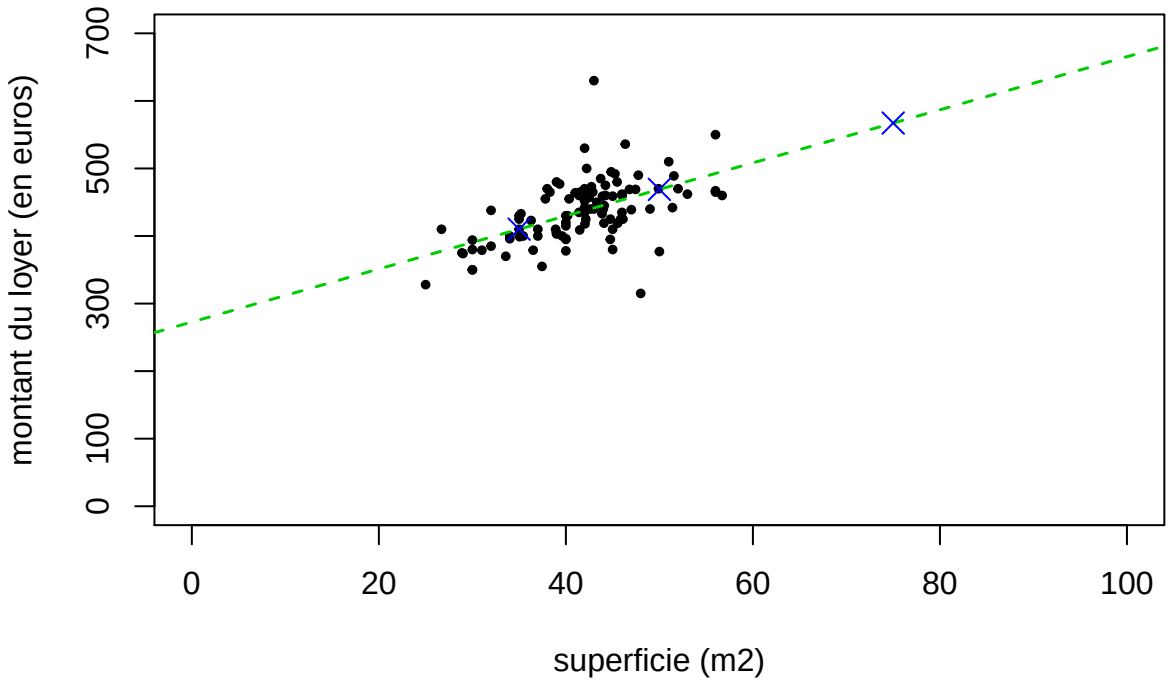
```r
# a.x0 + b :
# res.lm2$coefficients[2]*x0 + res.lm2$coefficients[1]
```

On remarque que l'argument *newdata* de la fonction **predict()** est de type *data.frame*, et doit avoir le même nom de colonne pour la variable explicative (ici, *SUPERFICIE*).

```r
x0=c(35,50,75)
x0 = as.data.frame(x0)
colnames(x0) <- "SUPERFICIE"
res.pred= predict(object=res.lm2, newdata=x0 )
res.pred
```

```
##        1        2        3
## 410.1596 469.0645 567.2392
```

```r
# Nuage de points avec la superficie en abscisse et le montant du loyer en ordonnée
plot(x=donnees2$SUPERFICIE, y=donnees2$LOYERCC, pch=20, cex=0.8, col=1,xlim=c(0,100), ylim=c(0,700), xla

# ajout de la doite de régression, en vert (col=3) et en pointillés (lty=2)
abline(a=res.lm2$coefficients[1], b= res.lm2$coefficients[2],col=3, lty=2, lwd=1.5)

# points prédits en bleu (col=4)
points(x=x0[,1], y=res.pred, pch=4, cex=1.5, col=4)
```


montant du loyer (en euros)
0
100
300
500
700
0
20
40
60
80
100
superficie (m2)