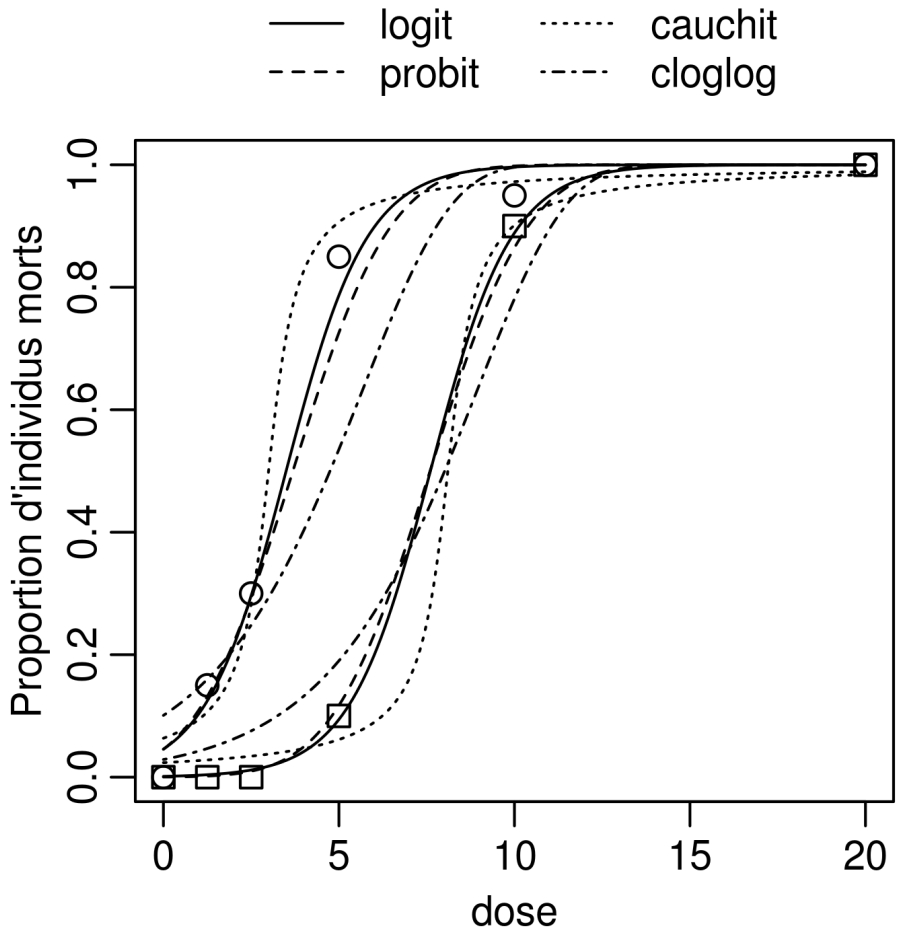
GLM : Generalized Linear Models



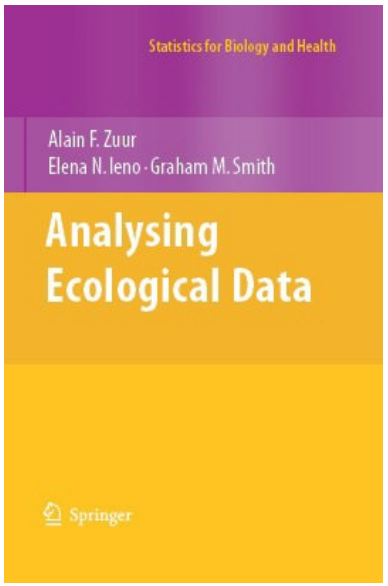
G. San Martin

gilles.sanmartin@gmail.com
Centre Wallon de Recherche Agronomique

cra-w

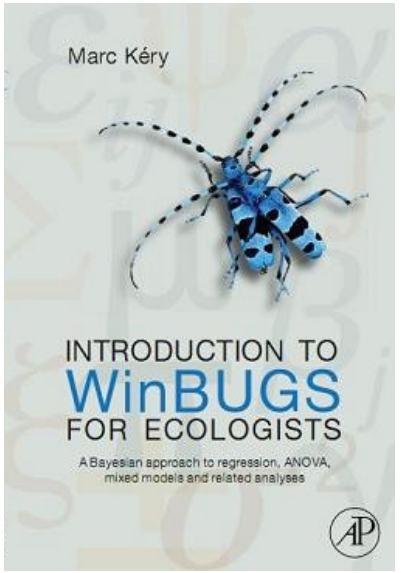
Quelques livres

Formation principalement basée sur 4 livres.
Tous ont une approche unifiée "moderne" (GLM) et certains font le lien avec les stats classiques

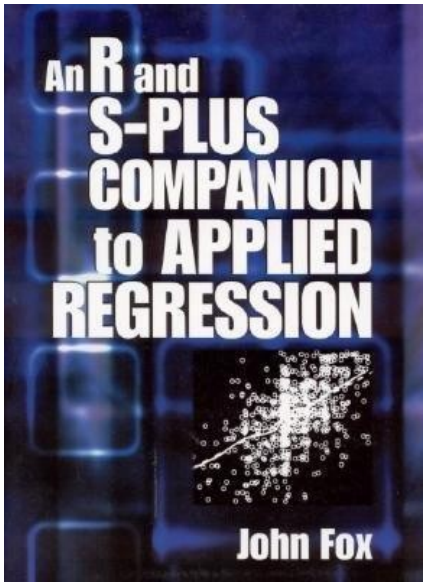


Zuur et al.
Le plus appliqué.
Pour être opérationnel le plus vite possible

Gelman & Hill
Le plus détaillé tout en restant très accessible



Kery
Analyse classique et Bayésienne de Jeux de données simulés



Fox
Le plus simple aborde des problèmes rarement abordés

Objectifs

Qu'est-ce qu'un GLM, à quoi çà sert ?

Illustrer :

Exemples des principaux types de GLM

La plupart des tests statistiques classiques sont des cas particuliers de GLM

Insister sur :

Comment interpréter les sorties du logiciel ?

Comment en faire une représentation graphique ?

Quelles sont les conditions d'application, comment les vérifier et comment solutionner les problèmes ?

GLMs : qu'est-ce que c'est ?

Régression établissant le lien entre une variable à expliquer/prédire et une ou plusieurs variables descriptives/explicatives

Paramètres à estimer

Relation linéaire additive

$$y = \alpha + \beta_1 * x_1 + \beta_2 * x_2 + \ldots + \epsilon$$

Variable "dépendante"
a expliquer :

données
continues
binaires
comptage
%

Variables "explicatives"
Quantitatives
Qualitatives
Interactions

Résidus = Erreur
parties de la variabilité
que l'on ne peut pas
expliquer avec ces variables

Distrib Normale
Distrib de Poisson
Distrib Binomiale
etc...

GLMs : qu'est-ce que c'est ?

(General) Linear Model

y = variable continue à distribution à peu près normale

Résidus : distribution Normale

Méthode d'estimation = Sum of Squares

R : lm() - SAS : PROC GLM

Generalized Linear Model

y = variable continue ou de comptage ou binaire ou %,...

Résidus : distribution Normale ou Poisson ou Binomiale,...

Méthode d'estimation = Maximum Likelihood

R : glm() - SAS : PROC GENMOD

Autres abréviations :

GLIM - GLZ

GLMs : à quoi çà sert ?

Peuvent s'utiliser pour un très grand nombre de questions :

Prédiction
Prédire y en fonction d'une série de variables explicatives x1, x2, ...

Contrôle statistique
Relation entre y et x1 après avoir enlevé l'effet de x2
(= en contrôlant x2 = indépendamment de x2)

Sélection de modèles
Quelles sont les variables les plus importantes pour prédire y ?

Interactions
Est-ce que l'effet de x1 sur y dépend de x2

Partition de la variance
Quelle % de la variation de y est expliqué par x1 seul, x2 seul et par x1 et x2 en commun ?

Non indépendance - designs complexes
Prendre en compte la non indépendance des données (données groupées, mesures répétées, corrélation spatiale, temporelle, etc...)
(ea grâce aux modèles mixtes)

Programme

Part 1 : (General) Linear Model (LM)

On va s'intéresser principalement aux variables explicatives
Y sera toujours une variable quantitative continue approximativement normale

1 X quantitatif = régression linéaire simple
1 X qualitatif à 2 niveaux = test de student
1 X qualitatif à n niveaux = ANOVA
Comparaisons multiples
Plusieurs X quantitatifs = régression multiple
Plusieurs X quantitatifs ou qualitatifs = ANCOVA
Interactions
Relations non linéaires

Programme

Part 2 : Generalized Linear Model (GLM)

La partie concernant les variables explicatives change peu.
On va s'intéresser aux Y et aux distributions des résidus

Y = données de comptage (distribution de Poisson)
Y binaire (régression logistique - distribution binomiale)
Y = % (distribution binomiale)
Tables de contingence (distribution de Poisson)

Part 1 : General Linear Model

Régression linéaire simple : 1 x quantitatif

Il s'agit ici de trouver la meilleure droite passant par un nuage de points

Exemple : relation entre les doses de fertilisants et la production de tomates

Concepts à assimiler :

Pente, intercept, résidus

Interprétation géométrique des paramètres

R^2, % de variance expliquée

Valeurs prédites

Influence des valeurs extrêmes

Utilité de centrer les variables explicatives

Régression linéaire simple : 1 x quantitatif

Représentation algébrique du modèle :

$$y = \alpha + \beta * x + \epsilon$$

$$\epsilon \sim Normale(0, \sigma^2)$$

Notation équivalente :

$$y \sim Normale(\alpha + \beta * x, \sigma^2)$$

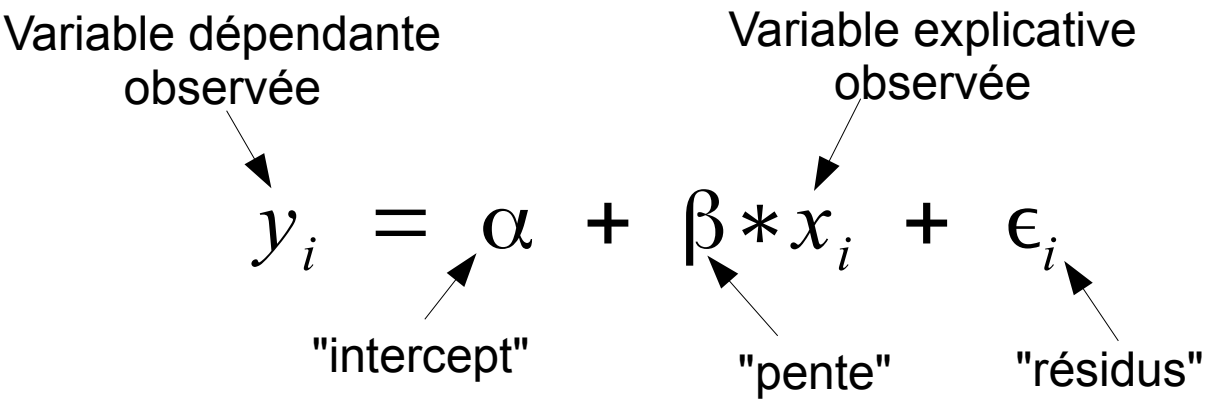
Avec :

$$\hat{y} = \alpha + \beta * x$$

$$\epsilon = y - \hat{y}$$

Régression linéaire simple : 1 x quantitatif

Représentation algébrique du modèle :



Intercept :
valeur prédite de y quand x = 0

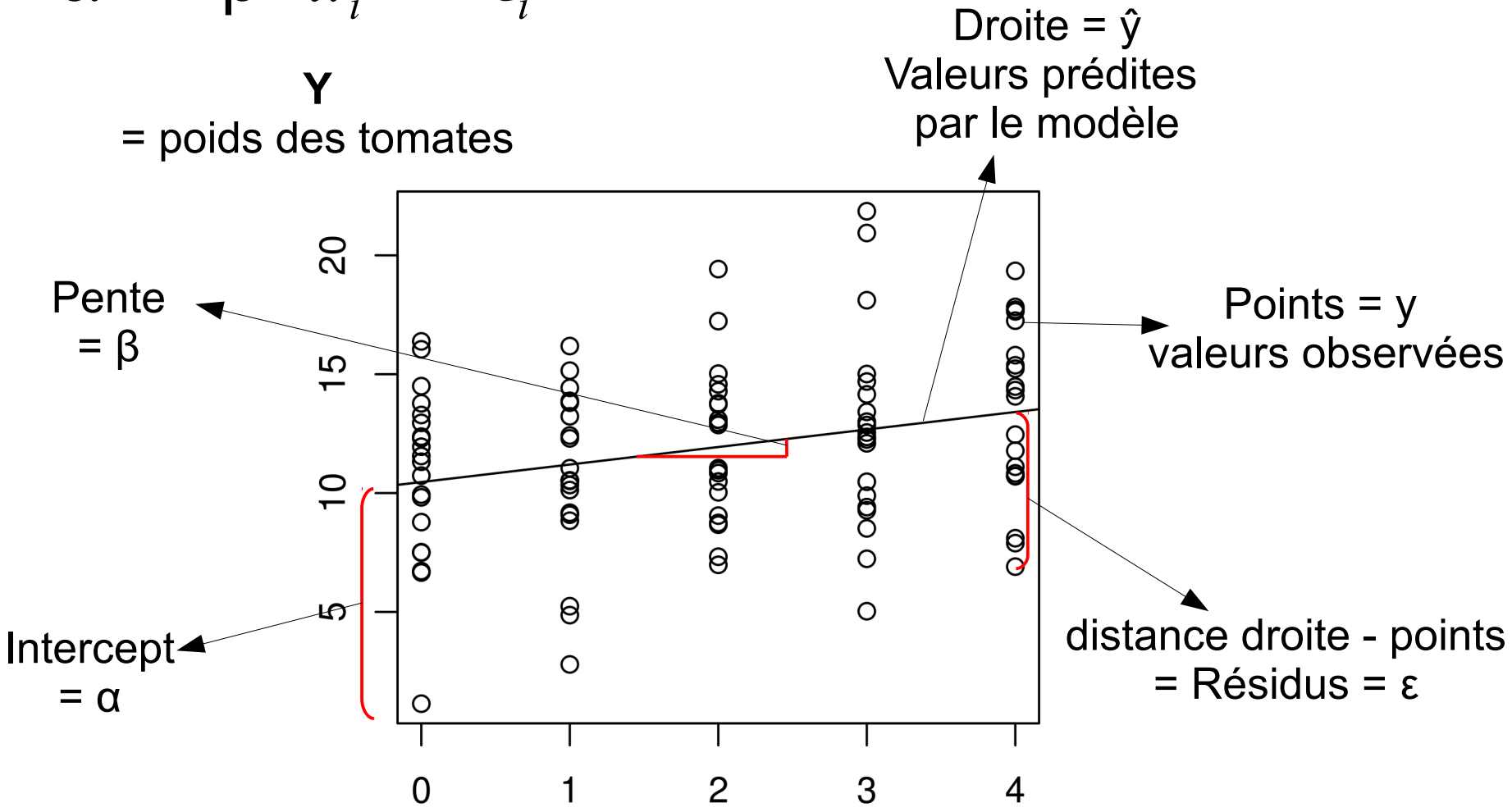
Pente (= "Slope") :
de combien augmente y quand x augmente de une unité ?

Résidus :
différence entre les valeurs observées et les valeurs prédites

Régression linéaire simple : 1 x quantitatif

Représentation géométrique du modèle :

$$y_i = \alpha + \beta * x_i + \epsilon_i$$



La droite optimale est celle qui minimise les résidus
(ie la somme des carrés des résidus)

Régression linéaire simple : 1 x quantitatif

Représentation algébrique du modèle :

"Valeurs prédites par le modèle"

$$\hat{y}_i = \alpha + \beta * x_i$$

Les résidus sont la différence entre les valeurs observées et les valeurs prédites

$$\epsilon_i = y_i - \hat{y}_i$$

Les résidus suivent une distribution Normale de moyenne 0 et de variance sigma²

$$\epsilon \sim Normale\left(0, \sigma^2\right)$$

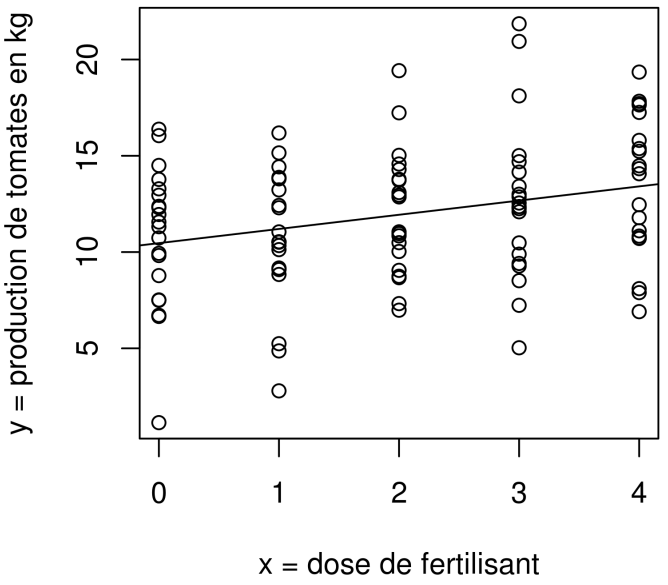
3 paramètres doivent être estimés : l'intercept, la pente et la variance des résidus

Régression linéaire simple : 1 x quantitatif

Exemple : production tomates ~ dose fertilisant

On génère des données (n=100) pour avoir un intercept (alpha) de 10 kg, une pente (beta) de 0.75 kg et une variance des résidus (sigmasq) de 16 kg².
On a 5 doses de fertilisant (0-4) et 20 observations par dose.

```
> alpha <- 10
> beta <- 0.75
> sigmasq <- 16
>
> n <- 100
> x <- rep(0:4, each = n/5)
>
> set.seed(1)
> y <- alpha + beta * x +
    rnorm(n = n, mean = 0, sd = sqrt(sigmasq))

> # autre manière de faire strictement identique :
> set.seed(1)
> y <- rnorm(n = n, mean = (alpha + beta * x) ,
    sd = sqrt(sigmasq))
```



Régression linéaire simple : 1 x quantitatif

```
> mod <- lm(y ~ x)
> summary(mod)

Call:
lm(formula = y ~ x)

Residuals:
    Min      1Q  Median      3Q     Max
-9.3209 -2.4158  0.0329  2.3406  9.1842

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  10.4621     0.6254  16.727  < 2e-16 ***
x             0.7367     0.2553   2.885  0.00481 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.611 on 98 degrees of freedom
Multiple R-squared:  0.0783,  Adjusted R-squared:  0.0689
F-statistic: 8.325 on 1 and 98 DF,  p-value: 0.004808
```

Estimation du modèle

Résumé du modèle

Intercept (alpha) estimé

Pente (beta) estimée

erreur standard des résidus

Régression linéaire simple : 1 x quantitatif

Interprétation

Quand on ne met aucun fertilisant (x=0) on estime que la production moyenne de tomates est de 10.4621 kg

Quand la dose de fertilisant augmente d'une unité, la production de tomates augmente de 0.7367 kg (dans la limite des doses testées)

On peut prédire la production de tomate en fonction de la dose de fertilisant. Par exemple on estime que pour une dose de 1.42 unités de fertilisant, on aura en moyenne une production de 10.4621 + 0.7367 * 1.42 = 11.51 kg de tomates

Autour de ces valeurs prédites, les résidus ont un écart-type estimé (erreur standard) de 3.611 kg
Attention il ne s'agit PAS de l'erreur standard des valeurs prédites !

```
> mod <- lm(y ~ x)
> summary(mod)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  10.4621     0.6254  16.727  < 2e-16 ***
x             0.7367     0.2553   2.885  0.00481 **
(...)
Residual standard error: 3.611 on 98 degrees of freedom
Multiple R-squared:  0.0783,  Adjusted R-squared:  0.0689
F-statistic: 8.325 on 1 and 98 DF,  p-value: 0.004808
```

Régression linéaire simple : 1 x quantitatif

Inférence pour les (General) Linear Models

On teste toujours par défaut l'hypothèse nulle que chaque coefficient est = 0

Deux méthodes paramétriques principales

Dans ce cas-ci elles sont strictement équivalentes mais ce ne sera pas toujours le cas !

Méthode 1 :

coefficient / erreur standard suit une loi de Student
à n-k degrés de liberté (k = nombres de paramètres)

```
> summary(mod)
(...)
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  10.4621     0.6254  16.727  < 2e-16 ***
x             0.7367     0.2553   2.885  0.00481 **
```

Estimate +- 2*Std.Error --> intervales de confiance approximatifs
utiliser les quantiles de la loi de student pour avoir des valeurs exactes
ou `confint(mod)`

Régression linéaire simple : 1 x quantitatif

Méthode 2 : Comparaison de modèles emboîtés

Pour tester si la pente est significativement différente de 0, on compare un modèle complet y ~ a + bx avec un modèle avec une pente nulle y ~a.
La statistique utilisée ici est liée au carré des résidus (RSS : Residuals Sum of Squares) et suit une distribution de Fisher (F)

```
> mod0 <- lm(y ~ 1)
> mod1 <- lm(y ~ x)
> anova(mod0, mod1)

Analysis of Variance Table

Model 1: y ~ 1
Model 2: y ~ x
  Res.Df    RSS Df Sum of Sq      F   Pr(>F)
1     99 1386.4
2     98 1277.9  1    108.56 8.3253 0.004808 **
```

Modèle avec juste 1 intercept

Modèle avec intercept (implicite) et pente

$$F = \frac{\frac{RSS_0 - RSS_1}{df_0 - df_1}}{\frac{RSS_1}{df_1}}$$

<u>Attention !!</u> : on peut aussi procéder comme suit. Ici les résultats sont identiques. Mais dans de nombreux cas cette méthode ne donne pas le résultat escompté !!!! --> très déconseillée

```
> anova(mod1)
          Df  Sum Sq Mean Sq F value   Pr(>F)
x          1  108.56  108.56  8.3253 0.004808 **
Residuals 98 1277.88   13.04
```

Régression linéaire simple : 1 x quantitatif

Méthode 2 : Comparaison de modèles emboîtés

Si on veut tester si l'intercept est significativement différent de 0 (on a rarement besoin de tester cette hypothèse ...) on compare un modèle complet avec un modèle sans intercept

```
> mod0 <- lm(y ~ -1 + x)
> mod1 <- lm(y ~ x)
> anova(mod0, mod1)
Analysis of Variance Table

Model 1: y ~ -1 + x
Model 2: y ~ x
  Res.Df    RSS Df Sum of Sq     F    Pr(>F)
1     99 4926.4
2     98 1277.9  1    3648.5 279.8 < 2.2e-16 ***
---
```

Modèle sans intercept
--> on force la droite à passer par 0

Régression linéaire simple : 1 x quantitatif

Méthode 2 : Comparaison de modèles emboîtés

Exemple de calcul à la main

```
> mod0 <- lm(y ~ 1)
> mod1 <- lm(y ~ x)
> anova(mod0, mod1)
Analysis of Variance Table

Model 1: y ~ 1
Model 2: y ~ x
  Res.Df    RSS Df Sum of Sq      F   Pr(>F)
1     99 1386.4
2     98 1277.9  1    108.56 8.3253 0.004808 **
---

>
>
> (RSS0 <- sum(resid(mod0)^2))
[1] 1386.435
> (RSS1 <- sum(resid(mod1)^2))
[1] 1277.876
> (F <- ((RSS0 - RSS1) / (99-98)) / (RSS1/98))
[1] 8.325333
> pf(q= F, df1=1, df2= 98, lower.tail=FALSE)
[1] 0.004808336
```

Résidus du modèle

$df_0 - df_1$ df_1

$df_0 - df_1$ df_1

$$F = \frac{\frac{RSS_0 - RSS_1}{df_0 - df_1}}{\frac{RSS_1}{df_1}}$$

df =
nbre de données -
nombre de paramètres + 1
= n-k+1

Pex pour modèle H1 :
k = intercept, slope et sigma
= 3

Pex pour modèle H0 :
k = intercept et sigma
= 2

Régression linéaire simple : 1 x quantitatif

Extraire des informations du modèle

R permet de récupérer n'importe quelle valeur calculée afin de la manipuler

```
> # coefficients
> coef(mod)
(Intercept)           x
 10.4620597   0.7367449

> # Intervalles de confiance
> confint(mod)
                2.5 %    97.5 %
(Intercept) 9.2208763 11.703243
x           0.2300339  1.243456

> y[1:5]
[1]  7.494185 10.734573  6.657486 16.381123 11.318031

> # Valeurs prédites pour chaque x observé
> fitted(mod)[1:5]
       1        2        3        4        5
10.46206 10.46206 10.46206 10.46206 10.46206

> # Résidus pour chaque valeur observée
> resid(mod)[1:5]
         1          2          3          4          5
-2.9678749  0.2725136 -3.8045741  5.9190636  0.8559714

> # On peut vérifier la manière de calculer les résidus
> y[1:5] - fitted(mod)[1:5]
         1          2          3          4          5
-2.9678749  0.2725136 -3.8045741  5.9190636  0.8559714
```

Régression linéaire simple : 1 x quantitatif

Extraire des informations du modèle

Utiliser `str()` pour visualiser la structure d'un objet et en extraire ce que vous voulez
L'objet retourné par `summary(mod)` contient des informations supplémentaires

```
> str(summary(mod))
List of 11
 $ call          : language lm(formula = y ~ x)
 $ terms         :Classes 'terms', 'formula' length 3 y ~ x
  .. ..- attr(*, "variables")= language list(y, x)
  .. ..- attr(*, "factors")= int [1:2, 1] 0 1
  (…)

> summary(mod)$coefficients
             Estimate Std. Error   t value     Pr(>|t|)
(Intercept) 10.4620597  0.6254491 16.727277 1.790848e-30
x            0.7367449  0.2553385  2.885365 4.808336e-03

> summary(mod)$coefficients[,"Std. Error"]
(Intercept)           x
  0.6254491   0.2553385

> summary(mod)$sigma
[1] 3.611032

> summary(mod)$r.square
[1] 0.07830056
```

Régression linéaire simple : 1 x quantitatif

Extraire des informations du modèle

Par exemple on peut recalculer les p valeurs des coefficients et leurs intervalles de confiance donnés par R

```
> summary(mod)
(...)
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  10.4621     0.6254  16.727  < 2e-16 ***
x             0.7367     0.2553   2.885  0.00481 **


> (tstat <- coef(mod) / summary(mod)$coefficients[,"Std. Error"])
(Intercept)           x
  16.727277    2.885365
> 2*pt(q=tstat, df = 97, lower.tail = FALSE)
 (Intercept)            x
2.440669e-30 4.818078e-03


> confint(mod)
                2.5 %    97.5 %
(Intercept) 9.2208763 11.703243
x           0.2300339  1.243456
> coef(mod) + qt(0.025, df= 98) * summary(mod)$coefficients[,"Std. Error"]
(Intercept)           x
  9.2208763   0.2300339
> coef(mod) + qt(0.975, df= 98) * summary(mod)$coefficients[,"Std. Error"]
(Intercept)           x
  11.703243    1.243456
```

Régression linéaire simple : 1 x quantitatif

Le coefficient de détermination : R^2

Le R^2 représente le % de variance expliquée par le modèle

Un $R^2 = 1$ signifie que le modèle prédit parfaitement les données, les résidus sont nuls, tous les points sont parfaitement alignés sur la droite (ou la courbe) dont la pente doit être non nulle.

Un $R^2 = 0$ signifie que le modèle n'a aucun pouvoir prédictif

Dans beaucoup de cas, le R^2 est égal au carré du coefficient de corrélation R

```
> summary(mod)$r.squared
[1] 0.07830056
> cor(y, x)^2
[1] 0.07830056
```

Régression linéaire simple : 1 x quantitatif

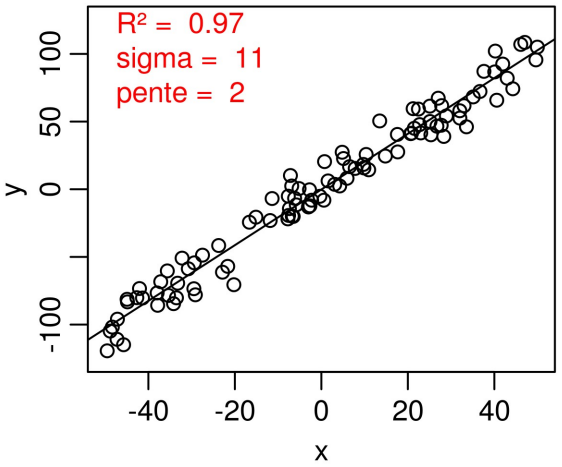
Le coefficient de détermination : R^2

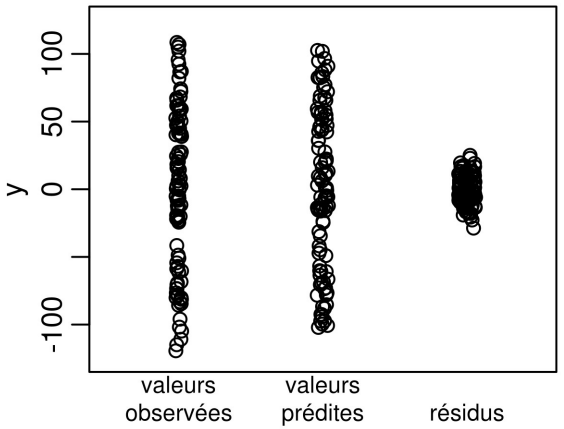
Il peut se calculer donc simplement comme le rapport entre la variance des valeurs prédites (variabilité expliquée par le modèle) et de la variance des valeurs observées (variabilité totale)

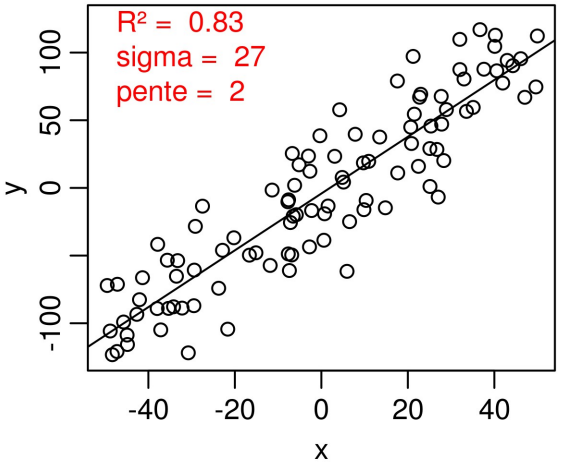
Ou encore : 1 - la variance des résidus (variabilité pas expliquée par le modèle) / la variance des valeurs observées

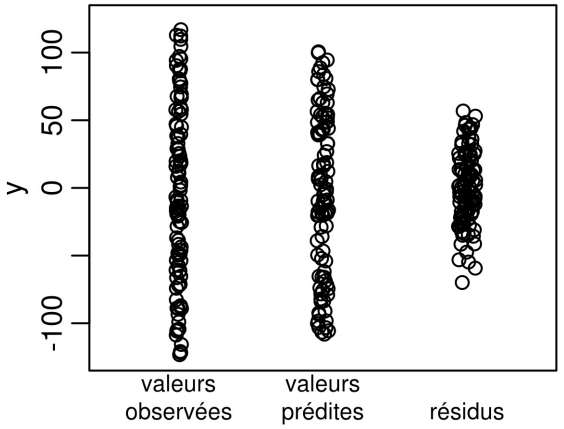
```
> summary(mod)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  10.4621     0.6254  16.727  < 2e-16 ***
x             0.7367     0.2553   2.885  0.00481 **

Residual standard error: 3.611 on 98 degrees of freedom
Multiple R-squared:  0.0783,  Adjusted R-squared:  0.0689
F-statistic: 8.325 on 1 and 98 DF,  p-value: 0.004808

> var(predict(mod))/ var(y)
[1] 0.07830056

> 1-(var(resid(mod))/ var(y))
[1] 0.07830056
```

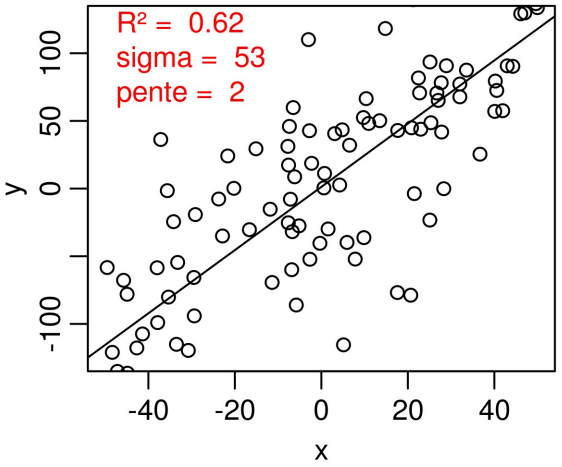
NB : en réalité le R^2 est calculé légèrement différemment :
1-[(SSres/n) / (SSy/n)]
ce qui revient au même

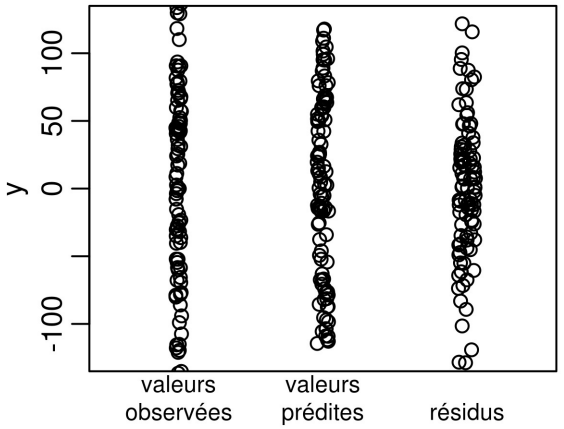








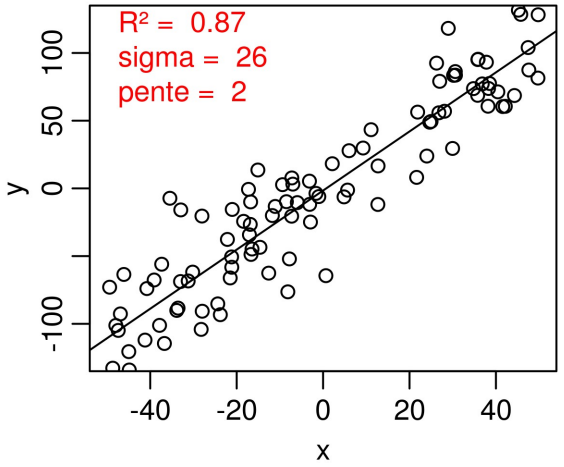


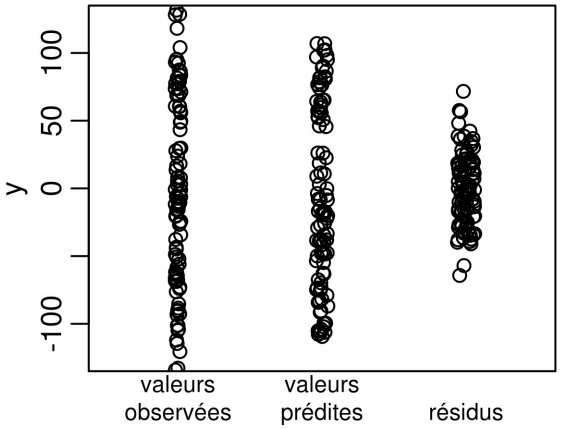


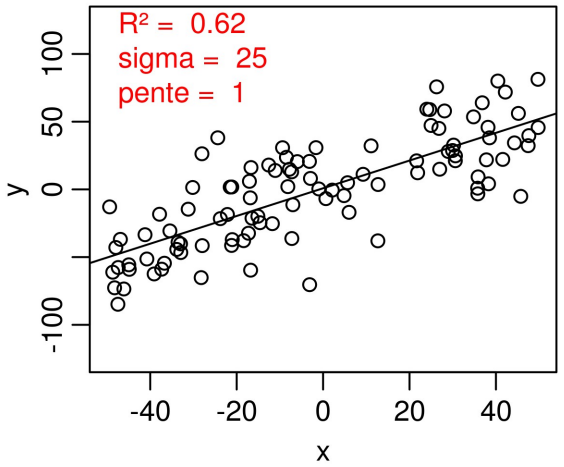
La pente est identique
La variance résiduelle change

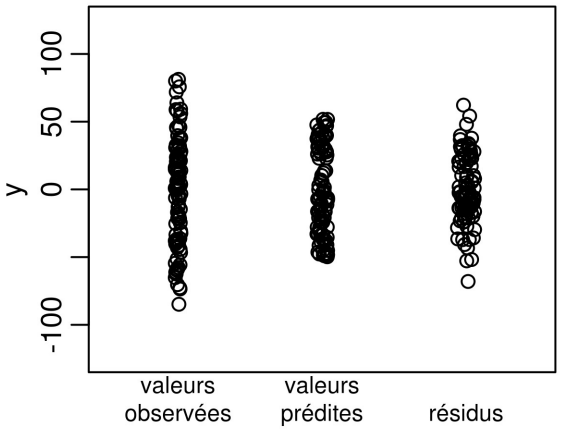
NB, la variance des valeurs observées (= Variance totale) est égale à la somme de la variance des résidus et de la variance des valeurs prédites

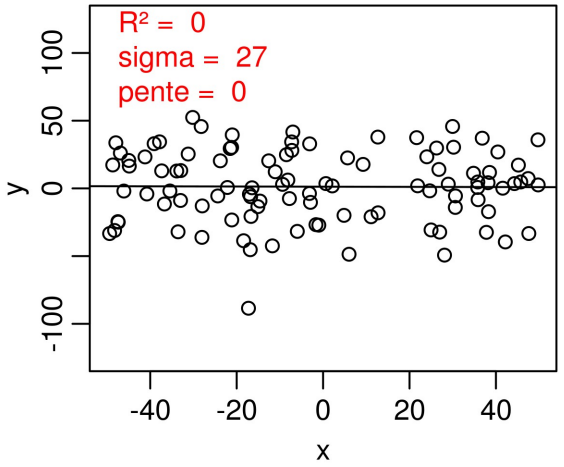
```
> var(y)
[1] 14.00439
> var(predict(mod)) + var(resid(mod))
[1] 14.00439
```

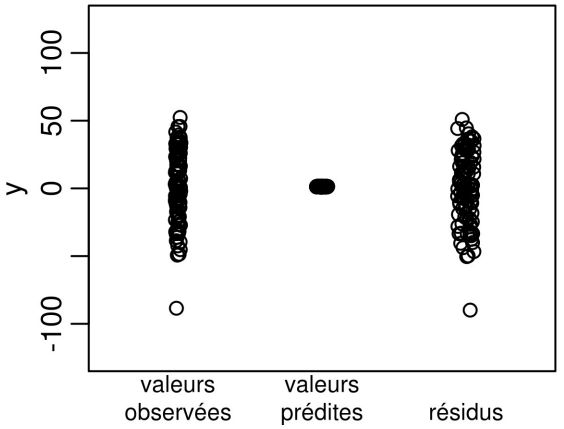




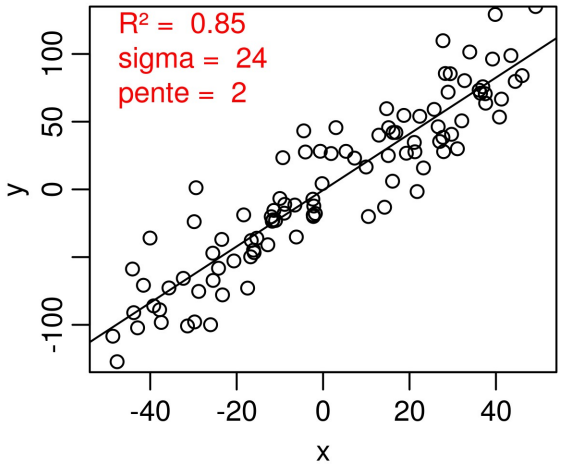


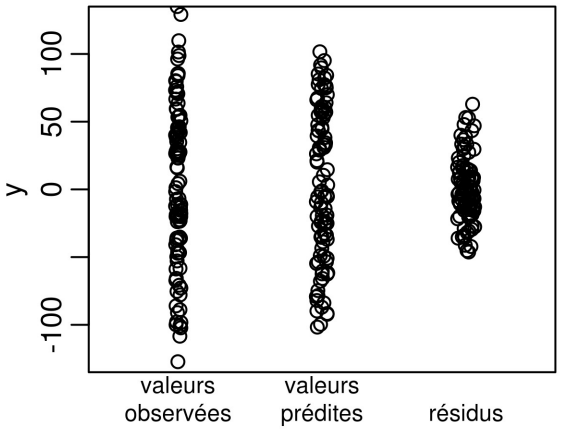


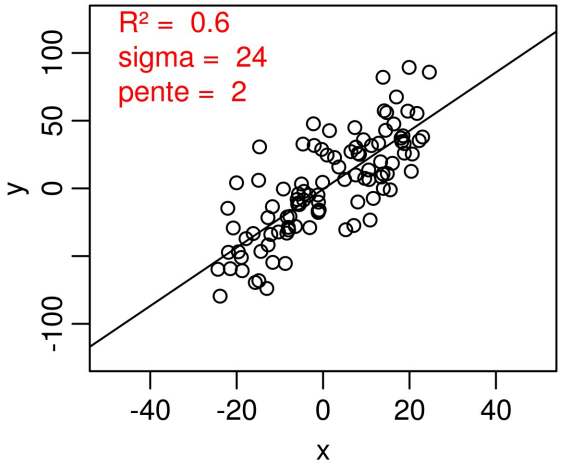


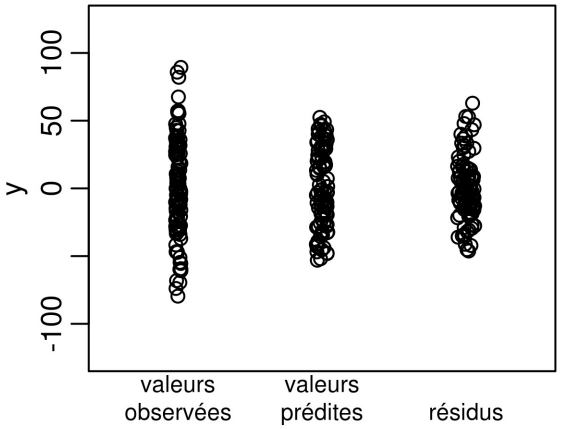


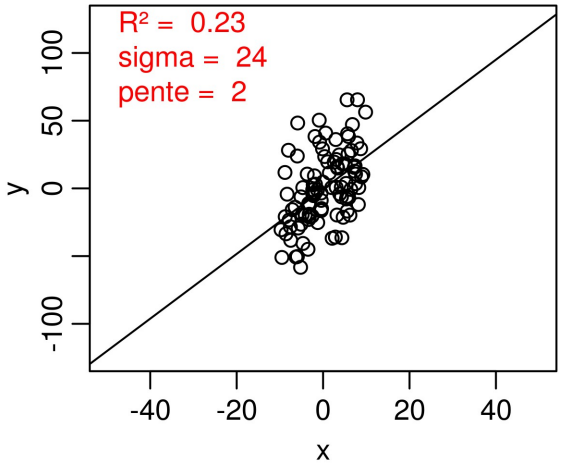
La pente change
La variance résiduelle est identique

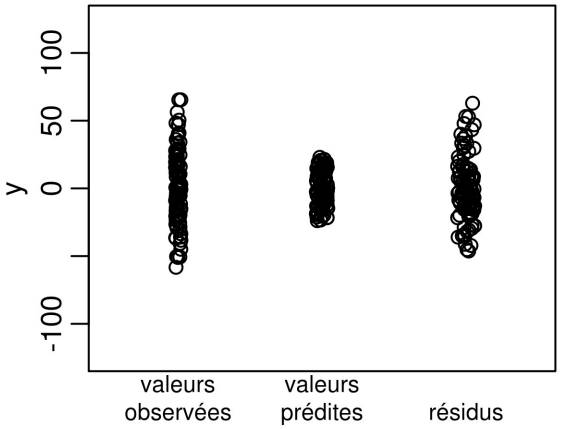












La pente est identique
La variance résiduelle est identique

Mais la gamme de valeurs de x change

Régression linéaire simple : 1 x quantitatif

Représentation graphique et valeurs prédites

```
> # on crée un nouveau jeu de données avec moins d'observations
> n <- 15
> x <- rep(0:4, each = n/5)
> set.seed(1)
> y <- alpha + beta * x + rnorm(n = n, mean = 0, sd = sqrt(sigmasq))
> mod <- lm(y ~ x)
```

Dans le cas de la régression linéaire simple, la représentation graphique est très facile

Plot les points

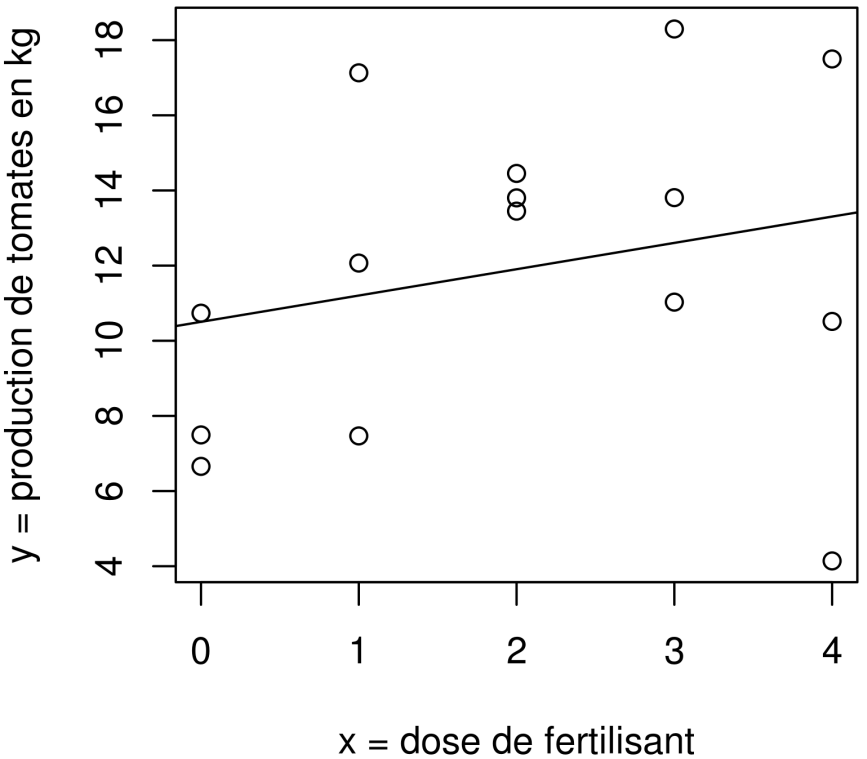
```
> plot( y ~ x, xlab= "x = dose de fertilisant",
    ylab = "y = production de tomates en kg")

> abline(mod)
```

Ajoute une ligne droite basée sur le modèle estimé

MAIS :

1) il y a de nombreux cas où `abline()` ne fonctionne pas

2) On aimerait afficher les erreurs standard des prédictions sur le graphique



Régression linéaire simple : 1 x quantitatif

Représentation graphique et valeurs prédites

La fonction `predict()` permet de prédire y pour des nouvelles valeurs de x (appelées xnew ici) et d'obtenir les erreurs standard

```
> newx <- data.frame(x= seq(0, 4, 0.01))

> y_hat <- predict(object = mod, newdata= newx, se.fit = TRUE)

> y_hat
$fit
       1        2        3        4        5        6        7
10.50290 10.50990 10.51690 10.52391 10.53091 10.53791 10.54491
      (...)

$se.fit
       1        2        3        4        5        6        7
1.888927 1.882636 1.876356 1.870086 1.863827 1.857579 1.851341
(...)

$df
[1] 13

$residual.scale
[1] 4.22377
```

Nouveaux x de 0 à 4 par pas de 0.01

Valeurs prédites

Erreur Standard des valeurs prédites

Régression linéaire simple : 1 x quantitatif

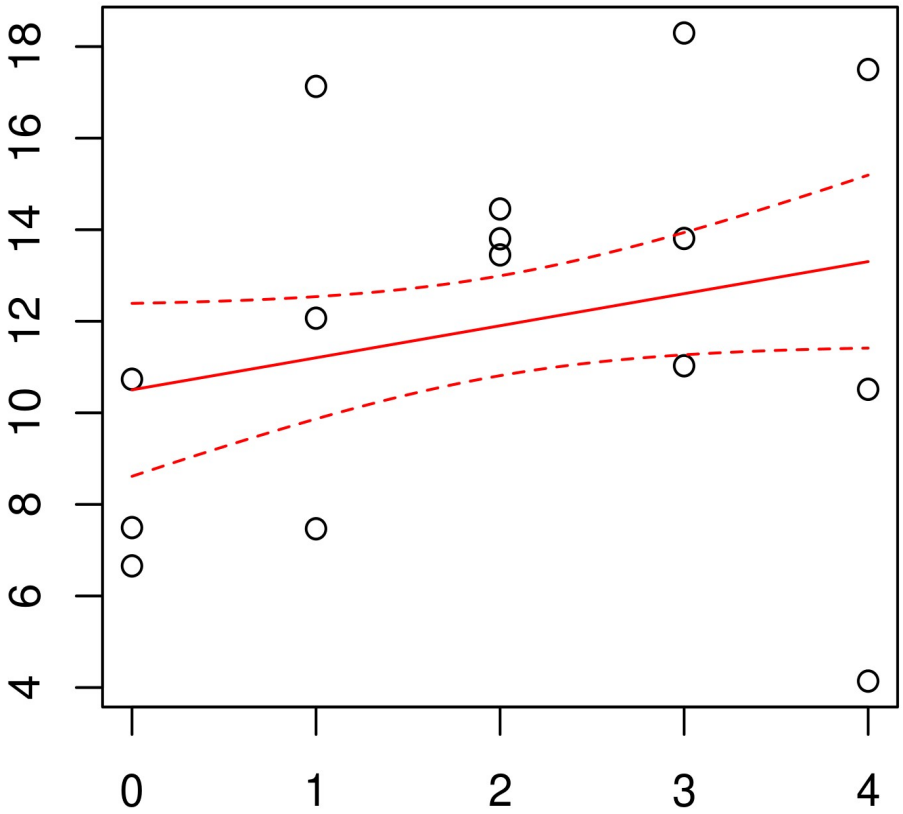
Représentation graphique et valeurs prédites

```
plot(y ~ x)

lines(y_hat$fit ~ newx$x, col = "red")

lines(y_hat$fit + y_hat$se.fit ~ newx$x,
    lty = 2, col = "red")
lines(y_hat$fit - y_hat$se.fit ~ newx$x,
    lty = 2, col = "red")
```

Ajoute une courbe reliant toutes les valeurs prédites

tracent les courbes correspondant aux valeurs prédites +- Erreur Standard

`lty = 2` : type de lignes = tirets

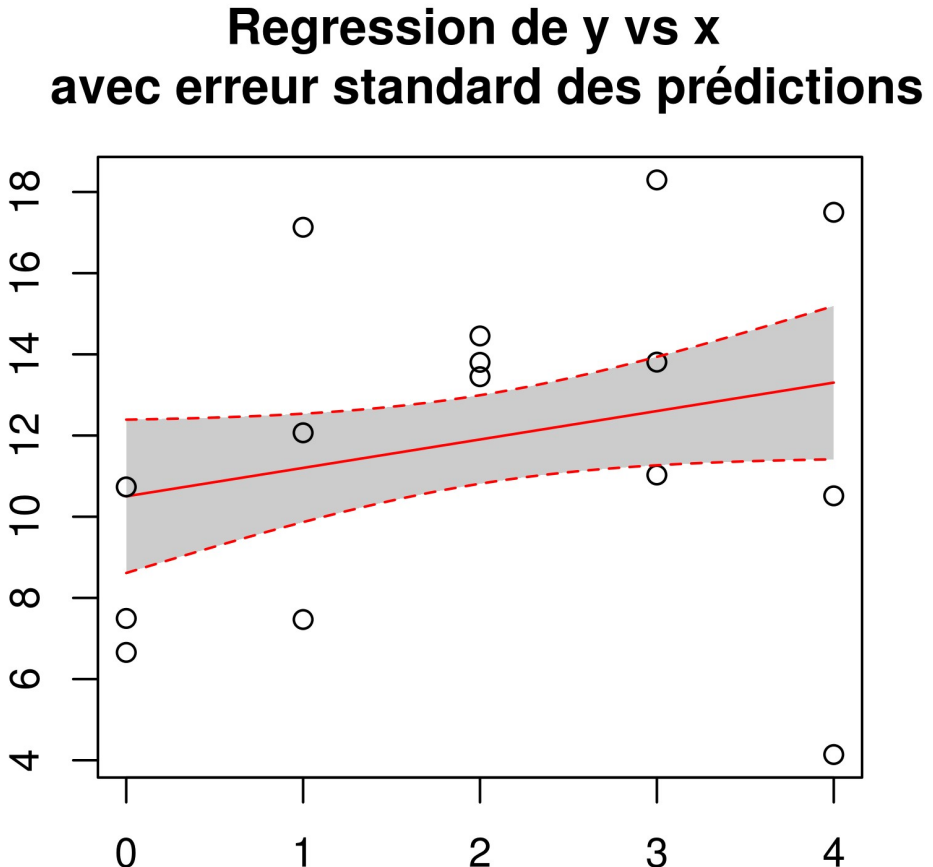


Régression linéaire simple : 1 x quantitatif

Représentation graphique et valeurs prédites

Représentation légèrement différente avec une surface grisée représentant les valeurs prédites +- erreur standard

```
plot(y ~ x, type = "n",
    main = "Regression de y vs x \n avec erreur standard des prédictions")

polygon(x = c(newx$x, rev(newx$x)),
        y = c(y_hat$fit + y_hat$se.fit, rev(y_hat$fit - y_hat$se.fit)),
        col = "grey80", border = NA)

lines(y_hat$fit ~ newx$x, col = "red")
points(y~x)

lines(y_hat$fit + y_hat$se.fit ~ newx$x,
    lty = 2, col = "red")
lines(y_hat$fit - y_hat$se.fit ~ newx$x,
    lty = 2, col = "red")
```



Régression linéaire simple : 1 x quantitatif

Représentation graphique et valeurs prédites

Prédictions "à la main" avec un peu d'algèbre

Exemple : Quelle est la valeur prédite de y pour x = 3.45 ?

```
> (y_hat <- coef(mod)[1] + coef(mod)[2]* 3.45)
(Intercept)
   12.91871
```

On peut utiliser une série de valeurs de x dans un vecteur :

```
> (xnew <-  seq(0, 4, 0.5))
[1] 0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0

> (y_hat <- coef(mod)[1] + coef(mod)[2]* xnew)
[1] 10.50290 10.85302 11.20313 11.55325 11.90337 12.25349 12.60361
[8] 12.95373 13.30384
```

Pour des problèmes plus complexes, il deviendra vite très utile de travailler avec un peu de calcul matriciel très simple...

Régression linéaire simple : 1 x quantitatif

Calcul matriciel basique

Somme (et différence)

$$\begin{pmatrix} 1 & 2 & 0 \\ 3 & 4 & 5 \end{pmatrix} + \begin{pmatrix} 5 & 2 & 1 \\ 3 & 2 & 1 \end{pmatrix} = \begin{pmatrix} 6 & 4 & 1 \\ 6 & 6 & 6 \end{pmatrix}$$

Produit scalaire

$$2 * \begin{pmatrix} 1 & 2 & 0 \\ 3 & 4 & 5 \end{pmatrix} = \begin{pmatrix} 2 & 4 & 0 \\ 6 & 8 & 10 \end{pmatrix}$$

Transposition

$$X = \begin{pmatrix} 1 & 2 & 0 \\ 3 & 4 & 5 \end{pmatrix} \Leftrightarrow X^T = X' = \begin{pmatrix} 1 & 3 \\ 2 & 4 \\ 0 & 5 \end{pmatrix}$$

```
> X <- matrix(c(1,3,2,4,0,5), 2, 3)
> Y <- matrix(c(5,3,2,2,1,1), 2, 3)
>
> X+Y
     [,1] [,2] [,3]
[1,]    6    4    1
[2,]    6    6    6


> 2*X
     [,1] [,2] [,3]
[1,]    2    4    0
[2,]    6    8   10

> t(X)
     [,1] [,2]
[1,]    1    3
[2,]    2    4
[3,]    0    5
```

Régression linéaire simple : 1 x quantitatif

Calcul matriciel basique

Produit matriciel : très utile !

$$YX' = YX^T = \begin{pmatrix} 5 & 2 & 1 \\ 3 & 2 & 1 \end{pmatrix} \begin{pmatrix} 1 & 3 \\ 2 & 4 \\ 0 & 5 \end{pmatrix} = \begin{pmatrix} 9 & 28 \\ 7 & 22 \end{pmatrix}$$

Détail du calcul :

$$\begin{pmatrix} 5*1+2*2+1*0 & 5*3+2*4+1*5 \\ 3*1+2*2+1*0 & 3*3+2*4+1*5 \end{pmatrix}$$

Dans R,
produit matriciel :
%*%

```
> Y %*% t(X)
     [,1] [,2]
[1,]    9   28
[2,]    7   22
```

Régression linéaire simple : 1 x quantitatif

Calcul matriciel basique

Inverse d'une matrice carrée : difficile à faire "à la main"

$$X = \begin{pmatrix} 1 & 1 \\ -1 & 0 \\ 3 & 1 \end{pmatrix}$$

Matrice carrée
n x n

$$X'X = \begin{pmatrix} 1 & -1 & 3 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ -1 & 0 \\ 3 & 1 \end{pmatrix} = \begin{pmatrix} 11 & 4 \\ 4 & 2 \end{pmatrix}$$

Inverse de la Matrice

$$[X'X]^{-1} = \begin{pmatrix} 0.3333 & -0.6666 \\ -0.6666 & 1.8333 \end{pmatrix}$$

Dans R,
Inverser une matrice :
`solve()`

```
> X <- matrix(c(1, -1, 3, 1, 0, 1), 3,2)
> t(X) %*% X
     [,1] [,2]
[1,]   11    4
[2,]    4    2
> solve(t(X) %*% X)
           [,1]       [,2]
[1,]  0.3333333 -0.6666667
[2,] -0.6666667  1.8333333
```

Régression linéaire simple : 1 x quantitatif

Calcul matriciel basique

Pourquoi se casser la tête avec cà ?
Beaucoup plus facile de calculer des valeurs prédites en particulier dans des cas plus complexes

$$\hat{Y} = X\,\hat{B} = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \\ 1 & x_4 \\ 1 & x_5 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} 1*\alpha+\beta*x_1 \\ 1*\alpha+\beta*x_2 \\ 1*\alpha+\beta*x_3 \\ 1*\alpha+\beta*x_4 \\ 1*\alpha+\beta*x_5 \end{pmatrix}$$

"model matrix" avec les valeurs de x pour lesquelles on veut une prédiction et une colonne de "1" correspondant à l'intercept alpha

vecteur des coefficients de régression

calcul des valeurs prédites comme on les aurait fait "à la main"

Régression linéaire simple : 1 x quantitatif

Représentation graphique et valeurs prédites

Prédictions "à la main" avec un peu de calcul matriciel

```
> X <- cbind(1, seq(0, 4, 0.01))
> X[1:5,]
     [,1] [,2]
[1,]    1 0.00
[2,]    1 0.01
[3,]    1 0.02
[4,]    1 0.03
[5,]    1 0.04
```

Nouvelles valeurs de X pour lesquelles on veut une prédiction
La première colonne correspond à l'intercept

```
> beta <- coef(mod)
> beta
(Intercept)           x
 10.5028981   0.7002367
```

Vecteur avec les coefficients

```
> y_hat <- X %*% beta
```

$$\hat{Y} = X\,\hat{B}$$

Le produit matriciel donne les valeurs prédites

```
> V <- as.matrix(vcov(mod))
> y_hat_se <- sqrt(diag(X %*% V %*% t(X)))
```

$$V\left(\hat{Y}\right) = X\,V\left(\hat{B}\right)X'$$

Cette formule utilisant la matrice de variance-covariance des coefficients donne les erreurs standard des prédictions

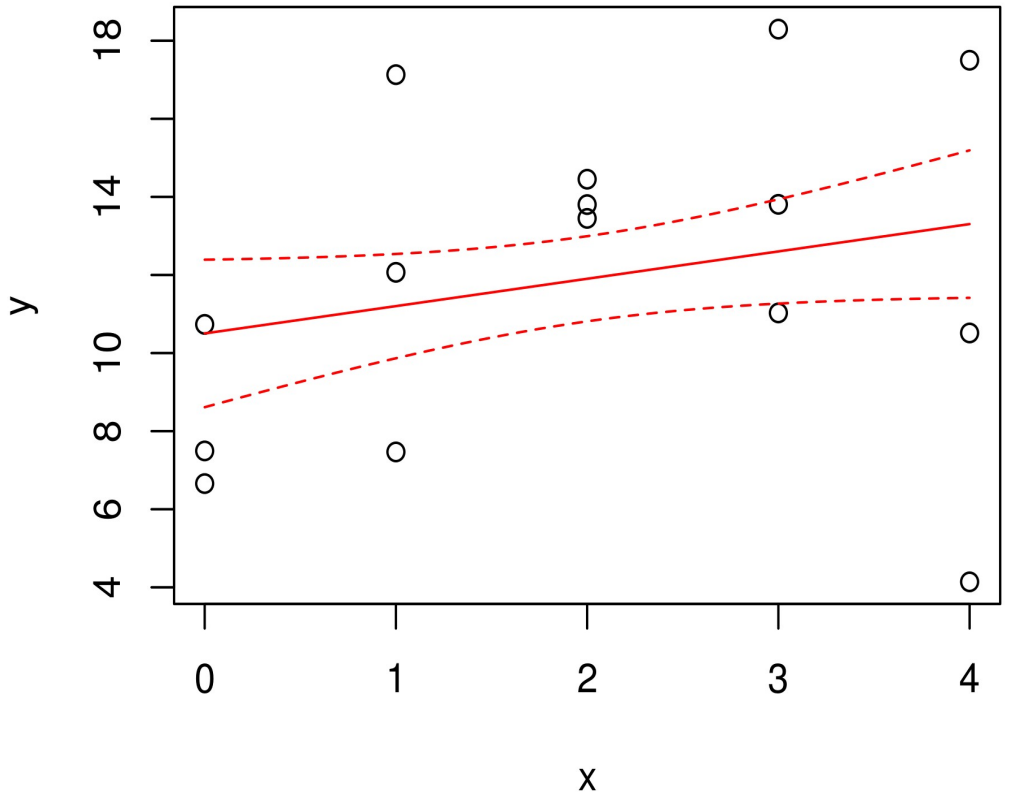
Régression linéaire simple : 1 x quantitatif

Représentation graphique et valeurs prédites

Prédictions "à la main" avec un peu de calcul matriciel

Résultats exactement identiques avec ceux obtenus via `predict()`

```
plot(y ~ x)
lines(y_hat ~ X[,2], col = "red")
lines(y_hat + y_hat_se ~ X[,2], lty = 2, col = "red")
lines(y_hat - y_hat_se ~ X[,2], lty = 2, col = "red")
```



Régression linéaire simple : 1 x quantitatif

Pas Important !

Illustration de l'utilisation du calcul matriciel :
recalculer les coefficients de régression, la variance résiduelle et les erreurs standard des coefficients

Coefficients de régression

$$\hat{B} = [X' X]^{-1} [X' Y]$$

Variance résiduelle

$$S^2 = \frac{1}{n-k+1} [Y' Y - \hat{B} X' Y]$$

Erreurs standard des coefficients

$$V(\hat{B}) = S^2 [X' X]^{-1} = \begin{pmatrix} S^2_{\beta_0} & Cov_{\beta_0\beta_1} & \cdots \\ Cov_{\beta_0\beta_1} & S^2_{\beta_1} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$$

Régression linéaire simple : 1 x quantitatif

Ce qui se calcule très facilement dans R :

Pas Important !

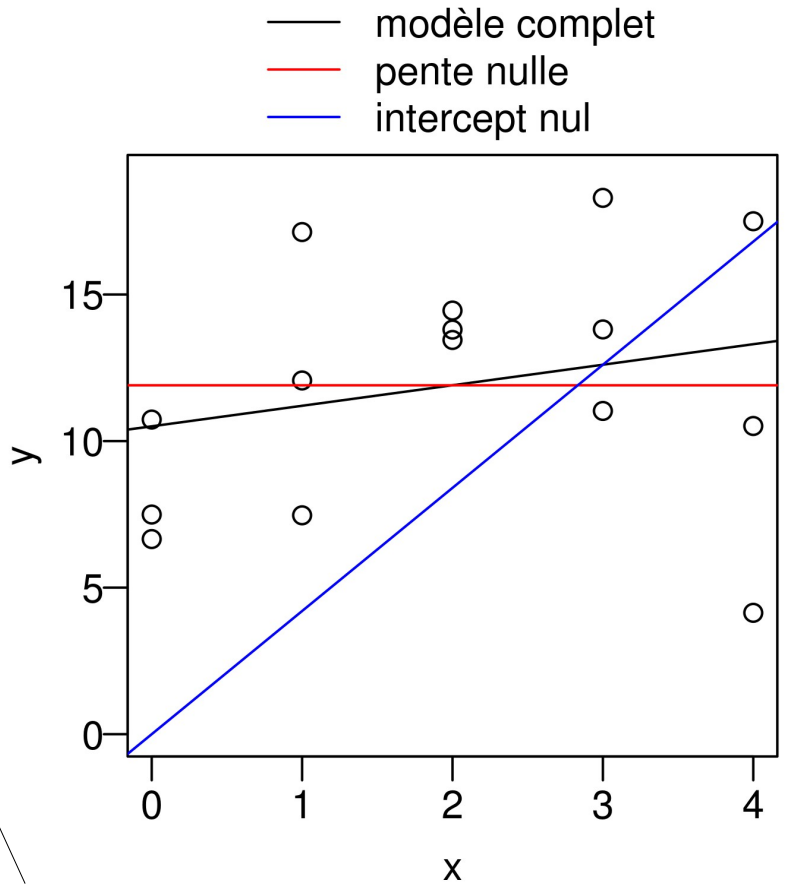
```
> summary(mod)

            Estimate Std. Error t value Pr(>|t|)
(Intercept)  10.5029     1.8889   5.560 9.22e-05 ***
x             0.7002     0.7712   0.908     0.38

Residual standard error: 4.224 on 13 degrees of freedom

> x <- cbind(1, x)
> # Coefficients
> (Bhat <- (solve(t(x) %*% x)) %*% (t(x) %*% y))
        [,1]
  10.5028981
x  0.7002367

> # variance résiduelle
> (Ssq <- as.vector(((t(y)%*%y) - (t(Bhat) %*% t(x) %*% y)))/
        (length(y)-length(Bhat)))
[1] 17.84023
> Ssq^0.5
[1] 4.22377

> # erreur standard des coefficients
> (varcov <- Ssq*(solve(t(x) %*% x)))
                      x
   3.568046 -1.1893488
x -1.189349  0.5946744
> diag(varcov)^0.5
                   x
1.8889273 0.7711513
```

Régression linéaire simple : 1 x quantitatif

Interprétation graphique de différentes hypothèses

Modèle complet vs pente nulle vs intercept nul

```
mod0 <- lm(y ~ x) # modèle complet
mod1 <- lm(y ~ 1) # modèle à pente nulle
mod2 <- lm(y ~ -1 + x) # modèle à intercept nul

dev.new(10/2.54, 10/2.54)
par(mar = c(3,3,4,3), mgp = c(1.8, 0.5, 0), las = 1)

plot(y ~ x, ylim = c(0, 19))
abline(mod0)
abline(mod1, col = "red")
abline(mod2, col = "blue")

legend("top", legend = c("modèle complet",
              "pente nulle", "intercept nul"),
         col = c("black", "red", "blue") ,
         lty = 1, bty = "n",  xpd = NA,
         inset = -0.3)
```



`bty = "n"` : pas de cadre autour de la légende

`xpd=NA` : permet de mettre la légende en dehors de la zone de graphique

`inset` : éloigne la légende du bord (en%)

`mar` : taille des marges en nb de lignes

`mgp` : position des "ticks" et étiquettes par rapport à l'axe

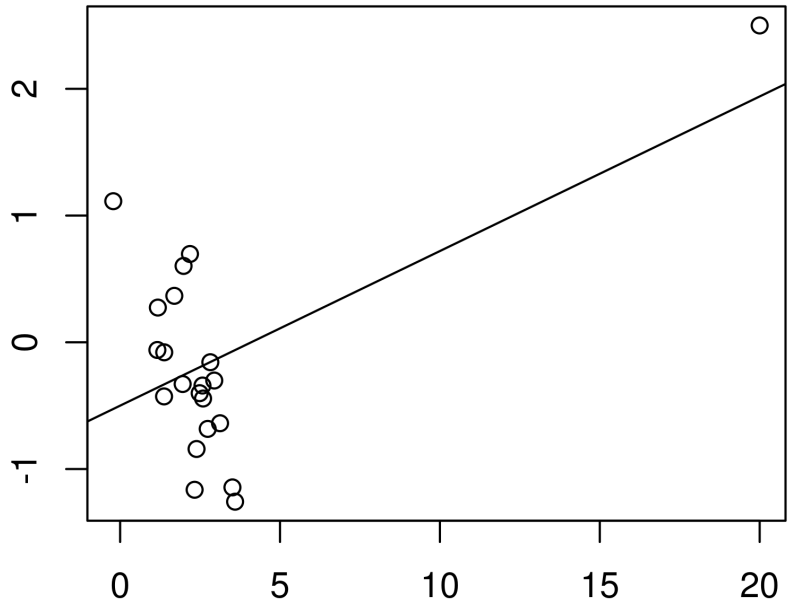
`las` : orientation des étiquettes

Régression linéaire simple : 1 x quantitatif

Attention aux valeurs extrêmes

Aussi appelées "outliers"

```
> set.seed(1)
> x <- c(rnorm(20,2,1),20)
> set.seed(12)
> y <- c(1 -0.5*x[1:20] + rnorm(20,0,0.5),2.5)

> plot(y~x)
> abline(lm(y~x))
> summary(lm(y~x))

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.50004    0.20273  -2.467  0.02333 *
x            0.12198    0.04106   2.970  0.00786 **

Residual standard error: 0.7322 on 19 degrees of freedom
Multiple R-squared:  0.3171,  Adjusted R-squared:  0.2812
F-statistic: 8.823 on 1 and 19 DF,  p-value: 0.00786
```

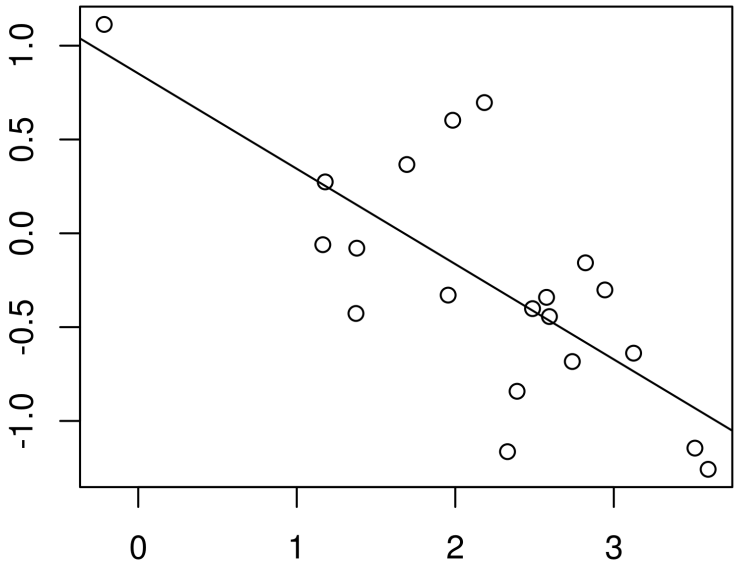


Régression linéaire simple : 1 x quantitatif

Attention aux valeurs extrêmes

Après élimination de la valeur extrême :

```
> plot(y[1:20]~x[1:20])
> abline(lm(y[1:20]~x[1:20]))
> summary(lm(y[1:20]~x[1:20]))

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.8518     0.2645   3.221 0.004742 **
x[1:20]      -0.5079     0.1119  -4.541 0.000253 ***

Residual standard error: 0.4453 on 18 degrees of freedom
Multiple R-squared:  0.5339,  Adjusted R-squared:  0.508
F-statistic: 20.62 on 1 and 18 DF,  p-value: 0.00025
```



Régression linéaire simple : 1 x quantitatif

Attention aux valeurs extrêmes

Que faire ?

Toujours une représentation graphique des données et du modèle !

Essayer d'éviter les "trous" dans les valeurs de x

Vérifier si les valeurs extrêmes ne sont pas des erreurs d'encodage ou de mesure

--> si oui : les éliminer
--> si non :

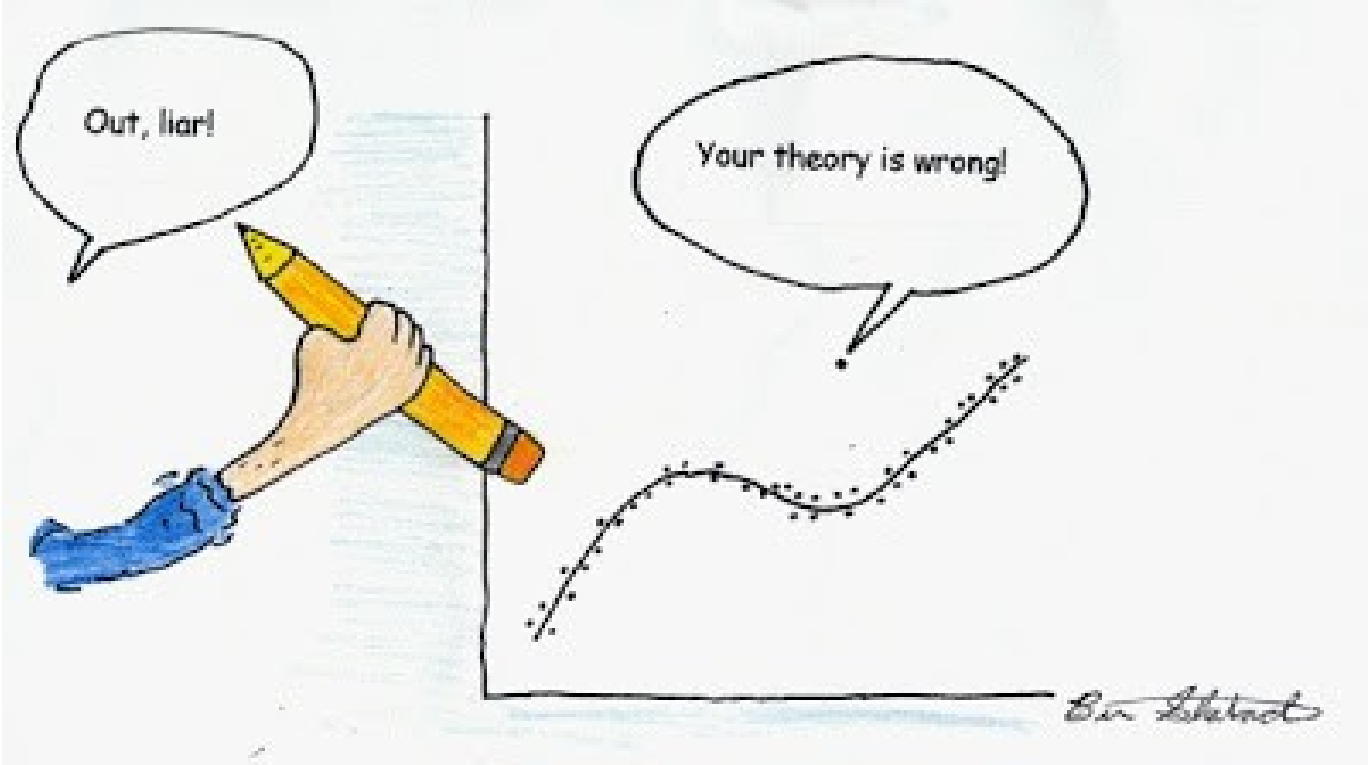
faire l'analyse avec et sans les valeurs extrêmes et discuter les résultats

Utiliser des méthodes robustes aux valeurs extrêmes

Régression linéaire simple : 1 x quantitatif

Attention aux valeurs extrêmes

Aussi appelées "outliers"



Prudence et honnêteté
(envers soi-même)

Régression linéaire simple : 1 x quantitatif

Quand faut-il centrer les données ?

Centrer une variable consiste à soustraire la moyenne à chaque valeur. On peut aussi soustraire une autre valeur conventionnelle.

Il faut se poser la question : est-ce que estimer y quand x = 0 a du sens ou est raisonnable ?

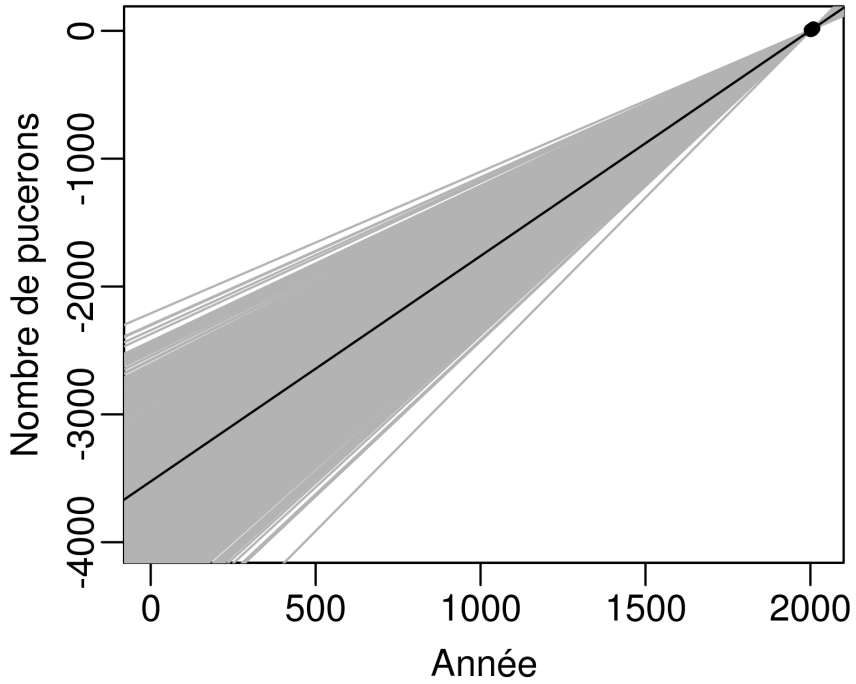
Centrer peut être utile :
- Pour avoir un Intercept interprétable ou placé à un endroit où il est utile et où son erreur standard est raisonable
- pour éviter une trop grande corrélation entre l'intercept et la pente

Régression linéaire simple : 1 x quantitatif

Quand faut-il centrer les données ?

```
> aphids <- c(5, 3, 7, 6, 10, 12, 8, 12, 18, 22, 20)
> year <- c(2000:2010)

> mod <- lm(aphids ~ year)
> summary(mod)

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -3524.9091   499.6924  -7.054 5.96e-05 ***
year            1.7636     0.2492   7.077 5.81e-05 ***
```

On estime que en l'an 0
il y avait -3524 pucerons
avec une erreur standard de 500...



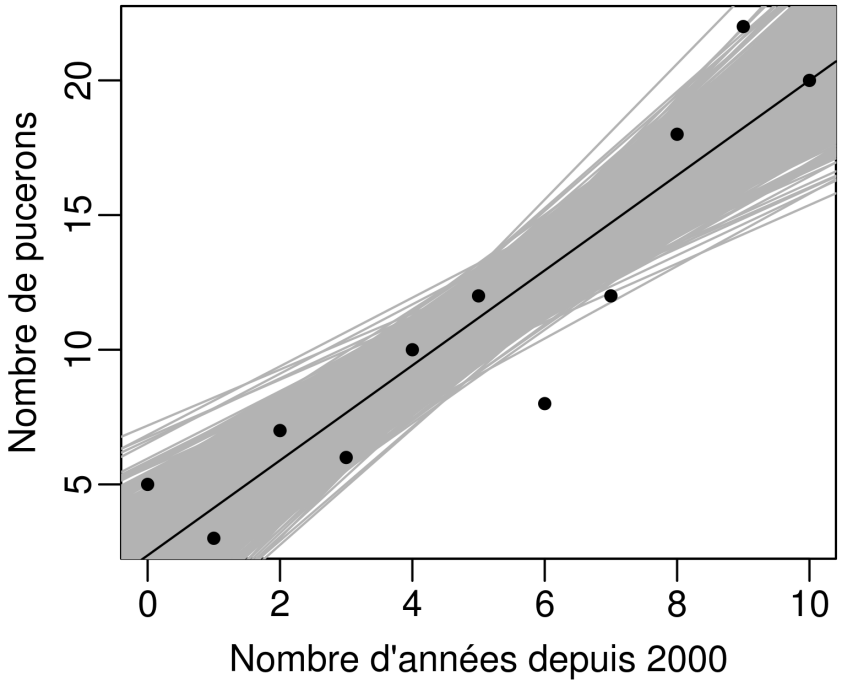
Régression linéaire simple : 1 x quantitatif

Quand faut-il centrer les données ?

```
> cyear <- year-2000
> mod <- lm(aphids ~ cyear)
> summary(mod)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   2.3636     1.4744   1.603    0.143
cyear         1.7636     0.2492   7.077 5.81e-05 ***
```

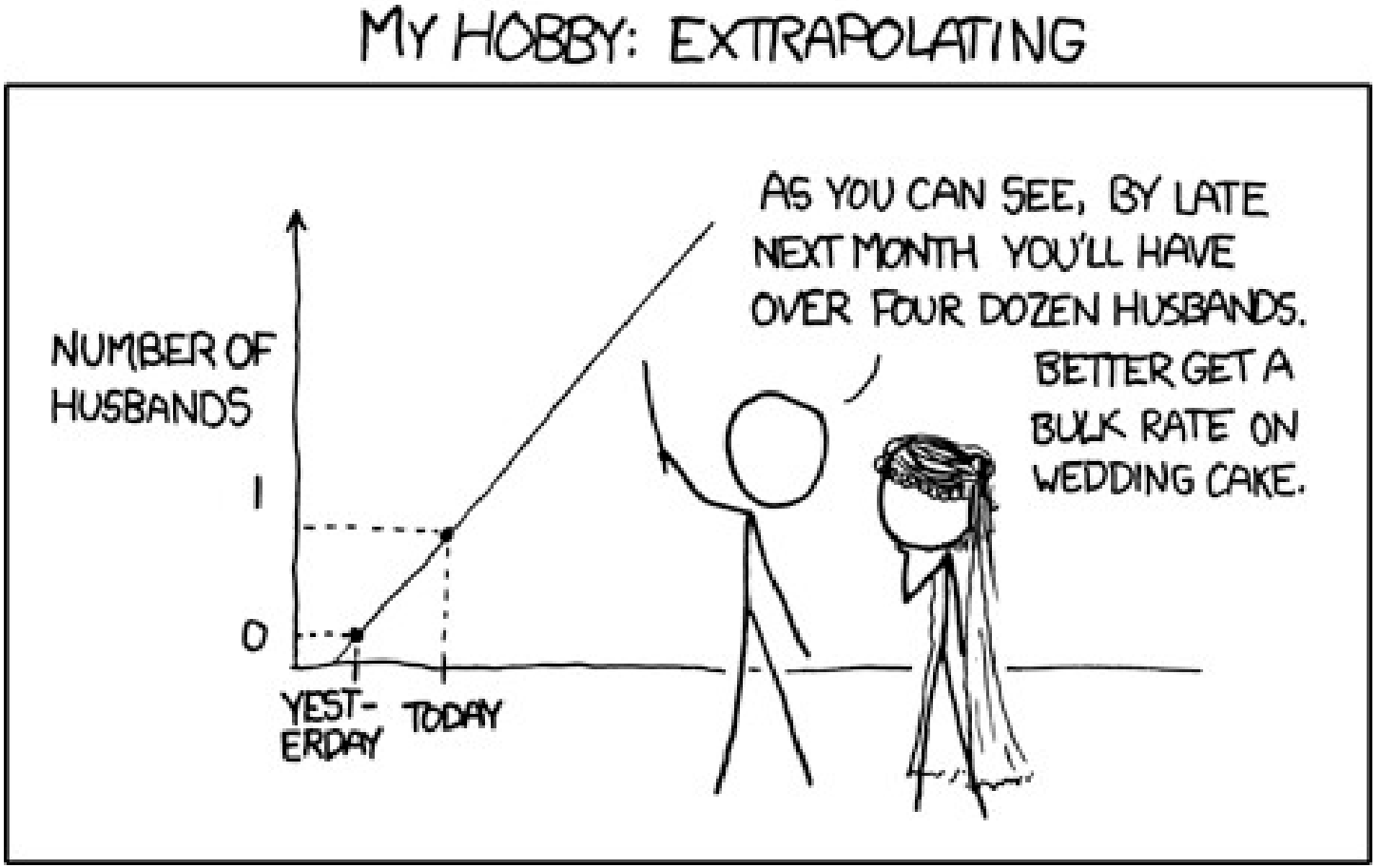
Si on centre la variable x
sur l'année 2000,
l'intercept estime maintenant
le nombre de pucerons en 2000
(~2.36 pucerons).

NB : la pente reste inchangée.



Régression linéaire simple : 1 x quantitatif

Attention à l'extrapolation !



CC by-nc : http://xkcd.com/605/

1 x qualitatif : test de Student - ANOVA

La formulation mathématique est exactement la même mais la variable explicative est un facteur (variable qualitative) transformé en "Dummy variable"

On compare les moyennes des deux groupes ou des n groupes. On se demande si au moins une des moyennes est différente.

Exemple : Différence de production de tomates entre deux ou 3 (ou n) variétés

Concepts à assimiler :

Dummy variables, codage de variables qualitatives contrastes

Comparaisons multiples

Paramétristation du modèle

1 x qualitatif : test de Student - ANOVA

Dummy variables

Il s'agit de variables composées uniquement de 0 et de 1 permettant de classer les données en groupes mutuellement exclusifs.

En général on choisi un niveau de référence (par défaut dans R : le premier dans l'ordre alphabétique) qui correspondra à l'intercept et on construit les autres variables par rapport à ce niveau.

1 x qualitatif : test de Student - ANOVA

Dummy variables

Exemple : comparaison de la production de tomates entre 2 variétés (variable qualitative à 2 niveaux)

Valeurs observées de y

Valeurs observées de x

Intercept

Var2

$$Y = X\,\hat{B} \qquad \begin{pmatrix} 21 \\ 23 \\ 16 \\ 26 \\ 34 \\ 28 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$$

La Col 2 indique simplement si l'observation appartient oui ou non à la variété 2

Les 3 premières observations (lignes) correspondent à la variété 1 les 3 suivantes à la variété 2

col1 correspond à α = moyenne de la variété 1 prise comme référence

Col 2 correspond à β = différence moyenne entre var 1 et var2

1 x qualitatif : test de Student - ANOVA

Dummy variables

Exemple : comparaison de la production de tomates entre 3 variétés (variable qualitative à 3 niveaux)

Les 3 premières observations (lignes) correspondent à la variété 1
les 3 suivantes à la variété 2
et les 3 dernières à la variété 3

$$Y = X\,\hat{B}$$

$$\begin{pmatrix} 21 \\ 23 \\ 16 \\ 26 \\ 34 \\ 28 \\ 10 \\ 8 \\ 6 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{pmatrix}$$

Intercept Var2 Var3

Col1 correspond à β_0 = moyenne de la variété 1 prise comme référence

Col 2 correspond à β_1 = différence moyenne entre var 1 et var 2

Col 3 correspond à β_2 = différence moyenne entre var 1 et var 3

1 x qualitatif : test de Student - ANOVA

Dummy variables

En pratique, R fait le boulot à notre place si la variable explicative est un facteur. On peut extraire la matrice utilisée avec la fonction `model.matrix()`

```
> n <- 4
> set.seed(123)
> variety1 <- rnorm(n,23,5) # mean = 23
> set.seed(12)
> variety2 <- rnorm(n,15,5) # mean = 15
>
> data <- data.frame(tomato= c(variety1,variety2),
                     variety= rep(c("variety1","variety2"), each=n))
> data
     tomato  variety
1 20.197622 variety1
2 21.849113 variety1
3 30.793542 variety1
4 23.352542 variety1
5  7.597162 variety2
6 22.885847 variety2
7 10.216278 variety2
8 10.399974 variety2
```

```
> mod <- lm(tomato ~ variety, data=data)
> model.matrix(mod)
  (Intercept) varietyvariety2
1           1                0
2           1                0
3           1                0
4           1                0
5           1                1
6           1                1
7           1                1
8           1                1
```

1 x qualitatif : test de Student - ANOVA

Dummy variables

```
> n <- 4
> set.seed(123)
> variety1 <- rnorm(n,23,5) # mean = 23
> set.seed(12)
> variety2 <- rnorm(n,15,5) # mean = 15
> set.seed(1)
> variety3 <- rnorm(n,10,5) # mean = 10
>
> data <- data.frame(tomato= c(variety1,variety2, variety3),
                     variety= rep(c("variety1","variety2", "variety3"), each=n))
> data
      tomato  variety
1  20.197622 variety1
2  21.849113 variety1
3  30.793542 variety1
4  23.352542 variety1
5   7.597162 variety2
6  22.885847 variety2
7  10.216278 variety2
8  10.399974 variety2
9   6.867731 variety3
10 10.918217 variety3
11  5.821857 variety3
12 17.976404 variety3
```

```
> mod <- lm(tomato ~ variety, data=data)
> model.matrix(mod)
   (Intercept) varietyvariety2 varietyvariety3
1            1                0                0
2            1                0                0
3            1                0                0
4            1                0                0
5            1                1                0
6            1                1                0
7            1                1                0
8            1                1                0
9            1                0                1
10           1                0                1
11           1                0                1
12           1                0                1
```

1 x qualitatif : test de Student - ANOVA

Une variable qualitative à 2 niveaux

Un tel LM est strictement identique à un test de Student de comparaison des moyennes

Construction du jeu de données pour 2 variétés avec 4 observations par variété :

```
> n <- 4
> set.seed(123)
> variety1 <- rnorm(n,23,5) # mean = 20
> set.seed(12)
> variety2 <- rnorm(n,15,5) # mean = 15

> # on construit un jeu de données
> data <- data.frame(tomato= c(variety1,variety2),
                     variety= rep(c("variety1","variety2"), each=n))
> data
     tomato  variety
1 20.197622 variety1
2 21.849113 variety1
3 30.793542 variety1
4 23.352542 variety1
5  7.597162 variety2
6 22.885847 variety2
7 10.216278 variety2
8 10.399974 variety2
```

1 x qualitatif : test de Student - ANOVA

Une variable qualitative à 2 niveaux

Un tel LM est strictement identique à un test de Student de comparaison des moyennes

```
> # t test
> t.test(variety1, variety2, var.equal=TRUE)
t = 2.7151, df = 6, p-value = 0.03487
```

On estime que le poids moyen des tomates est de 24.048 kg quand x = 0 c'est à dire pour la variété 1. La probabilité d'obtenir un tel poids par hasard (si la production réelle était nulle) est très faible : p = 0.00018 Ce test n'a pas beaucoup d'intérêt ici...

```
> mod <- lm(tomato ~ variety, data=data)
> summary(mod)

                Estimate Std. Error t value Pr(>|t|)
(Intercept)       24.048      2.936   8.191 0.000178 ***
varietyvariety2  -11.273      4.152  -2.715 0.034867 *
```

On estime que en moyenne la variété 2 produit 11.27 kg de tomates en moins que la variété 1. La probabilité d'obtenir une telle différence uniquement par hasard (si il n'y avait aucune différence) est faible : p = 0.0349

```
> anova(mod)
Analysis of Variance Table

Response: tomato
          Df Sum Sq Mean Sq F value  Pr(>F)
variety    1 254.18 254.179   7.372 0.03487 *
Residuals  6 206.87  34.479
```

Le test de comparaison de modèles est toujours identique dans ce cas-ci

1 x qualitatif : test de Student - ANOVA

Intervalles de confiance sur les paramètres

Classiquement on peut obtenir un intervalle de confiance à 95 % approximatif en prenant le paramètre +- 2* son erreur standard.
Par exemple la différence de production serait comprise entre `-11.273 - 2*4.152` et `-11.273 + 2*4.152`

Mais comme le nombre d'observations est très faible ici, il vaut mieux utiliser les quantiles de la loi de Student ou la fonction `confint()` :

```
> mod <- lm(tomato ~ variety, data=data)
> summary(mod)

                 Estimate Std. Error t value Pr(>|t|)
(Intercept)        24.048      2.936   8.191 0.000178 ***
varietyvariety2  -11.273      4.152  -2.715 0.034867 *

> confint(mod)
                    2.5 %    97.5 %
(Intercept)      16.86422 31.232186
varietyvariety2 -21.43307 -1.113705
```

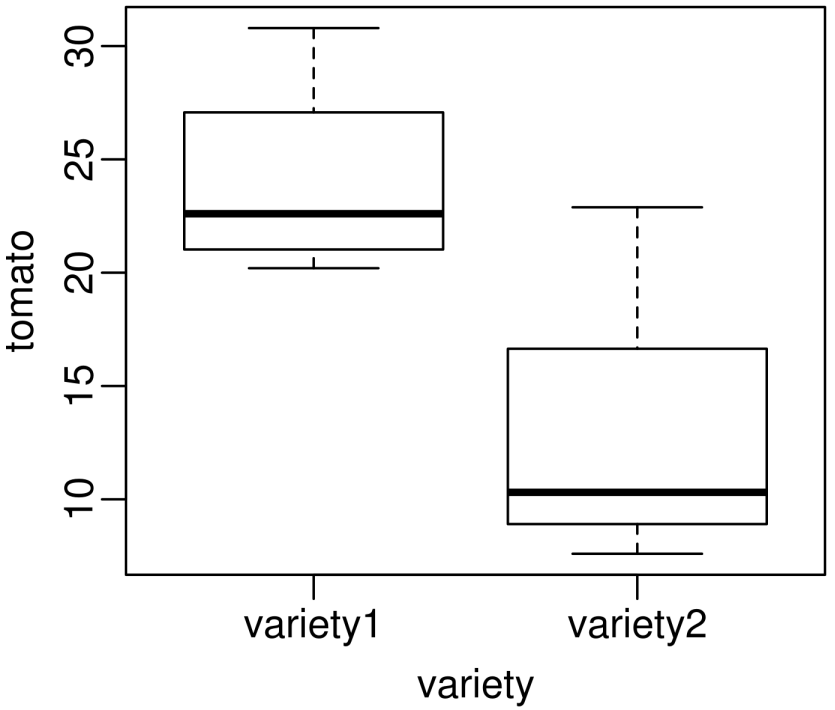
Si on répétait l'expérience 1000 fois l'estimation de la différence de production serait comprise dans 95 % des cas entre -21.4 et -1.11 kg

1 x qualitatif : test de Student - ANOVA

Représentation graphique

Si on représente les données par défaut, on obtient un boxplot. Il s'agit d'une représentation des données mais pas du modèle.

```
> par(mar= c(3,3,1,1),mgp=c(1.75,0.5, 0))
> plot(tomato ~variety , data=data)
```



1 x qualitatif : test de Student - ANOVA

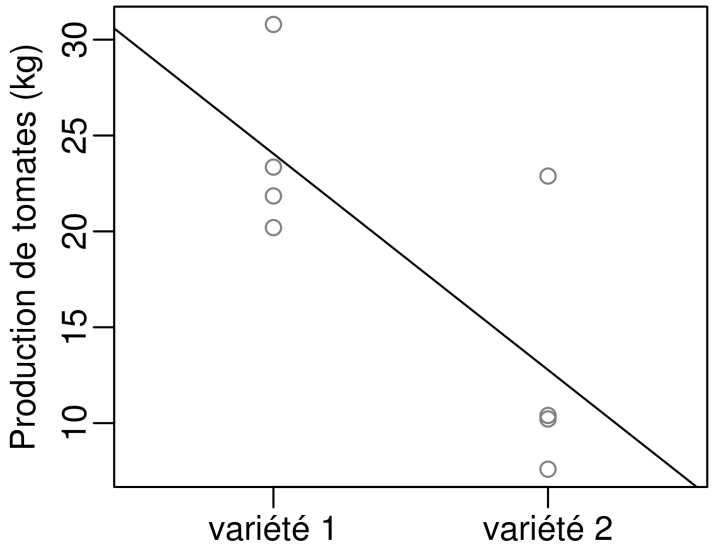
Représentation graphique

Pour représenter les données sous forme de points il faut transformer la variable x (un facteur) en numérique.
La représentation du modèle sous forme d'une droite (`abline`) n'est pas vraiment adaptée ici.

`xaxt="n"` : Supprime l'axe des x

```
> position <- as.numeric(data$variety)-1
> plot(tomato ~ position, data=data, xaxt="n", xlim = c(-0.5, 1.5),
+ col = "grey50", xlab = "", ylab = "Production de tomates (kg)" )
> axis(side = 1, at = c(0,1), labels = c("variété 1", "variété 2"))
> abline(mod)
```

On ajoute l'axe des x manuellement



1 x qualitatif : test de Student - ANOVA

Représentation graphique

On calcule les valeurs prédites et leurs erreurs standard

Avec la fonction `predict()`

```
> (newdata <- data.frame(variety = as.factor(c("variety1", "variety2"))))
   variety
1 variety1
2 variety2

> (pred <- predict(mod, newdata, se.fit=TRUE))
$fit
       1        2
24.04820 12.77482

$se.fit
       1        2
2.935938 2.935938
```

Nouveau jeu de données : les valeurs de x pour lesquelles on veut une prédiction

1 x qualitatif : test de Student - ANOVA

Représentation graphique

On calcule les valeurs prédites et leurs erreurs standard

Avec le calcul matriciel :

```
> beta <- coef(mod)
> X <- cbind(1, c(0,1))
> X
     [,1] [,2]
[1,]    1    0
[2,]    1    1

> pred <- X %*% beta

> V <- as.matrix(vcov(mod))
> se <- sqrt(diag(X %*% V %*% t(X)))
> pred
          [,1]
[1,] 24.04820
[2,] 12.77482
> se
[1] 2.935938 2.935938
```

Matrice X pour laquelle on veut une prédiction.
La première ligne représente la variété 1
La deuxième ligne représente la variété 2

1 x qualitatif : test de Student - ANOVA

Représentation graphique

On ajoute les valeurs prédites et leurs erreurs standard sur le graphique

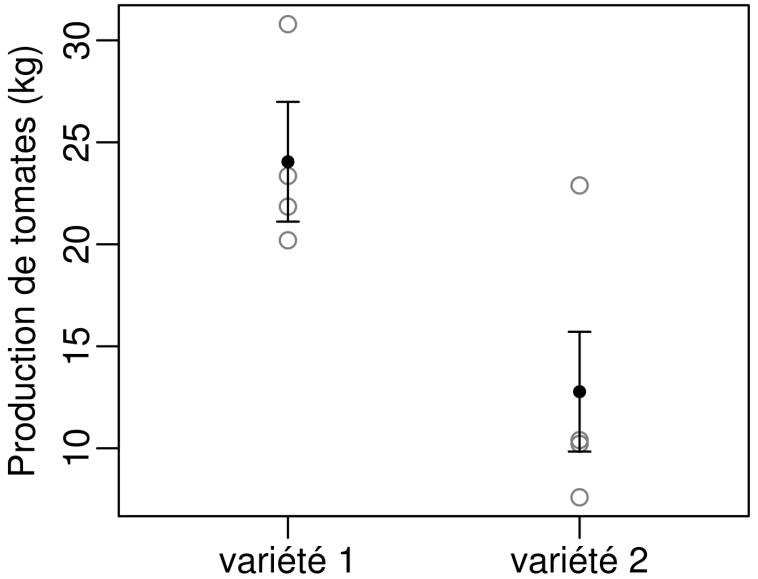
```
position <- as.numeric(data$variety)-1
plot(tomato ~ position, data=data, xaxt="n", xlim = c(-0.5, 1.5),
     col = "grey50", xlab = "", ylab = "Production de tomates (kg)" )
axis(side = 1, at = c(0,1), labels = c("variété 1", "variété 2"))
```

Ajoute les 2 points représentant les moyennes estimées au graphique existant

```
points(x = c(0,1), y = pred, pch=20 )
arrows(x0 = c(0,1), y0 = pred-se, x1 = c(0,1), y1 = pred + se,
       angle=90, length = 0.05, code = 3)
```

Ajoute les barres d'erreur

NB : On appelle parfois les valeurs prédites par le modèle : "Least Square Means"



1 x qualitatif : test de Student - ANOVA

Une variable qualitative à n niveaux

Il s'agit de ce qu'on appelle généralement une ANOVA

mais l'approche GLM met plus l'accent sur l'estimation des paramètres que sur le tableau d'analyse de la variance

```
# Génération du jeu de données : 4 variétés, 4 répétitions
n <- 4
beta0 <- 23
beta1 <- -2
beta2 <- -10
beta3 <- -12
sigma <- 4
B <- c(beta0, beta1, beta2, beta3)

variety <- rep(c("var1", "var2", "var3", "var4"), each = n)
X <- model.matrix(~ variety)

set.seed(1)
y <- X %*% B + rnorm(4*n, 0, sigma)

d <- data.frame(tomato= y, variety= variety)
> d
      tomato variety
1  20.494185    var1
2  23.734573    var1
3  19.657486    var1
4  29.381123    var1
5  22.318031    var2
(...)
```

On crée les Dummy variables à l'aide de la fonction `model.matrix()`

```
> X
   (Intercept) varietyvar2 varietyvar3 varietyvar4
1            1           0           0           0
2            1           0           0           0
3            1           0           0           0
4            1           0           0           0
5            1           1           0           0
6            1           1           0           0
7            1           1           0           0
8            1           1           0           0
9            1           0           1           0
10           1           0           1           0
11           1           0           1           0
12           1           0           1           0
13           1           0           0           1
14           1           0           0           1
15           1           0           0           1
16           1           0           0           1
```

1 x qualitatif : test de Student - ANOVA

Une variable qualitative à n niveaux

Il s'agit de ce qu'on appelle généralement une ANOVA

mais l'approche GLM met plus l'accent sur l'estimation des paramètres que sur le tableau d'analyse de la variance

```
> mod <- lm(tomato ~ variety, data=d)
> summary(mod)

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    23.317      2.045  11.403 8.52e-08 ***
varietyvar2    -1.582      2.892  -0.547 0.594356
varietyvar3    -8.145      2.892  -2.816 0.015561 *
varietyvar4   -14.073      2.892  -4.866 0.000387 ***

Residual standard error: 4.09 on 12 degrees of freedom
Multiple R-squared:  0.714,   Adjusted R-squared:  0.6425
F-statistic: 9.987 on 3 and 12 DF,  p-value: 0.001393
```

On estime par exemple que la variété 2 produit en moyenne, 1.58 kg en moins que la variété 1 (choisie arbitrairement comme référence) mais qu'on aurait pu obtenir le même effet par hasard (p=0.59)

```
> mod0 <- lm(tomato ~ 1, data=d)
> mod1 <- lm(tomato ~ variety, data=d)
> anova(mod0,mod1)
Analysis of Variance Table

Model 1: tomato ~ 1
Model 2: tomato ~ variety
  Res.Df    RSS Df Sum of Sq      F   Pr(>F)
1     15 701.82
2     12 200.71  3    501.11 9.9871 0.001393 **
```

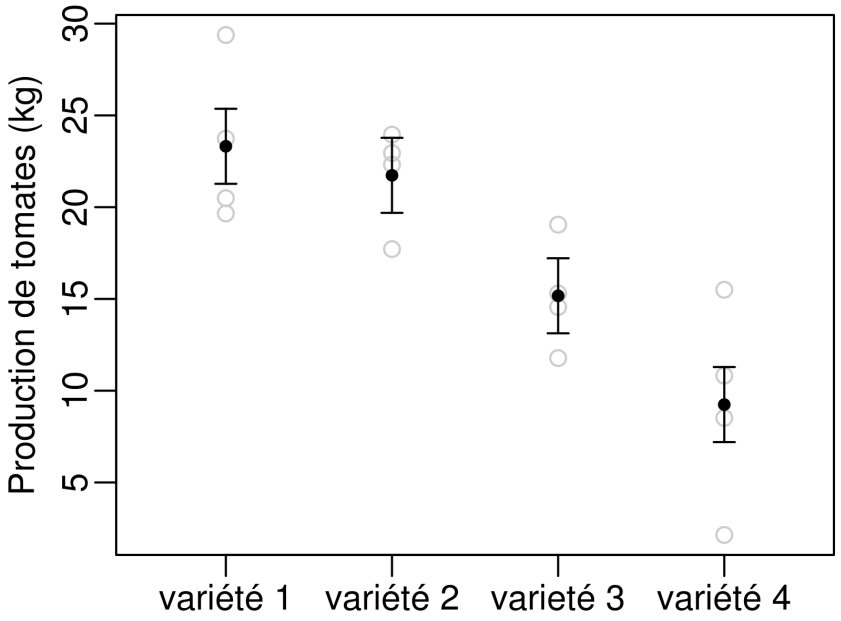
La comparaison de modèles emboîtés ne teste plus la même hypothèse ! Elle teste l'hypothèse qu'au moins une des variétés a une production différente de d'une autre variété quelconque. C'est en général cette question qu'il faut se poser en premier !!

1 x qualitatif : test de Student - ANOVA

Une variable qualitative à n niveaux

Représentation graphique

```
> position <- as.numeric(d$variety)-1
> plot(tomato ~ position, data=d, xaxt="n", xlim = c(-0.5, 3.5),
+      col = "grey80", xlab = "",ylab = "Production de tomates (kg)" )
> axis(side = 1, at = c(0,1,2,3), labels =
        c("variété 1", "variété 2", "varieté 3", "variété 4"))

> pred <- predict(mod, data.frame(variety =
        as.factor(c("var1", "var2", "var3", "var4"))), se.fit=TRUE)

> beta <- coef(mod)
> X <- rbind(c(1,0,0,0), c(1,1,0,0),
                 c(1,0,1,0), c(1,0,0,1))
> X
     [,1] [,2] [,3] [,4]
[1,]    1    0    0    0
[2,]    1    1    0    0
[3,]    1    0    1    0
[4,]    1    0    0    1

> pred <- X %*% beta

> V <- as.matrix(vcov(mod))
> se <- sqrt(diag(X %*% V %*% t(X)))

> points(x = c(0,1,2,3), y = pred, pch=20 )
> arrows(x0 = c(0,1,2,3), y0 = pred-se, x1 = c(0,1,2,3), y1 = pred + se,
    angle=90, length = 0.05, code = 3)
```



1 x qualitatif : test de Student - ANOVA

Une variable qualitative à n niveaux

NB : dans ce cas, les valeurs prédites par le modèle sont exactement égales aux moyennes de chaque variété.

Dans ce cas précis, utiliser les valeurs prédites n'est pas particulièrement utile mais ces valeurs seront différentes et plus utiles dans de nombreux autres cas

Par contre, les erreurs standard des prédictions sont proches mais pas exactement égales aux erreurs standard de la moyenne

```
> pred
          [,1]
[1,] 23.316842
[2,] 21.734793
[3,] 15.172017
[4,]  9.244057
> aggregate(d$tomato, by = d["variety"], mean)
  variety         x
1    var1 23.316842
2    var2 21.734793
3    var3 15.172017
4    var4  9.244057
>
> se
[1] 2.044836 2.044836 2.044836 2.044836
> aggregate(d$tomato, by = d["variety"], sd)[,2] / sqrt(4)
[1] 2.204319 1.380567 1.497924 2.777886
```

Moyennes calculées pour chaque variété

Erreurs standard de la moyenne= S/sqrt(n)

1 x qualitatif : test de Student - ANOVA

Post-hoc tests - comparaisons multiples

Cette méthode de prédiction permet en fait de tester de nombreuses hypothèses une fois que le modèle a été estimé (post-hoc) en calculant les valeurs prédites

Estimer la différence entre var3 et var4

Moyenne prédite pour var 3 - Moyenne prédite pour var 4

```
> c( 1, 0, 1, 0) %*% coef(mod) - c( 1, 0, 0, 1) %*% coef(mod)
          [,1]
[1,] 5.927961

> c( 0, 0, 1, -1) %*% coef(mod)
          [,1]
[1,] 5.927961
```

Version simplifiée donnant les mêmes résultats

1 x qualitatif : test de Student - ANOVA

Post-hoc tests - comparaisons multiples

Estimer la différence entre var2 et var3

```
> c( 1, 1, 0, 0) %*% coef(mod) - c( 1, 0, 1, 0) %*% coef(mod)
          [,1]
[1,] 6.562776
> c( 0, 1, -1, 0) %*% coef(mod)
          [,1]
[1,] 6.562776
```

Est-ce que les variétés 1 et 2 ont en moyenne une productivité différente des variétés 3 et 4 ?
Soit est-ce que 1/2(var1 + var2) - 1/2(var3 + var4) = 0 ?

```
> 0.5*(c( 1, 0, 0, 0) %*% coef(mod) + c( 1, 1, 0, 0) %*% coef(mod)) -
+      0.5*(c( 1, 0, 1, 0) %*% coef(mod) + c( 1, 0, 0, 1) %*% coef(mod))
          [,1]
[1,] 10.31778

> c( 0, 0.5, -0.5, -0.5) %*% coef(mod)
          [,1]
[1,] 10.31778
```

1 x qualitatif : test de Student - ANOVA

Post-hoc tests - comparaisons multiples

Est-ce que la variété 1 produit 2 fois plus que la variété 4 ?
Soit est-ce que var1 - 2* var4 = 0 ?

```
> c( 1, 0, 0, 0) %*% coef(mod) - 2*c( 1, 0, 0, 1) %*% coef(mod)
          [,1]
[1,] 4.828728
> c( -1, 0, 0, -2) %*% coef(mod)
          [,1]
[1,] 4.828728
```

1 x qualitatif : test de Student - ANOVA

Post-hoc tests - comparaisons multiples

On peut rassembler toutes ces hypothèses/questions dans une seule matrice de contrastes et estimer l'effet et son erreur standard ainsi que la statistique t et sa p-valeur.
Mais ces p-valeurs ne tiennent pas en compte le fait qu'on fait de nombreux tests

```
> X <- matrix(c(
+       0, 0, 1, -1,
+       0, 1, -1, 0,
+       0, 0.5, -0.5, -0.5,
+       -1, 0, 0, -2), 4,4, byrow = TRUE)
> row.names(X) <- c("Var3 - Var4", "Var2 - Var3",
                    "0.5(var1 + var2) - 0.5(var3 + var4)", "Var1 - 2*Var4")
>
> pred <- X %*% coef(mod)
> V <- as.matrix(vcov(mod))
> se <- sqrt(diag(X %*% V %*% t(X)))
>
> posthoc <- data.frame(pred = pred, se = se)
> posthoc$t <- posthoc$pred/posthoc$se
> posthoc$p <- 2*pt(posthoc$t, df = mod$df.residual, lower.tail = FALSE)
> posthoc
                                       pred       se        t           p
Var3 - Var4                        5.927961 2.891835 2.049896 0.0628832633
Var2 - Var3                        6.562776 2.891835 2.269416 0.0424827812
0.5(var1 + var2) - 0.5(var3 + var4) 10.317780 2.044836 5.045774 0.0002865308
Var1 - 2*Var4                      4.828728 4.572392 1.056062 0.3117380646
```

1 x qualitatif : test de Student - ANOVA

Post-hoc tests - comparaisons multiples

Le problème est que lorsqu'on multiplie de la sorte les tests non indépendants, on augmente aussi le risque global de se tromper en rejetant l'hypothèse nulle en particulier quand on teste plusieurs hypothèses non indépendantes sur un même jeu de données.
(les p-valeurs sont trop petites)

Il existe différentes techniques pour ajuster les p-valeurs pour prendre en compte le risque global de se tromper.

Certaines méthodes sont plus conservatives que d'autres.
Il n'existe malheureusement pas de moyen de trancher objectivement entre ces méthodes
qui vont favoriser plutôt l'un ou l'autre type d'erreur

1 x qualitatif : test de Student - ANOVA

Post-hoc tests - comparaisons multiples

Une de ces méthodes est implémentée dans le package "multcomp". Il suffit de fournir le modèle et la matrice de contrastes X à la fonction glht (general linear hypotheses tests). Il utilise distributions de Student ou Normale multivariées. Un avantage de cette méthode est qu'elle est généralisable aux Generalized Linear Models et aux modèles mixtes (moyennant certaines approximations).

```
> posthoc
                                          pred       se        t            p
Var3 - Var4                           5.927961 2.891835 2.049896 0.0628832633
Var2 - Var3                           6.562776 2.891835 2.269416 0.0424827812
0.5(var1 + var2) - 0.5(var3 + var4) 10.317780 2.044836 5.045774 0.0002865308
Var1 - 2*Var4                         4.828728 4.572392 1.056062 0.3117380646
> library(multcomp)
> modmc <- glht(mod, linfct = X)
> summary(modmc)

Linear Hypotheses:
                                       Estimate Std. Error t value Pr(>|t|)
Var3 - Var4 == 0                          5.928      2.892   2.050    0.167
Var2 - Var3 == 0                          6.563      2.892   2.269    0.117
0.5(var1 + var2) - 0.5(var3 + var4) == 0 10.318      2.045   5.046   <0.001 ***
Var1 - 2*Var4 == 0                        4.829      4.572   1.056    0.624
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
(Adjusted p values reported -- single-step method)
```

p valeurs non ajustées

1 x qualitatif : test de Student - ANOVA

Post-hoc tests - comparaisons multiples

Classiquement, on fait d'abord un test global (comparaison de modèles emboîtés, ANOVA).

Si ce test est non significatif, on arrête là, aucune des comparaisons multiples ne devrait être significatives.

Si il est significatif on effectue les comparaisons multiples avec correction des p-valeurs en sachant que plus on fait de tests plus on perd de puissance.
--> il faut se limiter aux hypothèses vraiment intéressantes.

Il peut arriver que le test global soit significatif et qu'aucun des tests post-hoc ne le soient...

1 x qualitatif : test de Student - ANOVA

Post-hoc tests - comparaisons multiples

Deux cas de comparaisons multiples sont particulièrement fréquents :
1) Comparer toutes les moyennes à une moyenne (souvent le témoin).
(cfr test de "Dunnett")
2) Comparer toutes les moyennes 2 à 2
(cfr test de "Tukey" - "All pairwise comparisons")
Le package multcomp fourni des facilités pour créer automatiquement la matrice de contrastes pour ces cas particuliers

```
> modmc <- glht(mod, linfct = mcp(variety = "Dunnet"))
> summary(modmc)

     Simultaneous Tests for General Linear Hypotheses

Multiple Comparisons of Means: Dunnett Contrasts

Fit: lm(formula = tomato ~ variety, data = d)

Linear Hypotheses:
                 Estimate Std. Error t value Pr(>|t|)
var2 - var1 == 0   -1.582      2.892  -0.547   0.9038
var3 - var1 == 0   -8.145      2.892  -2.816   0.0392 *
var4 - var1 == 0  -14.073      2.892  -4.866   <0.001 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Adjusted p values reported -- single-step method)
```

1 x qualitatif : test de Student - ANOVA

Post-hoc tests - comparaisons multiples

Contrastes de type "Tukey" (toutes les paires de comparaisons)

```
> modmc <- glht(mod, linfct = mcp(variety = "Tukey"))
> summary(modmc)

     Simultaneous Tests for General Linear Hypotheses

Multiple Comparisons of Means: Tukey Contrasts


Fit: lm(formula = tomato ~ variety, data = d)

Linear Hypotheses:
                  Estimate Std. Error t value Pr(>|t|)
var2 - var1 == 0   -1.582      2.892  -0.547  0.94558
var3 - var1 == 0   -8.145      2.892  -2.816  0.06502 .
var4 - var1 == 0  -14.073      2.892  -4.866  0.00189 **
var3 - var2 == 0   -6.563      2.892  -2.269  0.16010
var4 - var2 == 0  -12.491      2.892  -4.319  0.00494 **
var4 - var3 == 0   -5.928      2.892  -2.050  0.22404
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Adjusted p values reported -- single-step method)
```

1 x qualitatif : test de Student - ANOVA

Post-hoc tests - comparaisons multiples

Les résultats sont très proches d'un vrai test de Tukey.
L'avantage du package multcomp est que la méthode peut être généralisée à de nombreux autres cas que l'ANOVA à 1 vacteur

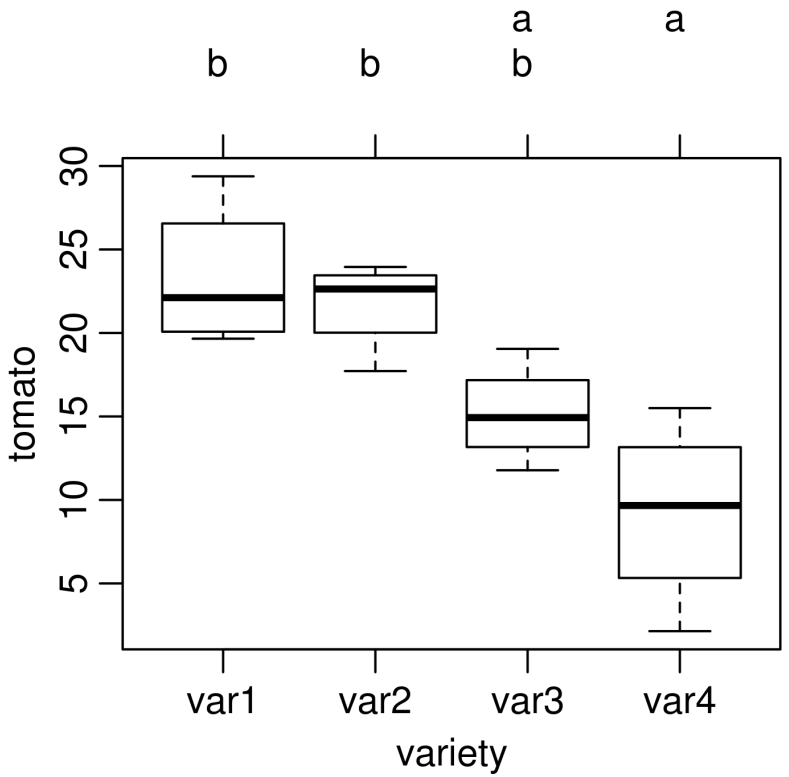
```
> TukeyHSD(aov(mod))
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = mod)

$variety
                diff       lwr        upr     p adj
var2-var1  -1.582049 -10.16762  7.0035232 0.9455739
var3-var1  -8.144824 -16.73040  0.4407475 0.0649556
var4-var1 -14.072785 -22.65836 -5.4872131 0.0018940
var3-var2  -6.562776 -15.14835  2.0227960 0.1600852
var4-var2 -12.490736 -21.07631 -3.9051645 0.0047624
var4-var3  -5.927961 -14.51353  2.6576113 0.2240467
```

1 x qualitatif : test de Student - ANOVA

Post-hoc tests - comparaisons multiples

Classiquement on représente les comparaisons de type "tukey" avec des lettres ("Compact Letters Display", cld)
Les niveaux partageant une même lettre ne sont pas statistiquement différents

```
> cld(modmc)
var1 var2 var3 var4
 "b"  "b" "ab"  "a"
> par(mar = c(3,3,4,3), mgp = c(1.75, 0.6, 0))
> plot(cld(modmc))
```

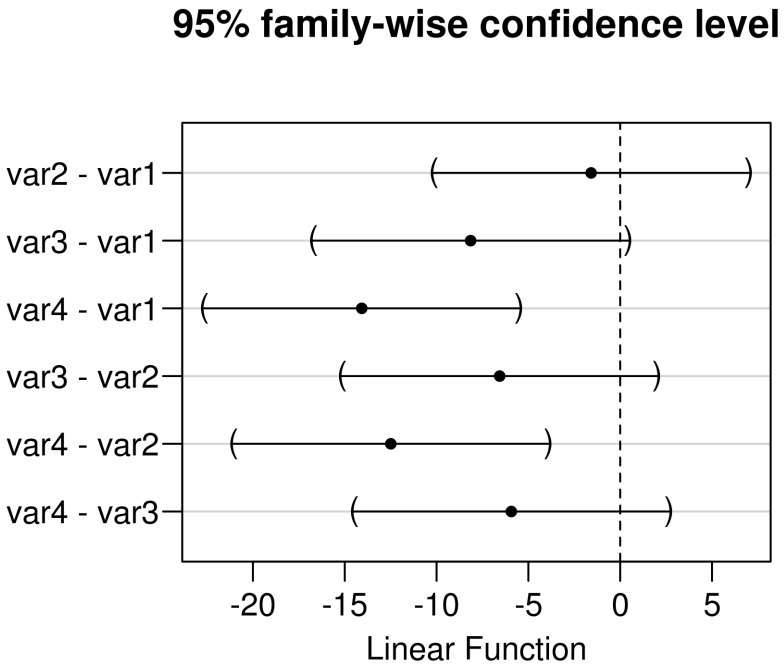


1 x qualitatif : test de Student - ANOVA

Post-hoc tests - comparaisons multiples

On peut aussi obtenir des intervalles de confiance corrigés pour la multiplicité des comparaisons

```
> confint(modmc)

     Simultaneous Confidence Intervals

Multiple Comparisons of Means: Tukey Contrasts


Fit: lm(formula = tomato ~ variety, data = d)

Quantile = 2.9689
95% family-wise confidence level


Linear Hypotheses:
                   Estimate lwr      upr
var2 - var1 == 0  -1.5820 -10.1676   7.0035
var3 - var1 == 0  -8.1448 -16.7304   0.4407
var4 - var1 == 0 -14.0728 -22.6584  -5.4872
var3 - var2 == 0  -6.5628 -15.1483   2.0228
var4 - var2 == 0 -12.4907 -21.0763  -3.9052
var4 - var3 == 0  -5.9280 -14.5135   2.6576
```

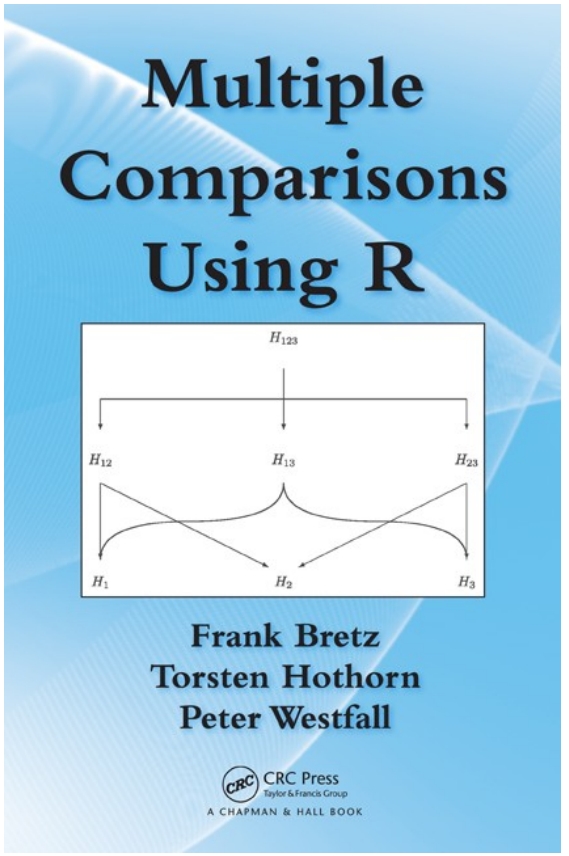



```
> plot(confint(modmc))
```

1 x qualitatif : test de Student - ANOVA

Post-hoc tests - comparaisons multiples

Voir le livre associé au package `multcomp` pour plus de détails.
Bretz et al. 2010



1 x qualitatif : test de Student - ANOVA

Changer le niveau de référence

On peut changer le niveau du facteur utilisé pour l'intercept

```
> mod <- lm(tomato ~  variety, data=d)
> summary(mod)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   23.317      2.045  11.403 8.52e-08 ***
varietyvar2   -1.582      2.892  -0.547 0.594356
varietyvar3   -8.145      2.892  -2.816 0.015561 *
varietyvar4  -14.073      2.892  -4.866 0.000387 ***

> d$variety <- relevel(d$variety, ref = "var4")
> mod <- lm(tomato ~ variety, data=d)
> summary(mod)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)    9.244      2.045   4.521 0.000701 ***
varietyvar1   14.073      2.892   4.866 0.000387 ***
varietyvar2   12.491      2.892   4.319 0.000997 ***
varietyvar3    5.928      2.892   2.050 0.062883 .
```

1 x qualitatif : test de Student - ANOVA

Changer le niveau de référence

On peut changer l'ordre d'affichage des niveaux du facteur

```
> d$variety <- factor(d$variety, levels =
                      c("var3", "var4", "var1", "var2"))
> mod <- lm(tomato ~ variety, data=d)
> summary(mod)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   15.172      2.045   7.420 8.06e-06 ***
varietyvar4   -5.928      2.892  -2.050   0.0629 .
varietyvar1    8.145      2.892   2.816   0.0156 *
varietyvar2    6.563      2.892   2.269   0.0425 *
```

1 x qualitatif : test de Student - ANOVA

Changer la paramétrisation du modèle

On peut obtenir deux modèles identiques, donnant les mêmes prédictions, les mêmes résidus etc... mais dont les paramètres ont une interprétation différente.

Par exemple lorsqu'on retire l'intercept dans un modèle de type ANOVA, le modèle estime les moyennes de chaque groupe ("mean parametrization") au lieu d'estimer les différences de moyennes ("effect parametrization")

1 x qualitatif : test de Student - ANOVA

Changer la paramétrisation du modèle

```
> mod1 <- lm(tomato ~  variety, data=d)
> summary(mod1)
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   23.317      2.045  11.403 8.52e-08 ***
varietyvar2   -1.582      2.892  -0.547 0.594356
varietyvar3   -8.145      2.892  -2.816 0.015561 *
varietyvar4  -14.073      2.892  -4.866 0.000387 ***

Residual standard error: 4.09 on 12 degrees of freedom
Multiple R-squared:  0.714,   Adjusted R-squared:  0.6425
F-statistic: 9.987 on 3 and 12 DF,  p-value: 0.001393
```

"Effect parametrization"

```
> mod2 <- lm(tomato ~ -1 + variety, data=d)
> summary(mod2)
             Estimate Std. Error t value Pr(>|t|)
varietyvar1   23.317      2.045  11.403 8.52e-08 ***
varietyvar2   21.735      2.045  10.629 1.85e-07 ***
varietyvar3   15.172      2.045   7.420 8.06e-06 ***
varietyvar4    9.244      2.045   4.521 0.000701 ***

Residual standard error: 4.09 on 12 degrees of freedom
Multiple R-squared:  0.9637,  Adjusted R-squared:  0.9516
F-statistic: 79.62 on 4 and 12 DF,  p-value: 1.554e-08
```

"Mean parametrization"

```
> anova(mod1, mod2)
  Res.Df    RSS Df Sum of Sq F Pr(>F)
1     12 200.71
2     12 200.71  0         0
```

Les deux modèles sont équivalents...
Mais les inférences ne testent pas les mêmes hypothèses !

Plusieurs x quantitatifs : régression multiple

Concepts à assimiler :

Effet marginal
Comparaison de modèles : Type I vs Type II/Type III
"Overfitting" - R^2 ajusté
Colinéarité/Multicolinéarité entre variables explicatives
Standardisation des variables explicatives

Plusieurs x quantitatifs : régression multiple

Exemple : on veut caractériser l'effet d'un fertilisant sur la production de tomates mais pendant l'expérience, les tomates ont été attaquées par le mildiou qui a provoqué de fortes pertes. On a donc mesuré également l'abondance du mildiou et on va le contrôler statistiquement à défaut d'avoir pu le contrôler expérimentalement.

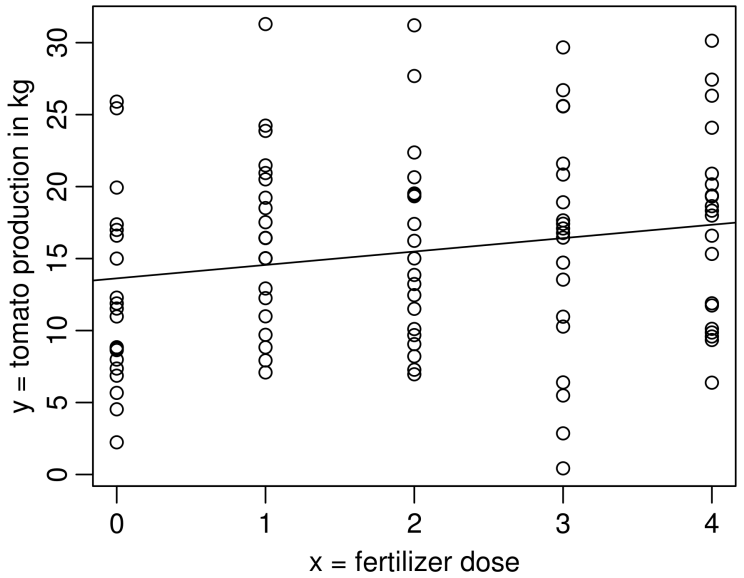
```
n <- 100
beta0 <- 25
beta1 <- 0.5
beta2 <- 5
sigma <- 5

fertilizer <- rep (0:4, each=n/5)
set.seed(1)
mildew <- runif(100,0,4)
set.seed(12)
tomato <- beta0 + beta1*fertilizer -
                  beta2*mildew + rnorm(n,0,sigma)
```

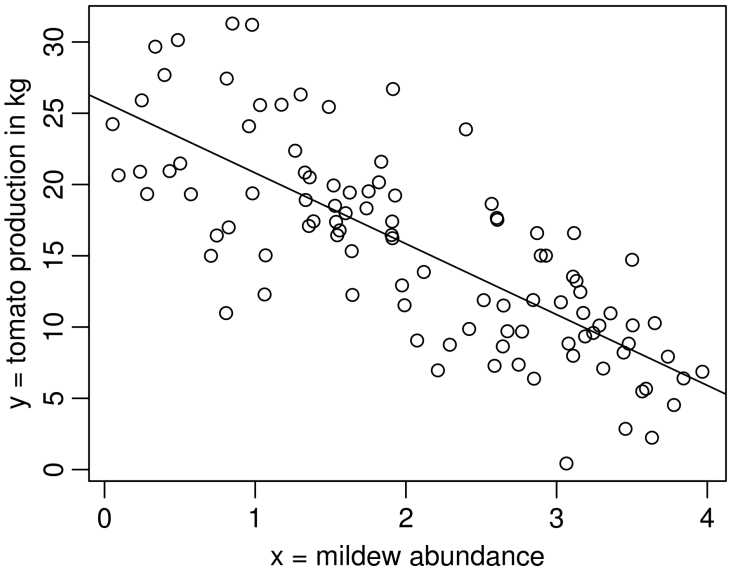
On utilise la distribution uniforme pour générer l'indice d'abondance de mildiou
Toutes les valeurs entre 0 et 4 ont la même probabilité d'être tirées

Plusieurs x quantitatifs : régression multiple

La relation tomates ~ fertilisant est assez faible et de plus les points sont fortement étalés autour de la droite (les résidus sont grands) ce qui masque la relation.
Une grande partie de ce "bruit" est dû à l'effet du mildiou...

```
par(mar = c(3,3,1,1), mgp = c(1.75, 0.6, 0))
plot(tomato ~ fertilizer, ylab =
        "y = tomato production in kg", xlab = "x = fertilizer dose")
abline(lm(tomato ~ fertilizer))
plot(tomato ~ mildew, ylab =
        "y = tomato production in kg", xlab = "x = mildew abundance")
abline(lm(tomato ~ mildew))
```





Plusieurs x quantitatifs : régression multiple

Effet marginal

Dans une régression multiple, l'effet de chaque variable explicative est estimée après avoir "enlevé", "contrôlé" l'effet des autres variables, comme si les autres variables étaient = 0

```
> model <- lm(tomato ~ fertilizer)
> summary(model)
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  13.6255     1.1862  11.486   <2e-16 ***
fertilizer    0.9309     0.4843   1.922   0.0575 .

> model <- lm(tomato ~ fertilizer + mildew)
> summary(model)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  24.0389     1.1420  21.051  < 2e-16 ***
fertilizer    0.8423     0.3071   2.743  0.00726 **
mildew       -4.9417     0.4078 -12.118  < 2e-16 ***
```

Dans une régression simple, l'effet du fertilisant est limite significatif.

Si on ajoute l'effet du mildiou, les erreurs standard sont nettement plus faible et l'effet du fertilisant devient clairement significatif.

On estime que lorsque la dose de fertilisant et l'infestation de mildiou sont nuls, la production moyenne de tomates est de 24.04 kg ± 1.14 kg

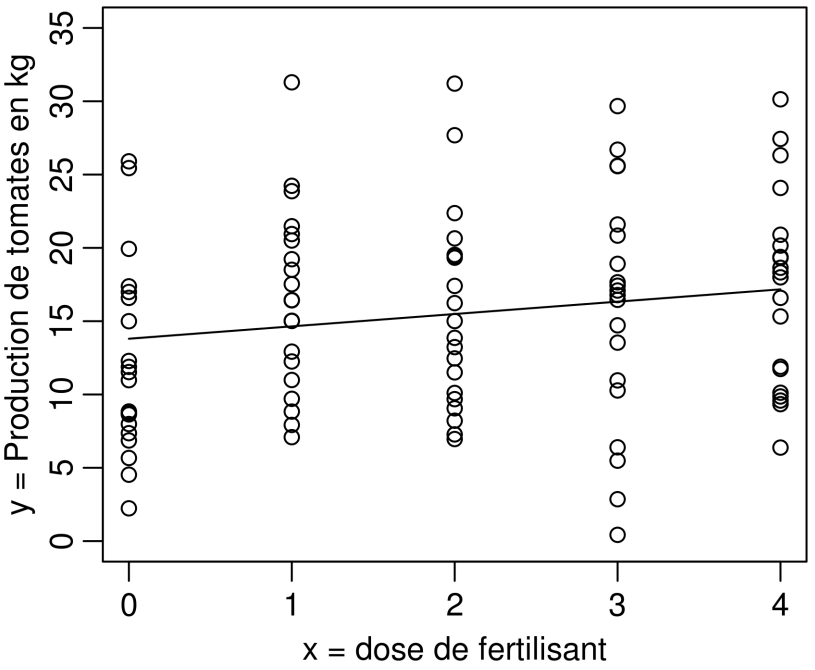
On estime que lorsque on augmente la dose de fertilisant de une unité, et que l'infestation de mildiou est 0, la production de tomate augmente de 0.84 kg ± 0.31 kg. Ici, la pente reste identique quelque soit l'infestation de mildiou (pas d'interaction)

Plusieurs x quantitatifs : régression multiple

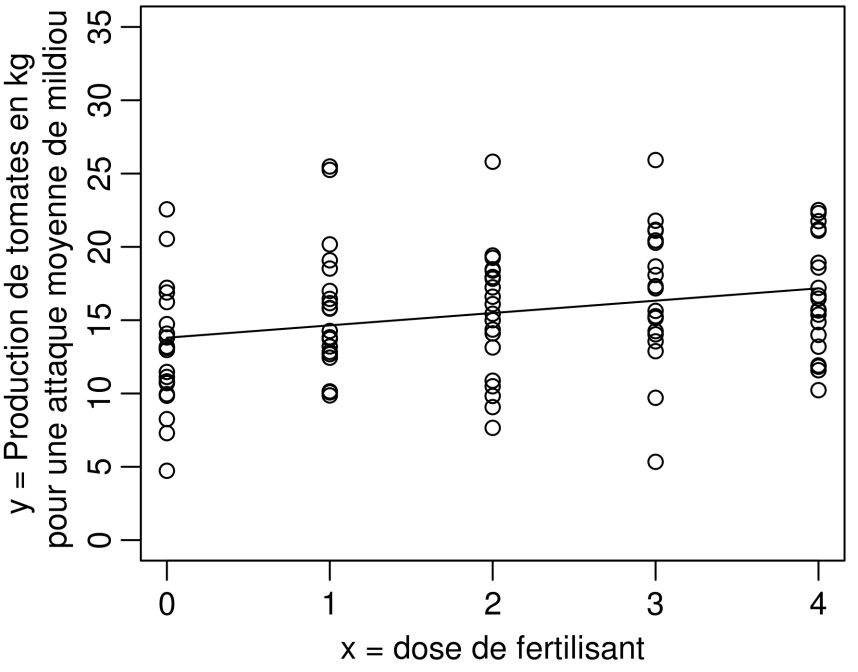
Effet marginal

Dans cet exemple, la régression multiple permet d'éliminer une partie du bruit qui est expliqué par l'effet du mildiou.

Relation tomates ~ fertilisant et données brutes



Même relation avec les données dont on a enlevé l'effet dû au mildiou



Plusieurs x quantitatifs : régression multiple

Effet marginal

On peut estimer à la main approximativement les coefficients de la régression multiple (dans ce cas simple) en utilisant comme variable dépendante les résidus d'une régression simple

```
> model_fertilizer <- lm(tomato ~ fertilizer)
> model_mildew <- lm(tomato ~ mildew)
>
> model <- lm(tomato ~ fertilizer + mildew)
> summary(model)
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  24.0389     1.1420  21.051  < 2e-16 ***
fertilizer    0.8423     0.3071   2.743  0.00726 **
mildew       -4.9417     0.4078 -12.118  < 2e-16 ***

> mod <- lm(resid(model_mildew) ~ fertilizer)
> summary(mod)
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -1.6836     0.7482  -2.250  0.02667 *
fertilizer    0.8418     0.3055   2.756  0.00698 **

> mod <- lm(resid(model_fertilizer) ~ mildew)
> summary(mod)
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  10.2304     0.9451   10.82   <2e-16 ***
mildew       -4.9389     0.4058  -12.17   <2e-16 ***
```

Les résidus de ces 2 modèles représentent les données une fois qu'on a enlevé l'effet du fertilisant ou du mildiou respectivement.

NB : degrés de liberté dans l'approche avec les résidus ne sont pas exact car ils ne prennent pas en compte l'estimation des résidus. Dans cet exemple où n est grand çà ne change pas grand chose

Plusieurs x quantitatifs : régression multiple

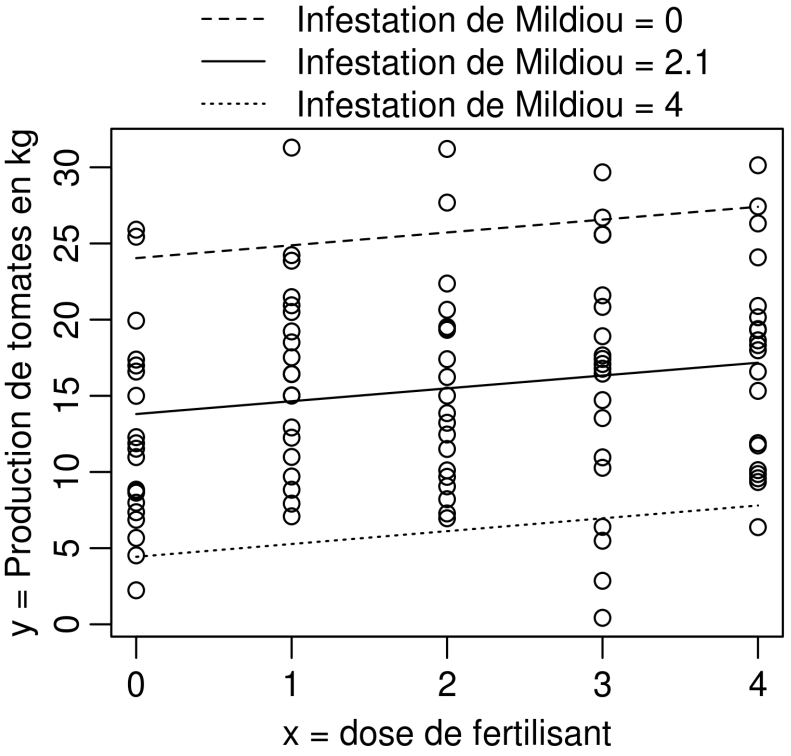
Prédiction et représentation graphique

On va en général représenter y ~ x1 en choisissant une valeur arbitraire pour les autres variables explicatives (souvent la moyenne)

```
(X <- cbind(1, c(0:4), mean(mildew)))
pred <- X %*% coef(mod)

(X <- cbind(1, c(0:4), 0))
pred0 <- X %*% coef(mod)

(X <- cbind(1, c(0:4), max(mildew)))
predmax <- X %*% coef(mod)
```

```
     [,1] [,2]  [,3]
[1,]    1    0 2.071
[2,]    1    1 2.071
[3,]    1    2 2.071
[4,]    1    3 2.071
[5,]    1    4 2.071
```

3 prédictions différentes




```
par(mar = c(3,4,4,2), mgp = c(1.75, 0.6, 0))
plot(tomato ~ fertilizer,
     ylab = "y = Production de tomates en kg",
     xlab = "x = dose de fertilisant")
lines(x= X[,2], y = pred, lty = 1)
lines(x= X[,2], y = pred0, lty = 2)
lines(x= X[,2], y = predmax, lty = 3)

legend(x = "top", inset = -0.3, xpd = NA, bty = "n", lty = c(2,1,3),
       legend = paste0("Infestation de Mildiou = ", round( c(0,mean(mildew), max(mildew)),1))
       )
```

Plusieurs x quantitatifs : régression multiple

Comparaison de modèles : Type I vs Type II/III

Dès que l'on a plusieurs variables explicatives, il existe plusieurs manières de faire des comparaisons de modèles emboîtés.

La fonction `anova()` calcule des "Sum of Square de type I"
Elle réalise une comparaison séquentielle des modèles.
Chaque effet est testé marginalement au précédent...
En conséquence l'ordre des variables change généralement les p valeurs*
--> ce n'est clairement pas ce qu'on veut tester en général

A éviter dans la plupart des cas !

```
> anova(lm(tomato ~ fertilizer + mildew))

Response: tomato
           Df  Sum Sq Mean Sq  F value    Pr(>F)
fertilizer  1  173.31  173.31   9.1929  0.003116 **
mildew      1 2768.14 2768.14 146.8352 < 2.2e-16 ***
Residuals  97 1828.65   18.85

> anova(lm(tomato ~ mildew + fertilizer))

Response: tomato
           Df Sum Sq Mean Sq F value  Pr(>F)
mildew      1 2799.6 2799.64 148.506 < 2e-16 ***
fertilizer  1  141.8  141.80   7.522 0.00726 **
Residuals  97 1828.7   18.85
```


*sauf si les variables explicatives sont parfaitement indépendantes et que le design est parfaitement balancé

Plusieurs x quantitatifs : régression multiple

Comparaison de modèles : Type I vs Type II/III

La fonction `drop1(mod, test="F")` calcule des "Sum of Square de type II/III"
Chaque effet est testé marginalement aux autres variables.
En conséquence l'ordre des variables ne change jamais les p valeurs
--> c'est presque toujours cette approche qui est l'approche désirée

```
> drop1(lm(tomato ~ fertilizer + mildew), test = "F")

           Df Sum of Sq    RSS    AIC F value  Pr(>F)
<none>                  1828.6 296.62
fertilizer  1     141.8 1970.5 302.08   7.522 0.00726 **
mildew      1    2768.1 4596.8 386.79 146.835 < 2e-16 ***

> drop1(lm(tomato ~ mildew + fertilizer), test = "F")

           Df Sum of Sq    RSS    AIC F value  Pr(>F)
<none>                  1828.6 296.62
mildew      1    2768.1 4596.8 386.79 146.835 < 2e-16 ***
fertilizer  1     141.8 1970.5 302.08   7.522 0.00726 **
```

On peut recalculer ces valeurs avec des comparaisons de modèles emboîtés pour comprendre ce que fait cette fonction

```
> anova(lm(tomato ~ mildew), lm(tomato ~ fertilizer+ mildew))
  Res.Df    RSS Df Sum of Sq     F  Pr(>F)
2     97 1828.7  1     141.8 7.522 0.00726 **
```

test effet "fertilizer"

```
> anova(lm(tomato ~ fertilizer), lm(tomato ~ fertilizer+ mildew))
  Res.Df    RSS Df Sum of Sq      F    Pr(>F)
2     97 1828.6  1    2768.1 146.84 < 2.2e-16 ***
```

test effet "mildew"

Plusieurs x quantitatifs : régression multiple

Comparaison de modèles : Type I vs Type II/III

La fonction `Anova()` du package `car` calcule des "Sum of Square de type II" (par défaut) ou de type III (voir plus loin) et donne ici les mêmes résultats que `drop1`.
Les comparaisons de type II et III donnent les mêmes résultats, tant qu'il n'y a pas d'interaction.

```
> library(car)
> Anova(lm(tomato ~ fertilizer + mildew))
Anova Table (Type II tests)

            Sum Sq Df F value  Pr(>F)
fertilizer  141.8  1   7.522 0.00726 **
mildew     2768.1  1 146.835 < 2e-16 ***
Residuals  1828.7 97

> Anova(lm(tomato ~ mildew + fertilizer))
Anova Table (Type II tests)
            Sum Sq Df F value  Pr(>F)
mildew     2768.1  1 146.835 < 2e-16 ***
fertilizer  141.8  1   7.522 0.00726 **
Residuals  1828.7 97
```

NB : la terminologie "Type I", "Type II", "Type III" "Sum of Squares" est héritée au départ de certains logiciels (ea SAS). Ces termes sont cependant assez vagues et ne correspondent pas toujours à la même chose (en particulier type II et type III).
Cette terminologie est en général évitée dans R où on construit soi-même les modèles à comparer et où on les compare avec `anova()`

Plusieurs x quantitatifs : régression multiple

"Overfitting" - R^2 ajusté

Lorsque le nombre de variables explicatives augmente par rapport aux nombre de données, les erreurs standard augmentent et le R^2 augmente jusqu'à atteindre 1 quand on a autant de variables explicatives que de données.

Le modèle prédit alors parfaitement les données mais n'est pas généralisable à un autre jeu de données.

Lorsque le modèle est "overfité" (= surparamétrisé?), des variables explicatives qui on un réel lien avec la variable dépendante peuvent devenir non significatives

On applique en général des méthodes de sélection de modèle qui vont d'une manière ou d'une autre déduire les variables explicatives les plus importantes et réduire les dimensions du modèle (voir fin du module 3)

Plusieurs x quantitatifs : régression multiple

"Overfitting" - R^2 ajusté

```
> set.seed(12)
> d <- as.data.frame(matrix(runif(100, 0, 1), 10,10))
> colnames(d) <- paste0("x", 1:10)
> d$y <- 6*d$x1 + rnorm(10,0,2)
> res <- list(
+     summary(lm(y ~ x1, data=d)),
+     summary(lm(y ~ x1 + x2 , data=d)),
+     summary(lm(y ~ x1 + x2 + x3 , data=d)),
+     summary(lm(y ~ x1 + x2 + x3 + x4 , data=d)),
+     summary(lm(y ~ x1 + x2 + x3 + x4 + x5 , data=d)),
+     summary(lm(y ~ x1 + x2 + x3 + x4 + x5 + x6 , data=d)),
+     summary(lm(y ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 , data=d)),
+     summary(lm(y ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 , data=d)),
+     summary(lm(y ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9 , data=d)),
+     summary(lm(y ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9 + x10, data=d))
+ )
```

On crée 10 variables explicatives mais seule x1 explique y.
le nombre d'observations = 10

```
> # tests pour x1 dans chacun des 10 modèles
> t(sapply(res, function(x) x$coefficients[2,]))
       Estimate Std. Error    t value   Pr(>|t|)
 [1,]   5.224982   1.870292  2.7936720 0.02342528
 [2,]   5.529417   2.169264  2.5489828 0.03816023
 [3,]   6.118395   2.073810  2.9503160 0.02560246
 [4,]   4.719254   2.560075  1.8434049 0.12460084
 [5,]   5.163240   4.091366  1.2619843 0.27552470
 [6,]   3.181097   5.063452  0.6282467 0.57441594
 [7,]  -9.577013  11.699751 -0.8185656 0.49905067
 [8,] -11.459542  16.781885 -0.6828519 0.61858566
 [9,]  -7.778722        NaN        NaN        NaN
[10,]  -7.778722        NaN        NaN        NaN
```

test pour x1 dans chacun des 10 modèles

Plusieurs x quantitatifs : régression multiple

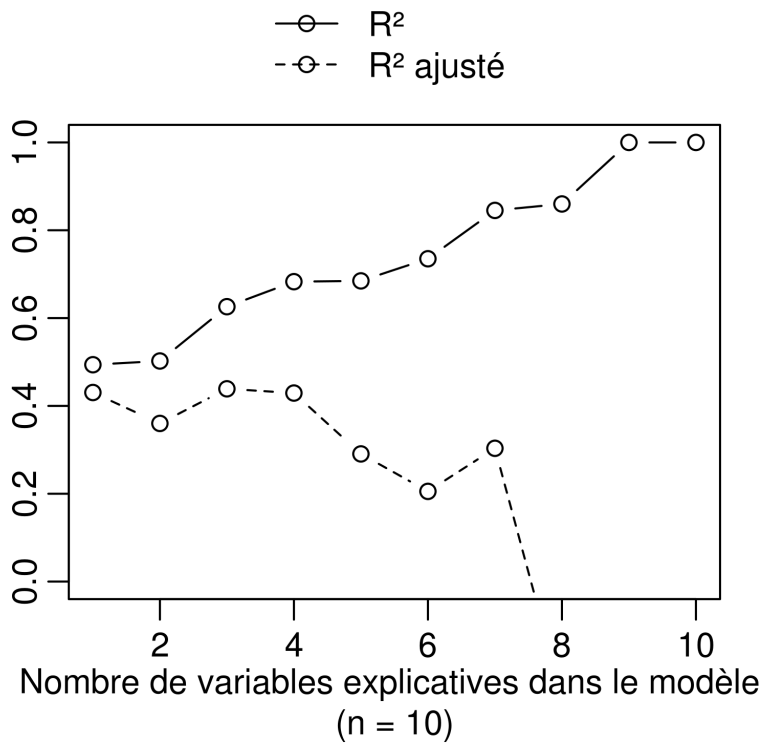
"Overfitting" - R² ajusté

Il existe une version ajustée du R² qui permet de comparer des modèles avec un nombre différent de paramètres et de données

$$R^2_{adj} = R^2 - \left(1 - R^2\right)\frac{p}{n - p - 1}$$

p représente le nombre de paramètres sans l'intercept et la variance résiduelle

```
> summary(lm(y ~ x1 + x2 + x3 + x4 + x5 , data=d))

Residual standard error: 2.203 on 4 degrees of freedom
Multiple R-squared:  0.6848, Adjusted R-squared:  0.2907
F-statistic: 1.738 on 5 and 4 DF,  p-value: 0.3062

> # extraction du R²
> (Rsq <- sapply(res, function(x) x$r.squared))
 [1] 0.4938184 0.5023192 0.6259839 0.6829472 0.6847550
0.7351300 0.8452258 0.8599036 1.0000000 1.0000000
> (Rsqadj <- sapply(res, function(x) x$adj.r.squared))
 [1]  0.4305456  0.3601247  0.4389759  0.4293050
0.2906988  0.2053899  0.3035160 -0.2608676        NaN
[10]        NaN
```



NB : l'idéal pour comparer des modèles est en général d'utiliser des méthodes par "critère d'information" (ea : AIC) que nous verrons plus loin

Plusieurs x quantitatifs : régression multiple

Colinéarité entre variables explicatives

En régression multiple, on estime se qui se passe pour y quand x1 augmente d'une unité et que les autres variables explicatives restent inchangées ce qui en fait n'est pas possible quand ces variables sont corrélées entre elles.

Lorsque les variables explicatives sont corrélées entre elles, on rencontre plusieurs problèmes :

1) les erreurs standard des paramètres augmentent

2) on peut conclure que certaines variables n'ont pas d'effet alors qu'elles ont un effet prises individuellement mais pas une fois qu'on a enlevé l'effet des autres variables avec lesquelles elle sont corrélées

Plusieurs x quantitatifs : régression multiple

Colinéarité entre variables explicatives

Tant qu'elle n'est pas excessive, le fait que la corrélation entre variables explicatives soit un problème ou pas va dépendre de la question posée et de l'interprétation que l'on veut faire des données.

On peut par exemple utiliser la décomposition de la variance pour montrer la proportion de la variance expliquée par chaque (groupe de) variable(s) individuellement ou en commun (voir plus loin)

Plusieurs x quantitatifs : régression multiple

Colinéarité entre variables explicatives

Exemple fictif : On évalue la relation entre la hauteur d'une personne et la longueur de son bras droit et de son bras gauche...
Ces 2 variables explicatives ont une corrélation de 0.994

```
> x1 <- 1:30
> x2 <- x1 + rnorm(30,0,1)
> cor(x1,x2)
[1] 0.9944307
>
> y <- 3 * x1 +  rnorm(30,0,5)

> summary(lm(y ~ x1))
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -3.44910    1.56648  -2.202   0.0361 *
x1           2.17374    0.08824  24.635   <2e-16 ***

> summary(lm(y ~ x2))
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -3.33677    1.71456  -1.946   0.0617 .
x2           2.14809    0.09579  22.426   <2e-16 ***

> summary(lm(y ~ x1 + x2))
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -3.4381     1.5967  -2.153   0.0404 *
x1            2.3094     1.0023   2.304   0.0291 *
x2           -0.1352     0.9949  -0.136   0.8929
```

Prises individuellement, la longueur du bras gauche et du bras droit sont évidemment fortement liées à la hauteur du corps

Si on les met dans le même modèle : les erreurs standard sont multipliées par 10 et on serait tenté de dire que la hauteur du corps est liée à la longueur du bras gauche mais pas du bras droit, ce qui est faux !

Plusieurs x quantitatifs : régression multiple

Colinéarité entre variables explicatives

Les erreurs standard sont multipliées par la racine carrée des "VIFs" ("Variance Inflation Factors") qui sont calculés sur base des R^2 de régressions de chaque variable explicative en fonction des autres.

```
> library(car)
> mod <- lm(y ~ x1 + x2)
> vif(mod)
      x1       x2
124.4989 124.4989
```

Plusieurs x quantitatifs : régression multiple

Colinéarité entre variables explicatives

Ce ne sont en effet pas les corrélations 2 à 2 entre variables explicatives qui sont importantes (colinéarité) mais bien leur corrélation multiple (multicolinéarité).

Si une variable explicative est parfaitement prédite par une combinaison des autres ($R^2=1$), son coefficient n'est pas estimable

```
> x1 <- rep(1:5, each = 10)
> x2 <- rep(1:5, times = 10)
> x3 <- x1 + x2 + rnorm(50,0,0.1)
> cor(cbind(x1,x2,x3))
          x1        x2        x3
x1 1.0000000 0.0000000 0.7067394
x2 0.0000000 1.0000000 0.7058506
x3 0.7067394 0.7058506 1.0000000
>
> set.seed(1)
> y <- 2 * x1 + 3*x2+ rnorm(50,0,5)
```

```
> mod <- lm(y ~ x1 + x2 )
> summary(mod)

             Estimate Std. Error t value Pr(>|t|)
(Intercept)    0.4899     1.8973   0.258    0.797
x1             1.9379     0.4242   4.568 3.57e-05 ***
x2             3.0662     0.4242   7.228 3.70e-09 ***

> mod <- lm(y ~ x1 + x2 + x3)
> summary(mod)
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    0.5545     1.9320   0.287    0.775
x1             3.6020     6.3369   0.568    0.573
x2             4.7282     6.3289   0.747    0.459
x3            -1.6705     6.3469  -0.263    0.794

> sqrt(vif(mod))
      x1       x2       x3
14.78828 14.76976 20.87676
```

Plusieurs x quantitatifs : régression multiple

Colinéarité entre variables explicatives

Que faire ?

Une fois qu'on a choisi les variables explicatives pertinentes, toujours examiner leurs corrélation par un scatterplot matrix et/ou les VIFs afin de repérer les problèmes potentiels.

Si des variables corrélées posent problème :

- les grouper : pex prendre la moyenne des bras gauche et droit, sommer leurs valeurs,...
- n'en garder qu'une, celle qui est la plus biologiquement pertinente et garder en mémoire pour l'interprétation que toute relation pourrait être due en fait aux variables que l'on a retirées
- utiliser les composantes principales d'une PCA : l'avantage est qu'on garde toute l'info mais le gros inconvénient est l'interprétation biologique qui est plus difficile

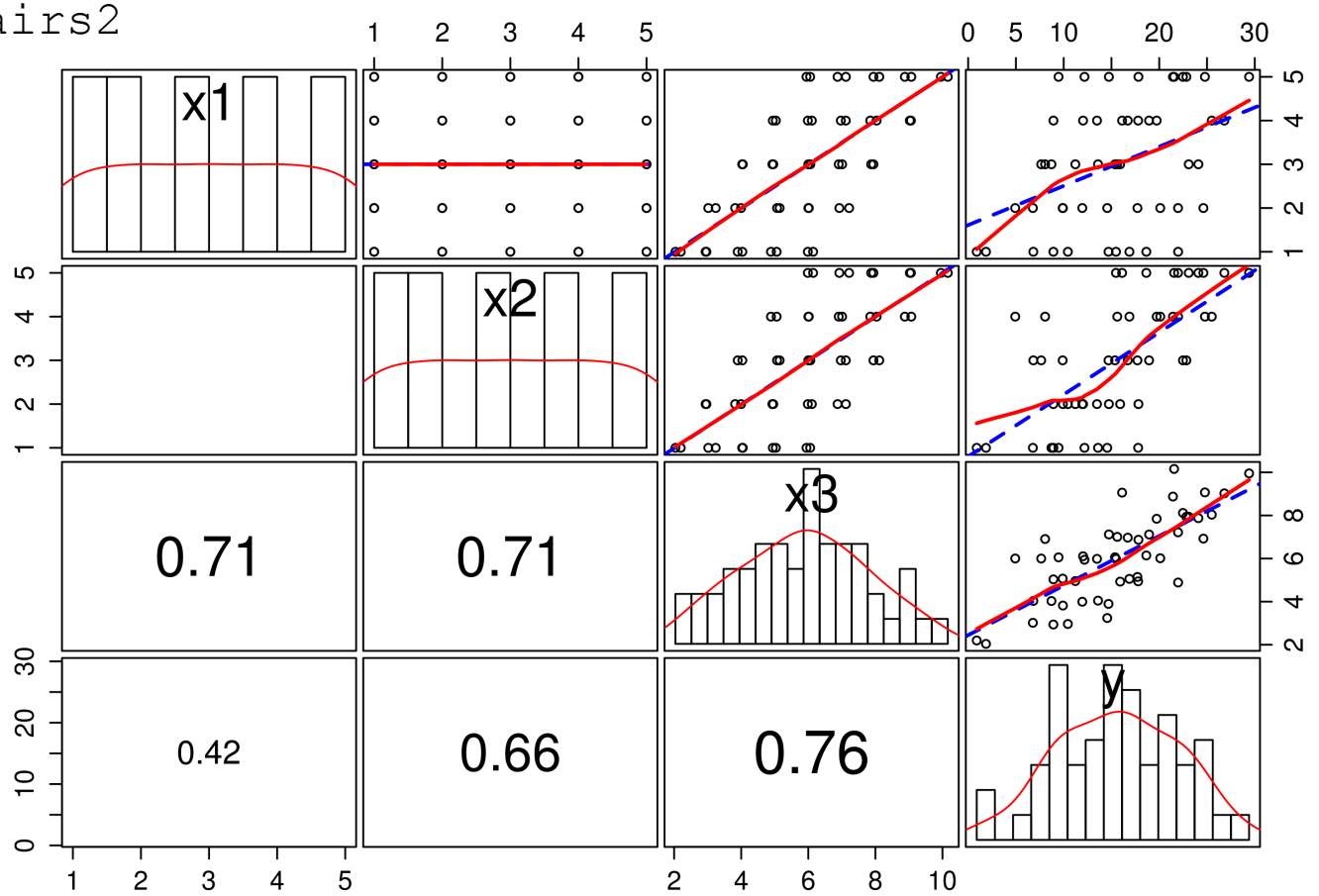
Plusieurs x quantitatifs : régression multiple

Colinéarité entre variables explicatives

Que faire ?

mettre en mémoire un script externe contenant la fonction personnalisée `pairs2`

```
> source("/home/gilles/stats/mytoolbox.R")
> pairs2(cbind(x1, x2, x3, y))
```



Plusieurs x quantitatifs : régression multiple

Quand faut-il standardiser les variables explicatives ?

La standardisation est utile quand on veut pouvoir comparer les coefficients de variables avec des unités différentes ou une même unité mais une variabilité différente.

Standardiser = (en général) soustraire la moyenne et diviser par l'écart-type. On obtient alors des variables directement comparables, sans unités. Quand la variable standardisée augmente de 1 unité, elle augmente de 1 écrat-type sur l'échelle d'origine.

Plusieurs x quantitatifs : régression multiple

Quand faut-il standardiser les variables explicatives ?

Exemple : on étudie une variable y en fonction de la hauteur de la végétation (en mètres), de la distance au site le plus proche (en mètres) et de la surface du site occupé (en m²)

```
n <- 30
set.seed(1)
hauteur <- runif(n, 0, 1)
set.seed(2)
distance <- runif(n, 100, 5000)
set.seed(3)
surface <- runif(n, 1000, 250000)

y <- 500 - 500 * hauteur + 3 * distance + 1 * surface + rnorm(n, 0, 250)
d <- data.frame(y = y, hauteur = hauteur, distance = distance , surface = surface)

> mod <- lm (y ~ hauteur + distance + surface, data = d)
> options(scipen = 3)
> summary(mod)

                Estimate   Std. Error  t value Pr(>|t|)
(Intercept)  528.5391468  130.7107795    4.044 0.000417 ***
hauteur     -535.4521010  149.5427467   -3.581 0.001381 **
distance       2.9985205    0.0303226   98.887  < 2e-16 ***
surface        1.0001021    0.0006405 1561.524  < 2e-16 ***
```

Pour éviter la notation scientifique

Plusieurs x quantitatifs : régression multiple

Quand faut-il standardiser les variables explicatives ?

Si on change les unités, les inférences restent les mêmes (regarder les t values) mais les coefficients changent...
On transforme pex. la distance en km et la surface en ha

```
> d2 <- d
> d2$distance <- d2$distance / 1000
> d2$surface <- d2$surface / 10000
> mod <- lm (y ~ hauteur + distance + surface, data = d2)
> options(scipen = 2)
> summary(mod)

Call:
lm(formula = y ~ hauteur + distance + surface, data = d2)
             Estimate Std. Error  t value Pr(>|t|)
(Intercept)   528.539    130.711    4.044 0.000417 ***
hauteur      -535.452    149.543   -3.581 0.001381 **
distance     2998.521     30.323   98.887  < 2e-16 ***
surface     10001.021      6.405 1561.524  < 2e-16 ***
```

Plusieurs x quantitatifs : régression multiple

Quand faut-il standardiser les variables explicatives ?

Si on standardise les variables explicatives, on peut comparer leurs coefficients mais l'interprétation biologique est plus délicate (les unités sont en quelque sorte des "écarts types")

```
> d3 <- d
> d2[,-1] <- scale(d2[,-1])
> mod <- lm (y ~ hauteur + distance + surface, data = d2)
> options(scipen = 2)
> summary(mod)

             Estimate Std. Error  t value Pr(>|t|)
(Intercept) 130278.56      42.04 3098.851  < 2e-16 ***
hauteur       -158.11      44.16   -3.581  0.00138 **
distance      4356.42      44.05   98.887  < 2e-16 ***
surface      67514.44      43.24 1561.524  < 2e-16 ***
```

Plusieurs x quantitatifs et/ou qualitatifs : ANCOVA

Plusieurs x quantitatifs et/ou qualitatifs : ANCOVA

Concepts à assimiler

Interprétation géométrique - paramétrisation

Interactions

Syntaxe des formules de modèles

Comparaisons de modèles : Règle de marginalité et conséquences quand on ne la respecte pas (type I vs type II vs Type III tests)

Plusieurs x quantitatifs et/ou qualitatifs : ANCOVA

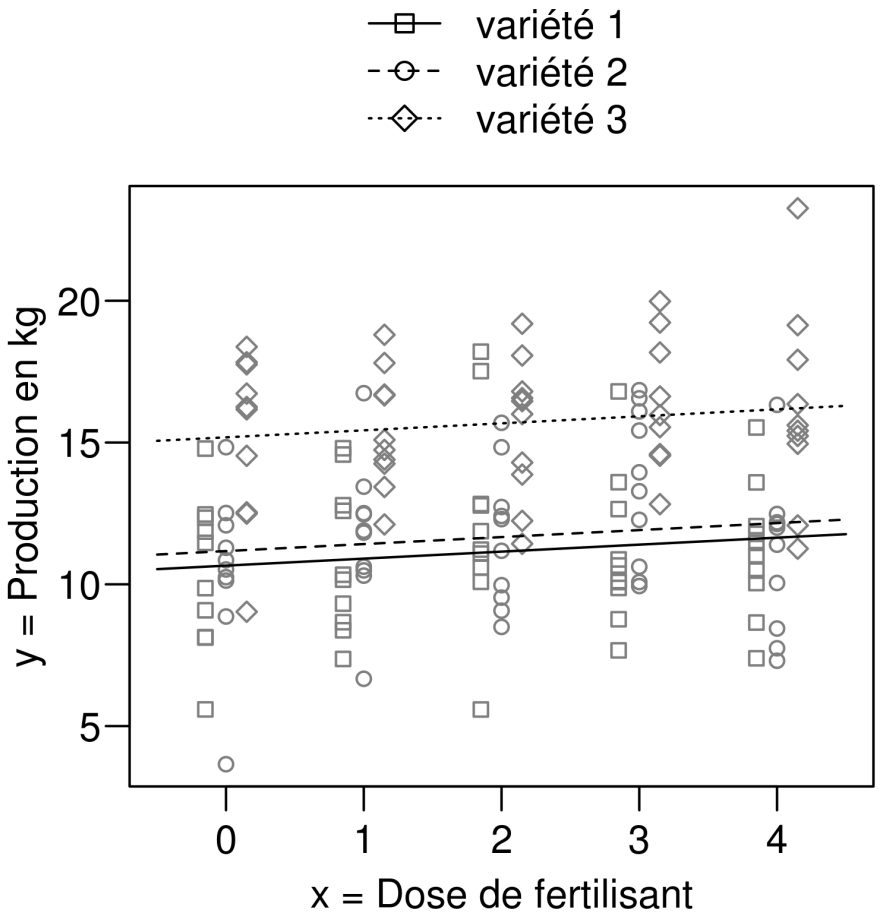
Effet de la dose de fertilisant sur 3 variétés de tomates sans interaction

Génération du jeu de données

```
> fertilizer <- rep (0:4, each=30)
> variety <- as.factor(rep(c(1, 2, 3), 50))
>
> B <- c( 10, 0.5 , 0.3, 5)
> X <- model.matrix(~ fertilizer + variety)
> set.seed(1)
> tomato <- X %*% B + rnorm(150, 0, 3)

> d <- data.frame(tomato = tomato, fertilizer=fertilizer, variety=variety)
> d
       tomato fertilizer variety
1    8.120639          0       1
2   10.850930          0       2
3   12.493114          0       3
4   14.785842          0       1
5   11.288523          0       2
6   12.538595          0       3
(...)
31  14.576039          1       1
32  10.491637          1       2
33  16.663015          1       3
34  10.338585          1       1
(...)
61  18.204853          2       1
62  11.182280          2       2
```

Plusieurs x quantitatifs et/ou qualitatifs : ANCOVA

```
> mod <- lm( tomato ~ fertilizer + variety , data=d)
> summary(mod)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  10.6585     0.4930  21.619  < 2e-16 ***
fertilizer    0.2467     0.1559   1.582    0.116
variety2      0.5146     0.5401   0.953    0.342
variety3      4.5257     0.5401   8.380 4.05e-14 ***
```



Plusieurs x quantitatifs et/ou qualitatifs : ANCOVA

```
> mod <- lm( tomato ~ fertilizer + variety , data=d)
> summary(mod)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  10.6585     0.4930  21.619  < 2e-16 ***
fertilizer    0.2467     0.1559   1.582    0.116
variety2      0.5146     0.5401   0.953    0.342
variety3      4.5257     0.5401   8.380 4.05e-14 ***

Residual standard error: 2.7 on 146 degrees of freedom
Multiple R-squared:  0.3726,  Adjusted R-squared:  0.3597
F-statistic:  28.9 on 3 and 146 DF,  p-value: 9.984e-15

> drop1(mod, test = "F")
Single term deletions

Model:
tomato ~ fertilizer + variety
           Df Sum of Sq    RSS    AIC F value    Pr(>F)
<none>                  1064.6 301.96
fertilizer  1     18.25 1082.9 302.51  2.5034    0.1158
variety     2    613.92 1678.6 366.26 42.0951 3.676e-15 ***
```

Intercept :
la production moyenne de la variété 1 pour une dose de fertilisant nulle est estimée à 10.66kg

fertilizer : lorsque la dose de fertilisant augmente d'une unité, la production augmente de 0.25 kg, pour la variété 1 (mais aussi pour toutes les variétés car il n'y a pas d'interaction)

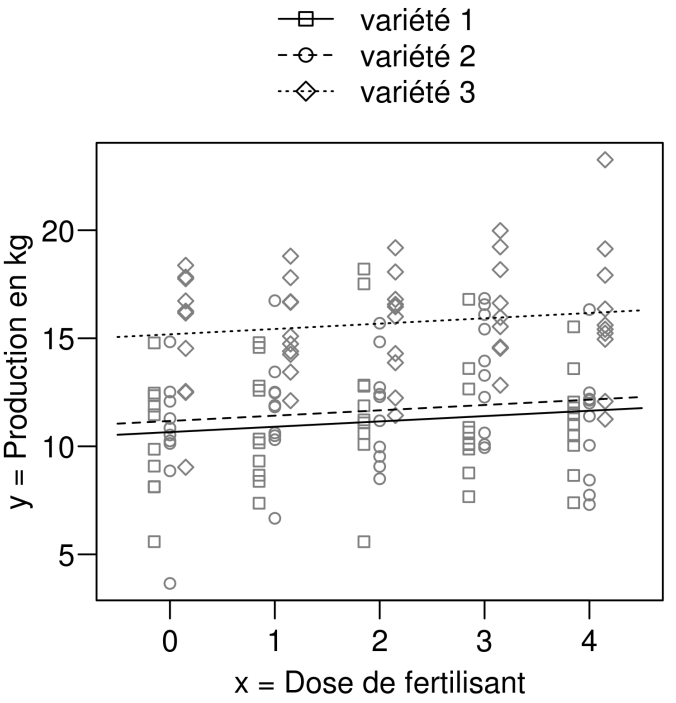
variety2 et varety3 représentent la différence d'intercept.
La variété 2 produirait en moyenne 0.52 kg en plus que la variété 1 quand la dose de fertilisant est nulle mais ceci reste valable quelle que soit la dose de fertilisant car il n'y a pas d'interaction

Teste l'effet global : Est-ce qu'au moins une variété a une productivité différente des autres ?

Plusieurs x quantitatifs et/ou qualitatifs : ANCOVA

```
dev.new(10/2.54, 10/2.54)
par(mar = c(3,3,4,1), mgp = c(1.75, 0.6, 0), las = 1)
plot(tomato ~ fertilizer, type = "n", xlim = c(-0.5, 4.5),
     ylab = "y = Production en kg", xlab = "x = Dose de fertilisant")

points(y = d[d$variety==1, "tomato"],
        x = d[d$variety==1,"fertilizer"] - 0.15, pch = 0, cex =0.8)
points(y = d[d$variety==2, "tomato"],
        x = d[d$variety==2,"fertilizer"], pch = 1, cex =0.8)
points(y = d[d$variety==3, "tomato"],
        x = d[d$variety==3,"fertilizer"] + 0.15, pch = 5, cex =0.8)
```

```
X1 <- cbind(1, c(-0.5,4.5), 0, 0)
X2 <- cbind(1, c(-0.5,4.5), 1, 0)
X3 <- cbind(1, c(-0.5,4.5), 0, 1)

pred1 <- X1 %*% coef(mod)
pred2 <- X2 %*% coef(mod)
pred3 <- X3 %*% coef(mod)

lines(y = pred1, x = X1[,2], lty = 1)
lines(y = pred2, x = X2[,2], lty = 2)
lines(y = pred3, x = X3[,2], lty = 3)
```




```
legend(x = "top", inset = -0.35, xpd = NA, bty = "n", lty = 1:3, pch = c(0,1,5),
       legend = c("variété 1", "variété 2", "variété 3"))
```

Interactions

On parle d'interaction quand l'effet d'une variable explicative dépend de la valeur d'une autre variable explicative.

Pex : la relation entre la quantité de fertilisant et la production de tomates pourrait être positive pour une variété, négative pour une autre et nulle pour une troisième.

La production pourrait aussi augmenter de 1 kg quand la dose augmente de 1 unité de fertilisant pour une variété et seulement de 0.8 kg pour une autre variété.

Interactions

Effet de la dose de fertilisant sur 3 variétés de tomates avec interaction

Génération du jeu de données

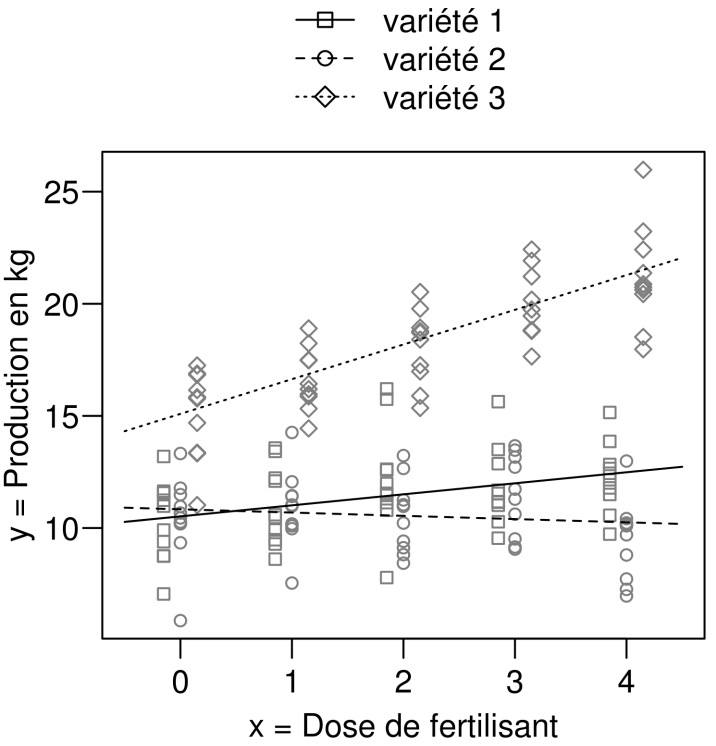
```
> fertilizer <- rep (0:4, each=30)
> variety <- as.factor(rep(c(1, 2, 3), 50))
>
> B <- c( 10, 0.7 , 0.3, 5, -0.7, 1)
> X <- model.matrix(~ fertilizer + variety + fertilizer:variety)
> X[145:150,]
    (Intercept) fertilizer variety2 variety3 fertilizer:variety2 fertilizer:variety3
145           1          4        0        0                    0                    0
146           1          4        1        0                    4                    0
147           1          4        0        1                    0                    4
148           1          4        0        0                    0                    0
149           1          4        1        0                    4                    0
150           1          4        0        1                    0                    4
> set.seed(1)
> tomato <- X %*% B + rnorm(150, 0, 2)
>
> d <- data.frame(tomato = tomato, fertilizer=fertilizer, variety=variety)
> d
       tomato fertilizer variety
1    8.747092          0       1
2   10.667287          0       2
3   13.328743          0       3
4   13.190562          0       1
5   10.959016          0       2
6   13.359063          0       3
7   10.974858          0       1
```

La colonne correspondant à l'interaction fertilisant : variété 2 contient le produit des x fertilisant * variété 2

Interactions

Ce modèle correspond à 3 droites d'intercept et de pente différents.
On estime l'intercept et la pente de la variété 1 ainsi que les différences d'intercept et de pente avec les autres variétés

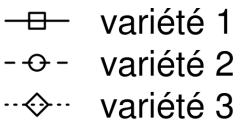
```
> mod <- lm( tomato ~ fertilizer + variety + fertilizer:variety, data=d)
> summary(mod)

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)          10.5183     0.4439  23.695  < 2e-16 ***
fertilizer            0.4915     0.1812   2.712   0.0075 **
variety2              0.3163     0.6278   0.504   0.6151
variety3              4.5727     0.6278   7.284 1.96e-11 ***
fertilizer:variety2  -0.6366     0.2563  -2.484   0.0141 *
fertilizer:variety3   1.0556     0.2563   4.119 6.40e-05 ***
```



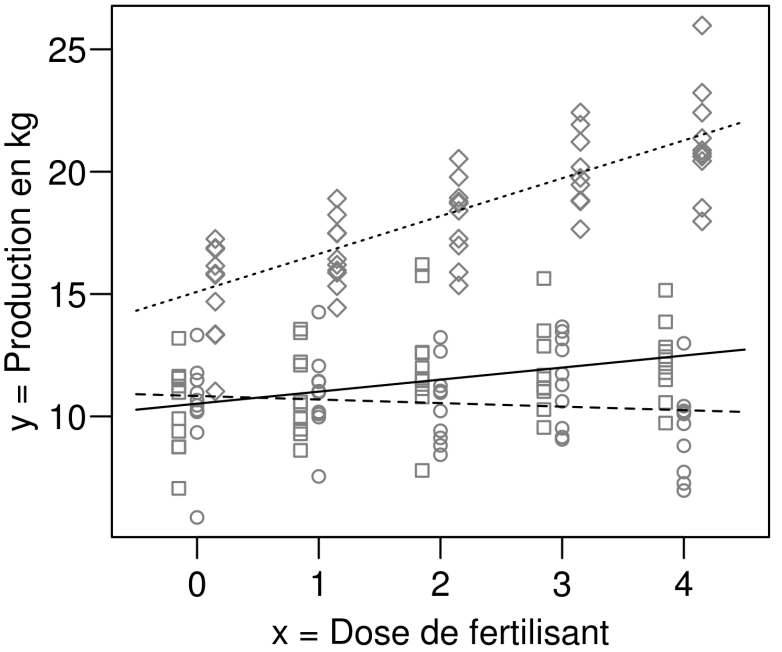
Interactions

Interprétation

```
> mod <- lm( tomato ~ fertilizer + variety + fertilizer:variety, data=d)
> summary(mod)

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)          10.5183     0.4439  23.695  < 2e-16 ***
fertilizer            0.4915     0.1812   2.712   0.0075 **
variety2              0.3163     0.6278   0.504   0.6151
variety3              4.5727     0.6278   7.284 1.96e-11 ***
fertilizer:variety2  -0.6366     0.2563  -2.484   0.0141 *
fertilizer:variety3   1.0556     0.2563   4.119 6.40e-05 ***
```



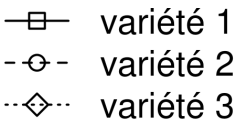
On estime que la variété 1 produit 10.52 kg de tomates pour une dose de fertilisant de 0 (Intercept) et que lorsqu'on augmente la dose de fertilisant d'une unité, la production augmente de 0.49 kg (coefficient "fertilizer") uniquement pour la variété 1.
Il est peu vraisemblable d'obtenir de telles valeurs uniquement par hasard (p<0.0001 et p = 0.0075)



Interactions

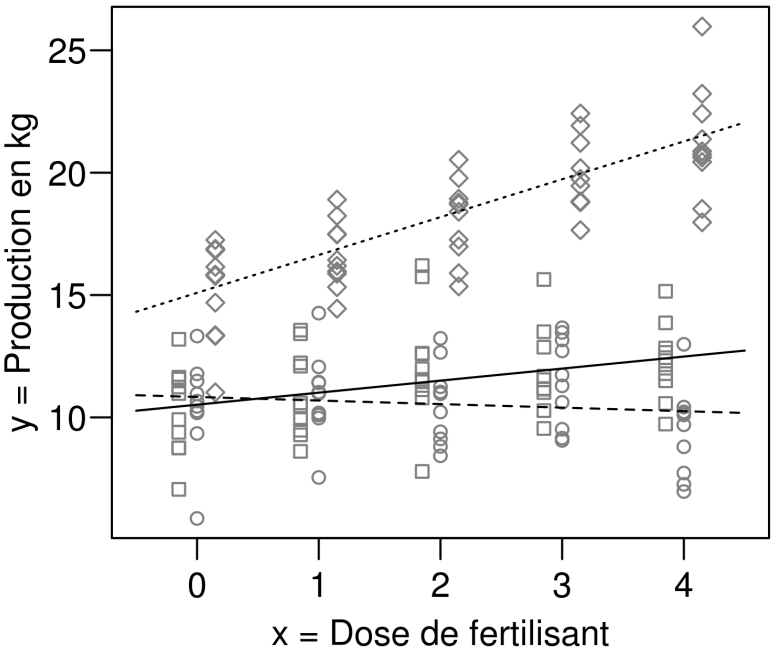
Interprétation

```
> mod <- lm( tomato ~ fertilizer + variety + fertilizer:variety, data=d)
> summary(mod)

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)          10.5183     0.4439  23.695  < 2e-16 ***
fertilizer            0.4915     0.1812   2.712   0.0075 **
variety2              0.3163     0.6278   0.504   0.6151
variety3              4.5727     0.6278   7.284 1.96e-11 ***
fertilizer:variety2  -0.6366     0.2563  -2.484   0.0141 *
fertilizer:variety3   1.0556     0.2563   4.119 6.40e-05 ***
```



Le coefficient "variety2" estime la différence d'intercept entre la variété 2 et la variété 1.
Le coefficient "fertilizer:variety2" estime la différence de pente entre la variété 1 et la variété 2

On estime donc que la variété 2 produit 10.52 + 0.32 kg de tomates pour une dose de fertilisant de 0 (Intercept) et que lorsqu'on augmente la dose de fertilisant d'une unité, la production de cette variété augmente de 0.49 - 0.64 kg (donc diminue de 0.15 kg).



Interactions

Interprétation

```
                  Estimate Std. Error t value Pr(>|t|)
(...)
variety2            0.3163     0.6278   0.504   0.6151
(...)
```

Le coefficient "variety2" est non significativement différent de 0 (p = 0.61).
Ça ne signifie pas qu'il n'y a pas de différence entre la variété 1 et 2 !
Ça signifie qu'il n'y a pas de différence quand la dose de fertilisant est 0 (ou du moins les données ne permettent pas de supporter une telle hypothèse)

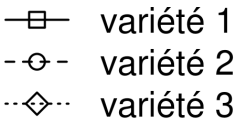
Si on centre variable "fertilizer" sur la valeur "4", l'effet "variety2" devient significatif.
On estime que quand la dose de fertilisant est 4, la variété 2 produit en moyenne 2.23 kg en moins que la variété 1

```
> d$fertilizer_c <- d$fertilizer - 4
> mod <- lm( tomato ~ fertilizer_c + variety + fertilizer_c:variety, data=d)
> summary(mod)

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)           12.4842     0.4439  28.123  < 2e-16 ***
fertilizer_c           0.4915     0.1812   2.712 0.007504 **
variety2              -2.2302     0.6278  -3.552 0.000516 ***
variety3               8.7949     0.6278  14.009  < 2e-16 ***
fertilizer_c:variety2 -0.6366     0.2563  -2.484 0.014140 *
fertilizer_c:variety3  1.0556     0.2563   4.119  6.4e-05 ***
```

Interactions

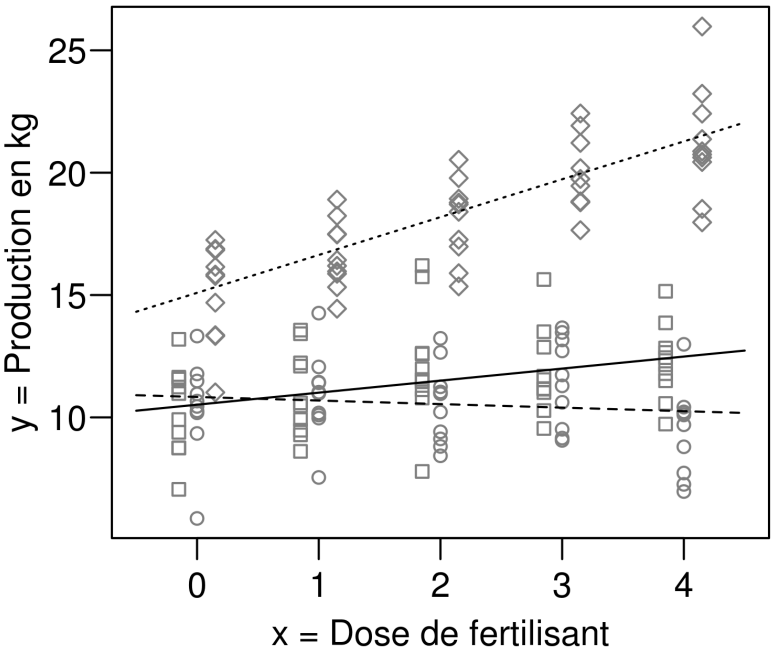
Interprétation

```
> mod <- lm( tomato ~ fertilizer + variety + fertilizer:variety, data=d)
> summary(mod)

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)          10.5183     0.4439  23.695  < 2e-16 ***
fertilizer            0.4915     0.1812   2.712   0.0075 **
variety2              0.3163     0.6278   0.504   0.6151
variety3              4.5727     0.6278   7.284 1.96e-11 ***
fertilizer:variety2  -0.6366     0.2563  -2.484   0.0141 *
fertilizer:variety3   1.0556     0.2563   4.119 6.40e-05 ***
```



Le coefficient "variety3" estime la différence d'intercept entre la variété 3 et la variété 1.
Le coefficient "fertilizer:variety3" estime la différence de pente entre la variété 1 et la variété 3

On estime donc que la variété 3 produit 10.52 + 4.57 kg de tomates pour une dose de fertilisant de 0 (Intercept) et que lorsqu'on augmente la dose de fertilisant d'une unité, la production de cette variété augmente de 0.49 + 1.06 kg (donc augmente de 1.55 kg).



Interactions

Représentation graphique

```
dev.new(10/2.54, 10/2.54)
par(mar = c(3,3,4,1), mgp = c(1.75, 0.6, 0), las = 1)
plot(tomato ~ fertilizer, type = "n", xlim = c(-0.5, 4.5),
     ylab = "y = Production en kg", xlab = "x = Dose de fertilisant")

points(y = d[d$variety==1, "tomato"],
       x = d[d$variety==1,"fertilizer"] - 0.15, pch = 0, cex =0.8, col = "grey50")
points(y = d[d$variety==2, "tomato"],
       x = d[d$variety==2,"fertilizer"], pch = 1, cex =0.8, col = "grey50")
points(y = d[d$variety==3, "tomato"],
       x = d[d$variety==3,"fertilizer"] + 0.15, pch = 5, cex =0.8, col = "grey50")

X1 <- cbind(1, c(-0.5,4.5), 0, 0, 0, 0)
X2 <- cbind(1, c(-0.5,4.5), 1, 0, 1*c(-0.5, 4.5), 0)
X3 <- cbind(1, c(-0.5,4.5), 0, 1, 0, 1*c(-0.5, 4.5))

pred1 <- X1 %*% coef(mod)
pred2 <- X2 %*% coef(mod)
pred3 <- X3 %*% coef(mod)

lines(y = pred1, x = X1[,2], lty = 1)
lines(y = pred2, x = X2[,2], lty = 2)
lines(y = pred3, x = X3[,2], lty = 3)

legend(x = "top", inset = -0.35, xpd = NA, bty = "n", lty = 1:3, pch = c(0,1,5),
       legend = c("variété 1", "variété 2", "variété 3"))
```

```
> X2
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1 -0.5    1    0 -0.5    0
[2,]    1  4.5    1    0  4.5    0
```

Interactions

Comparaisons multiples

On peut tester n'importe quelles hypothèses tout en contrôlant le risque global d'erreur de type I

Exemple : on teste les 4 questions suivantes correspondant chacune à 3 hypothèses :

1) Pour quelle variétés les pentes sont-elles différentes de 0 ?
2) Quelles variétés ont-elles des pentes différentes entre elles ?
3) Quelles variétés ont des productions différentes lorsque la dose de fertilisant = 0
4) Quelles variétés ont des productions différentes lorsque la dose de fertilisant = 4

```
> X <-
+ rbind("pente var1" = c(0,1,0,0,0,0),
+       "pente var2" = c(0,1,0,0,1,0),
+       "pente var3" = c(0,1,0,0,0,1),
+
+       "pente var2 - var1" = c(0,0,0,0,1,0),
+       "pente var3 - var1" = c(0,0,0,0,0,1),
+       "pente var3 - var2" = c(0,0,0,0,-1,1),
+
+       "production var2-var1 @dose=0" = c(1,0,1,0,0,0) - c(1,0,0,0,0,0),
+       "production var3-var1 @dose=0" = c(1,0,0,1,0,0) - c(1,0,0,0,0,0),
+       "production var3-var2 @dose=0" = c(1,0,0,1,0,0) - c(1,0,1,0,0,0),
+
+       "production var2-var1 @dose=4" = c(1,4,1,0,4,0) - c(1,4,0,0,0,0),
+       "production var3-var1 @dose=4" = c(1,4,0,1,0,4) - c(1,4,0,0,0,0),
+       "production var3-var2 @dose=4" = c(1,4,0,1,0,4) - c(1,4,1,0,4,0)
+       )
```

```
                          Estimate
(Intercept)                10.5183
fertilizer                  0.4915
variety2                    0.3163
variety3                    4.5727
fertilizer:variety2        -0.6366
fertilizer:variety3         1.0556
```

Interactions

Comparaisons multiples

On peut tester n'importe quelles hypothèses tout en contrôlant le risque global d'erreur de type I

Matrice de contrastes (non indépendants) résultant :

```
> X
                                [,1] [,2] [,3] [,4] [,5] [,6]
pente var1                         0    1    0    0    0    0
pente var2                         0    1    0    0    1    0
pente var3                         0    1    0    0    0    1
pente var2 - var1                  0    0    0    0    1    0
pente var3 - var1                  0    0    0    0    0    1
pente var3 - var2                  0    0    0    0   -1    1
production var2-var1 @dose=0       0    0    1    0    0    0
production var3-var1 @dose=0       0    0    0    1    0    0
production var3-var2 @dose=0       0    0   -1    1    0    0
production var2-var1 @dose=4       0    0    1    0    4    0
production var3-var1 @dose=4       0    0    0    1    0    4
production var3-var2 @dose=4       0    0   -1    1   -4    4
```

Interactions

Comparaisons multiples

On peut tester n'importe quelles hypothèses tout en contrôlant le risque global d'erreur de type I

Comparaisons multiples avec p-valeur ajustée

NB : si on ne veut pas contrôler pour le risque global d'erreur, on peut prendre chaque coefficient estimé ± 2* Std. Error pour obtenir un intervalle de confiance approximatif à 95 %

```
> library(multcomp)
> modmc <- glht(mod, linfct = X)
> summary(modmc)

Linear Hypotheses:
                                          Estimate Std. Error t value Pr(>|t|)
pente var1 == 0                             0.4915     0.1812   2.712   0.0570 .
pente var2 == 0                            -0.1451     0.1812  -0.801   0.9447
pente var3 == 0                             1.5470     0.1812   8.536   <0.001 ***
pente var2 - var1 == 0                     -0.6366     0.2563  -2.484   0.0993 .
pente var3 - var1 == 0                      1.0556     0.2563   4.119   <0.001 ***
pente var3 - var2 == 0                      1.6922     0.2563   6.603   <0.001 ***
production var2-var1 @dose=0 == 0           0.3163     0.6278   0.504   0.9925
production var3-var1 @dose=0 == 0           4.5727     0.6278   7.284   <0.001 ***
production var3-var2 @dose=0 == 0           4.2564     0.6278   6.780   <0.001 ***
production var2-var1 @dose=4 == 0          -2.2302     0.6278  -3.552   0.0047 **
production var3-var1 @dose=4 == 0           8.7949     0.6278  14.009   <0.001 ***
production var3-var2 @dose=4 == 0          11.0251     0.6278  17.562   <0.001 ***
```

Interactions

Remarques générales sur les interactions

Les interactions sont fréquentes dans la nature mais souvent de faible intensité.
Parfois elles peuvent cependant masquer complètement des effets principaux ("main effects").

Il faut en général beaucoup plus de données pour estimer les interactions que les effets principaux

On peut avoir des interactions de deuxième niveau (A*B*C) ou plus mais on arrive rapidement à des niveaux de complexité difficilement interprétables

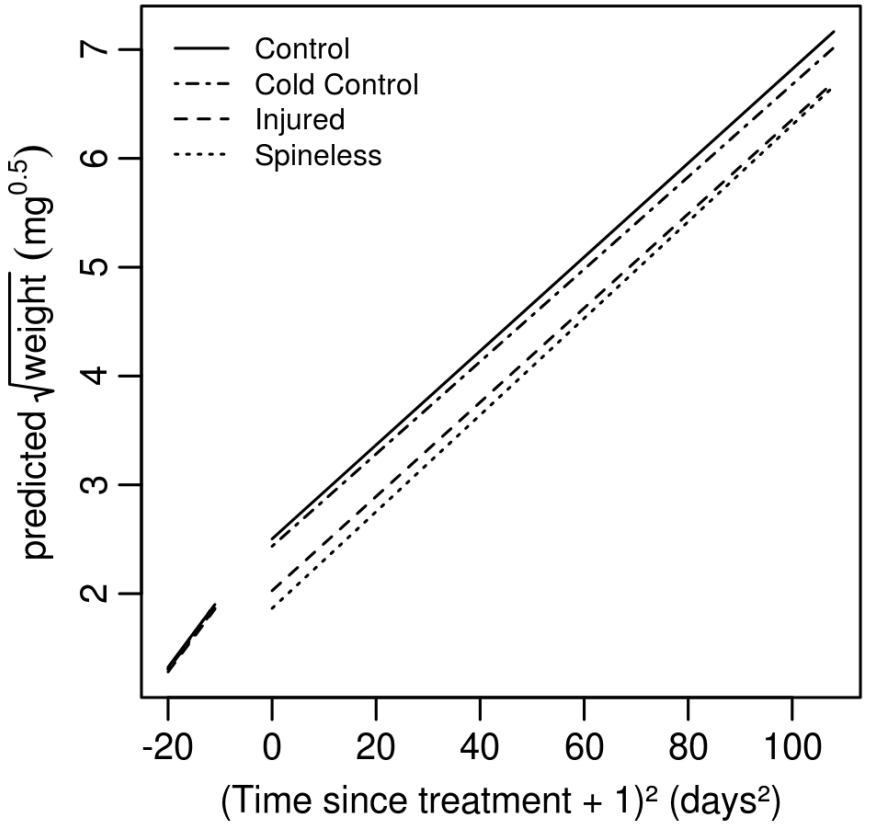
Pour des jeux de données très complexes où on peut difficilement définir a priori toutes les interactions potentielles, les analyses de Data Mining par arbres de régression (Regression Trees, Random Forests, etc...) peuvent être précieuses.

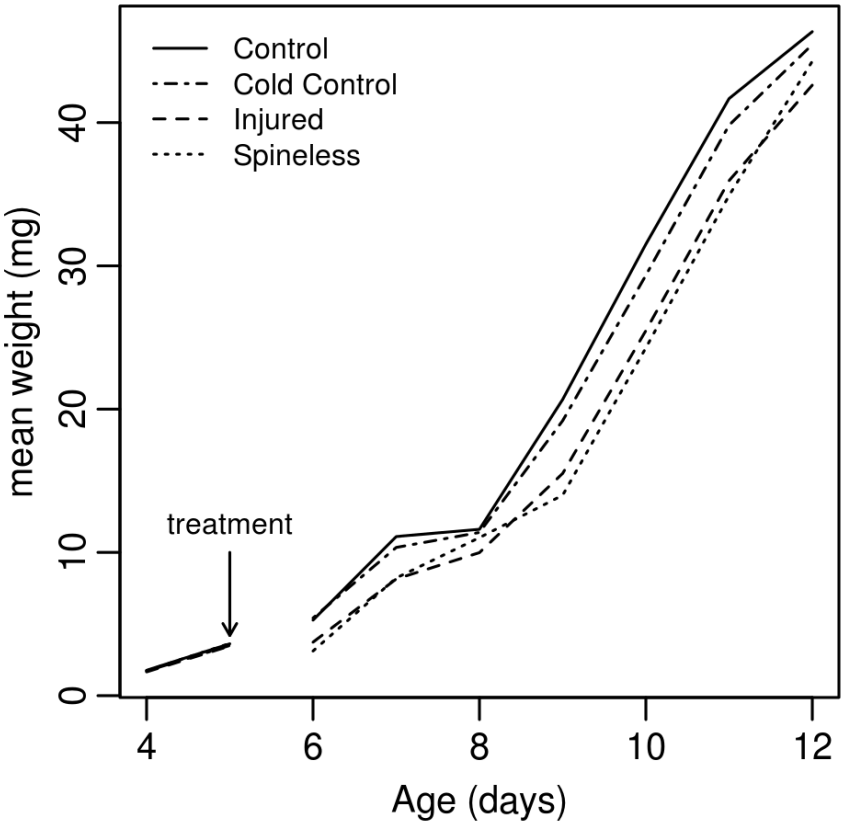
Interactions

Exemple réel d'interaction triple

poids de larves ~ temps * avant/après ablation d'épines * 4 types de traitements :

On veut savoir si il y a une différence de développement avant/après l'ablation et si cette différence dépend du type de traitement --> interaction triple (non significative ici)





Hautier et al. (in prep)

Interactions

Exemple simulé d'interaction triple : BACI design complet

On suit les populations d'une espèce chaque année sur 10 sites.
Après 10 ans on constate que les populations diminuent et on décide de mettre en place des mesures de gestion sur 5 sites.
On continue à suivre les 10 sites pendant 10 ans.

On veut savoir si l'effet de la gestion a été bénéfique par rapport aux sites non gérés.

BACI

BA : Before - After : dans ce cas avant/après la mise en place des mesures de gestion

CI : Control - Impact : sites contrôles sans gestion vs sites "impactés" avec de la gestion après une certaine date.

Interactions

Exemple simulé d'interaction triple : BACI design complet

Simulation des données

```
n <- 5
d <- data.frame(
    year = rep(-9:10, each = n*2),
    BA = factor(rep(c("before", "after"), each = 10*n*2),
                    levels = c("before", "after")),
    CI =rep(c("control", "impact"), times = 10*n*2)
)
X <- model.matrix(~ year * BA * CI, data=d)
B <- c(100, -1, 0, -5, -5, 0, 0, 3.5)

set.seed(1)
d$nb <- X %*% B + rnorm(200, 0, 5)
```

```
> d
    year     BA      CI        nb
1     -9 before control 105.86773
2     -9 before  impact 104.91822
3     -9 before control 104.82186
4     -9 before  impact 111.97640
5     -9 before control 110.64754
6     -9 before  impact  99.89766
7     -9 before control 111.43715
```

NB : La variable year correspond aux nombres d'années depuis le début de la gestion dans les sites "Impact"

Interactions

Exemple simulé d'interaction triple : BACI design complet

```
> mod <- lm( nb ~ year + BA + CI + year:BA + year:CI + BA:CI + year:BA:CI, data = d)
> summary(mod)

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)           100.71122    1.23124  81.796  < 2e-16 ***
year                   -1.05920    0.23063  -4.593 7.91e-06 ***
BAafter                -2.26006    1.88781  -1.197  0.23271
CIimpact               -5.49685    1.74124  -3.157  0.00185 **
year:BAafter           -4.70638    0.32616 -14.429  < 2e-16 ***
year:CIimpact           0.08212    0.32616   0.252  0.80149
BAafter:CIimpact        2.83589    2.66977   1.062  0.28947
year:BAafter:CIimpact   3.01824    0.46127   6.543 5.33e-10 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4.684 on 192 degrees of freedom
Multiple R-squared:  0.9387,  Adjusted R-squared:  0.9364
F-statistic: 419.7 on 7 and 192 DF,  p-value: < 2.2e-16
```

Interactions

Exemple simulé d'interaction triple : BACI design complet

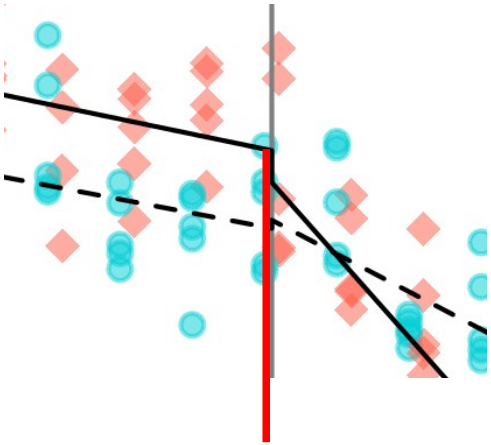
NB :

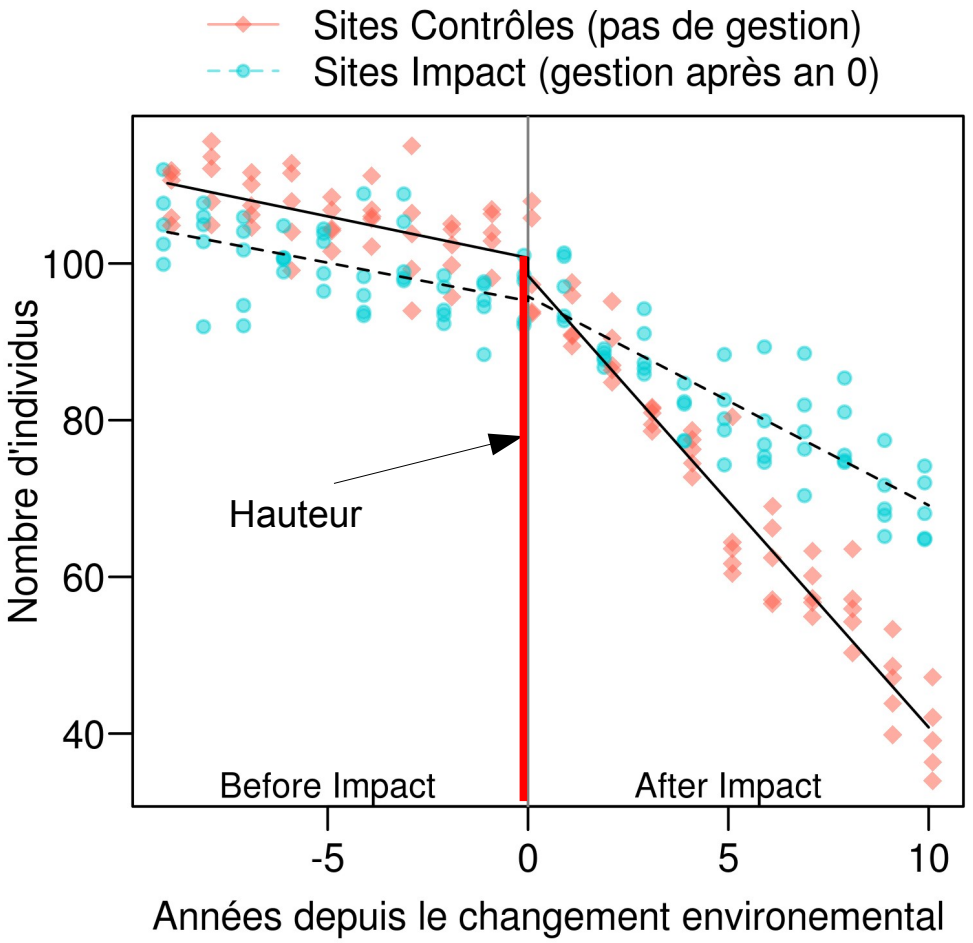
1) Pour ce genre d'études on suit les mêmes sites au cours du temps.
On devrait en tenir compte dans l'analyse
(ajout du site comme variable aléatoire --> voir plus loin, modèles mixtes)

2) Lorsqu'on compte un nombre d'individus, les résidus ne suivent en général pas une distribution normale et n'ont pas une variance homogène
--> il faudrait utiliser une distribution de Poisson --> voir plus loin GLM

Interactions

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	**100.71122**	**1.23124**	**81.796**	**< 2e-16**	*******
year	-1.05920	0.23063	-4.593	7.91e-06	***
BAafter	-2.26006	1.88781	-1.197	0.23271	
CIimpact	-5.49685	1.74124	-3.157	0.00185	**
year:BAafter	-4.70638	0.32616	-14.429	< 2e-16	***
year:CIimpact	0.08212	0.32616	0.252	0.80149	
BAafter:CIimpact	2.83589	2.66977	1.062	0.28947	
year:BAafter:CIimpact	3.01824	0.46127	6.543	5.33e-10	***

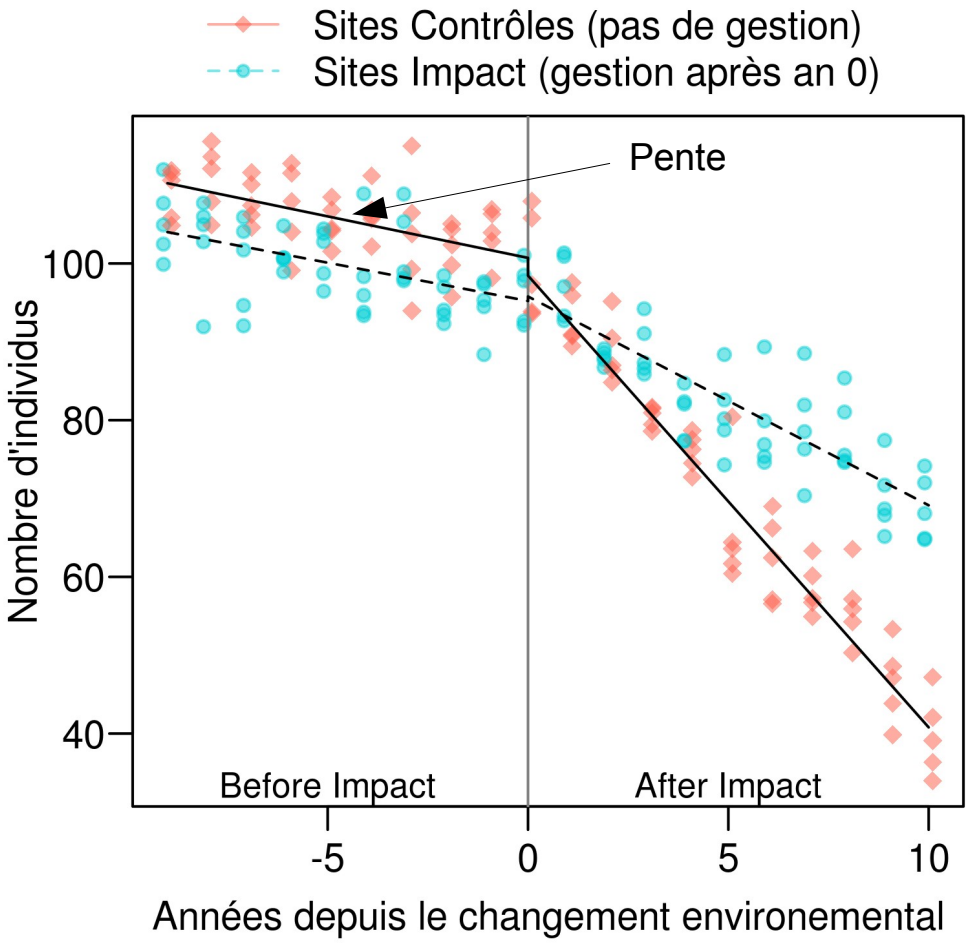




Intercept :
On estime qu'il y a en moyenne 100.71 individus dans les sites contrôles en l'an 0 (et avant la gestion mais çà n'a pas beaucoup de sens ici)

Interactions

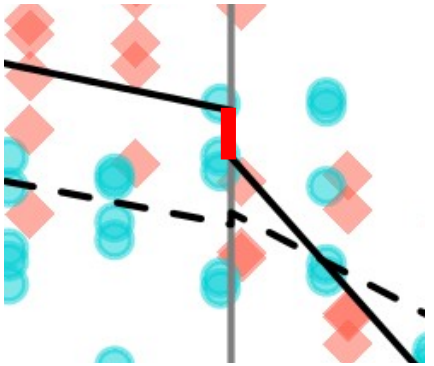
	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	100.71122	1.23124	81.796	< 2e-16	***
year	**-1.05920**	**0.23063**	**-4.593**	**7.91e-06**	*******
BAafter	-2.26006	1.88781	-1.197	0.23271	
CIimpact	-5.49685	1.74124	-3.157	0.00185	**
year:BAafter	-4.70638	0.32616	-14.429	< 2e-16	***
year:CIimpact	0.08212	0.32616	0.252	0.80149	
BAafter:CIimpact	2.83589	2.66977	1.062	0.28947	
year:BAafter:CIimpact	3.01824	0.46127	6.543	5.33e-10	***

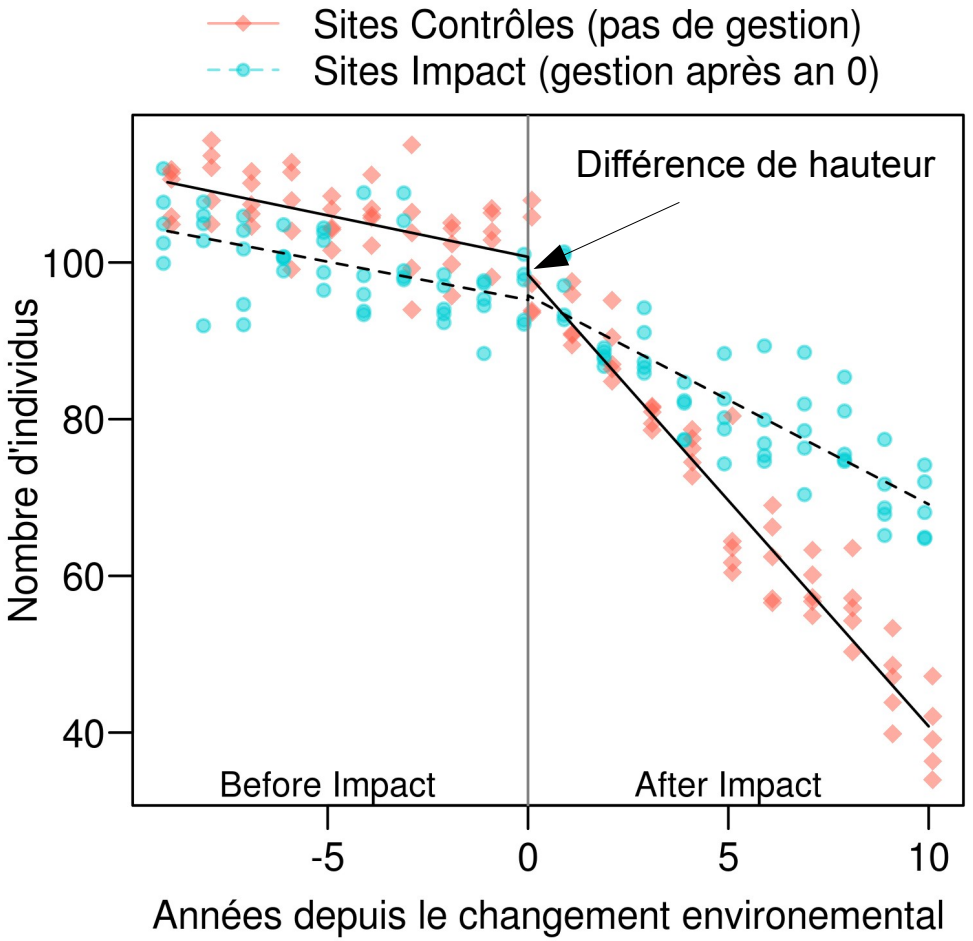


year :
On estime qu'on perdait en moyenne 1.06 individus par an sur les sites contrôles dans la période avant la gestion

Interactions

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	100.71122	1.23124	81.796	< 2e-16	***
year	-1.05920	0.23063	-4.593	7.91e-06	***
BAafter	**-2.26006**	**1.88781**	**-1.197**	**0.23271**	
CIimpact	-5.49685	1.74124	-3.157	0.00185	**
year:BAafter	-4.70638	0.32616	-14.429	< 2e-16	***
year:CIimpact	0.08212	0.32616	0.252	0.80149	
BAafter:CIimpact	2.83589	2.66977	1.062	0.28947	
year:BAafter:CIimpact	3.01824	0.46127	6.543	5.33e-10	***





BAafter :

On estime qu'il y a en moyenne une différence de -2.26 individus en l'an 0, dans les sites contrôles entre avant et après la mise en place de la gestion.

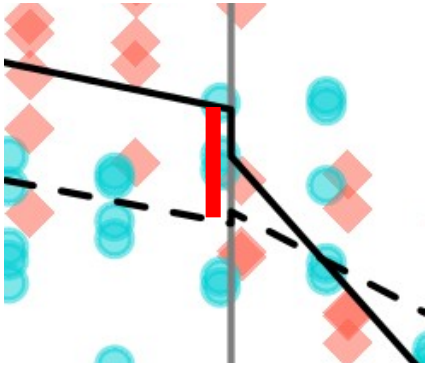
--> ce paramètre n'a pas beaucoup de sens biologique (il n'est d'ailleurs pas significatif).

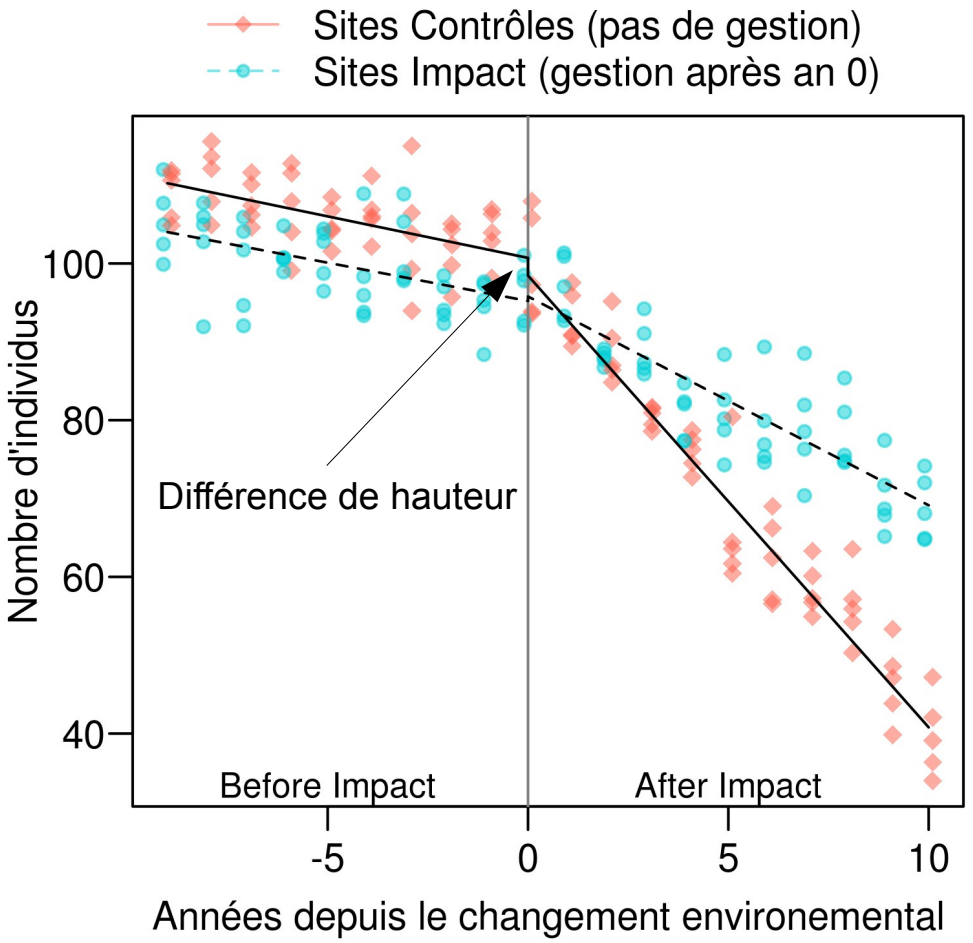
On pourrait donc dans ce cas enlever l'effet principal "BA" du modèle tout en gardant certaines de ses interactions.

On force alors les deux droites avant/après pour le contrôle à passer par le même point en l'an 0.

Interactions

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	100.71122	1.23124	81.796	< 2e-16	***
year	-1.05920	0.23063	-4.593	7.91e-06	***
BAafter	-2.26006	1.88781	-1.197	0.23271	
CIimpact	**-5.49685**	**1.74124**	**-3.157**	**0.00185**	**
year:BAafter	-4.70638	0.32616	-14.429	< 2e-16	***
year:CIimpact	0.08212	0.32616	0.252	0.80149	
BAafter:CIimpact	2.83589	2.66977	1.062	0.28947	
year:BAafter:CIimpact	3.01824	0.46127	6.543	5.33e-10	***



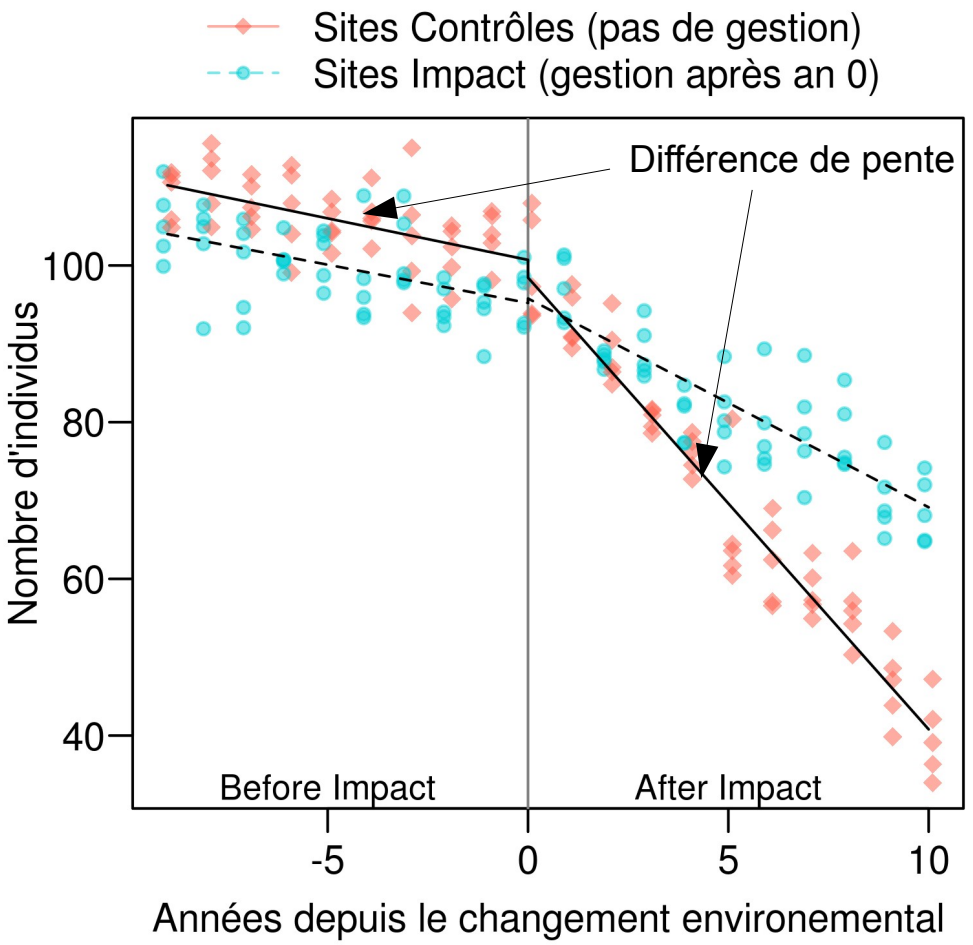


CIimpact:
On estime qu'il y a en moyenne une différence de -5.5 individus en l'an 0, entre les sites contrôle et les sites impact.

Si les sites ont bien été attribués au hasard, ceci ne devrait normalement arriver que par hasard.

Interactions

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	100.71122	1.23124	81.796	< 2e-16	***
year	-1.05920	0.23063	-4.593	7.91e-06	***
BAafter	-2.26006	1.88781	-1.197	0.23271	
CIimpact	-5.49685	1.74124	-3.157	0.00185	**
year:BAafter	**-4.70638**	**0.32616**	**-14.429**	**< 2e-16**	*******
year:CIimpact	0.08212	0.32616	0.252	0.80149	
BAafter:CIimpact	2.83589	2.66977	1.062	0.28947	
year:BAafter:CIimpact	3.01824	0.46127	6.543	5.33e-10	***

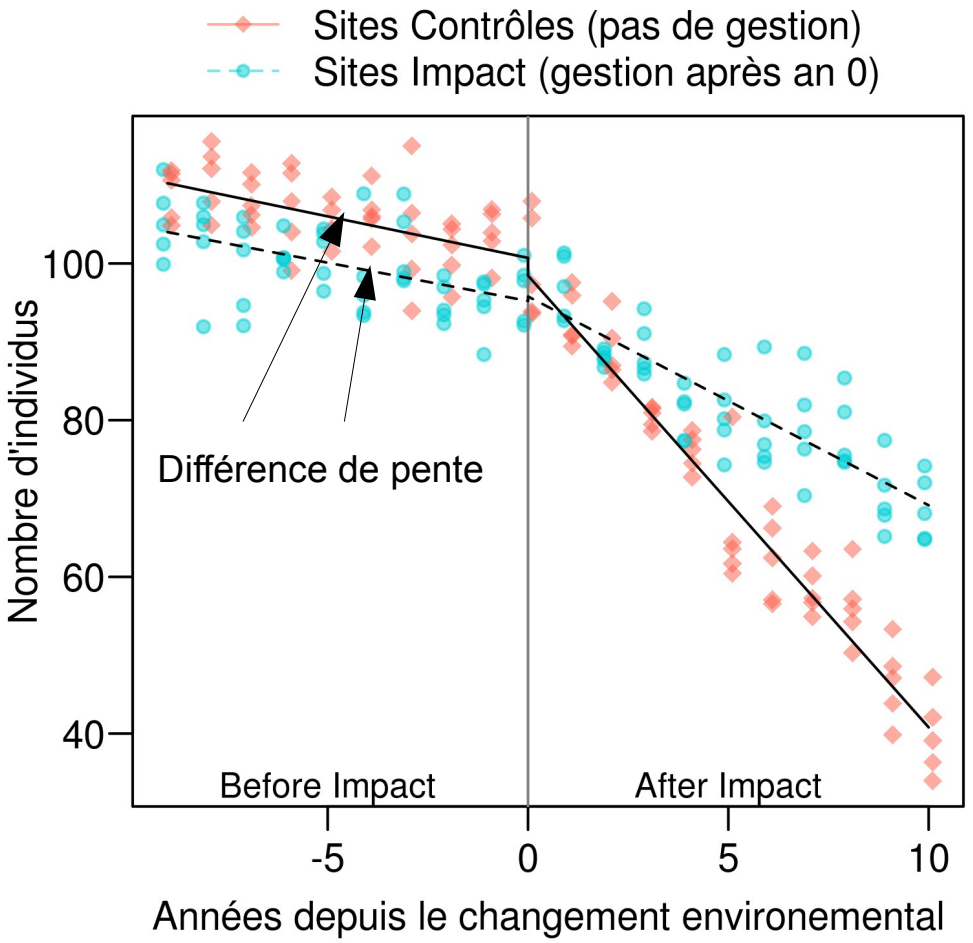


year :BAafter :
Après le début de la période de gestion, les sites contrôles ont perdu en moyenne 4.7 individus par an en plus des 1.05 qu'il perdaient déjà avant cette période. La pente de la droite contrôle est donc de -1.05 avant et de -1.05-4.7 = -5.75 après l'année 0.
year:BAafter représente donc la différence de pente entre avant et après l'année 0 pour les sites contrôle.

Il s'agit ici d'un cas (très) particulier où le début de la période d'impact coïncide avec un événement extérieur à la gesion qui a eu un impact sur l'ensemble des populations

Interactions

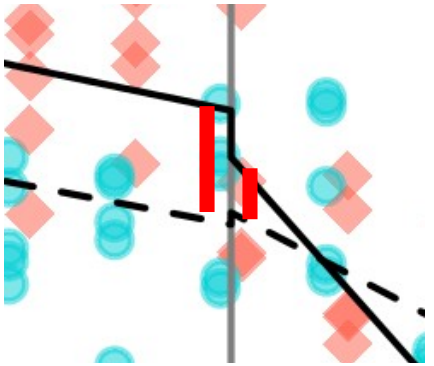
	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	100.71122	1.23124	81.796	< 2e-16	***
year	-1.05920	0.23063	-4.593	7.91e-06	***
BAafter	-2.26006	1.88781	-1.197	0.23271	
CIimpact	-5.49685	1.74124	-3.157	0.00185	**
year:BAafter	-4.70638	0.32616	-14.429	< 2e-16	***
year:CIimpact	**0.08212**	**0.32616**	**0.252**	**0.80149**	
BAafter:CIimpact	2.83589	2.66977	1.062	0.28947	
year:BAafter:CIimpact	3.01824	0.46127	6.543	5.33e-10	***

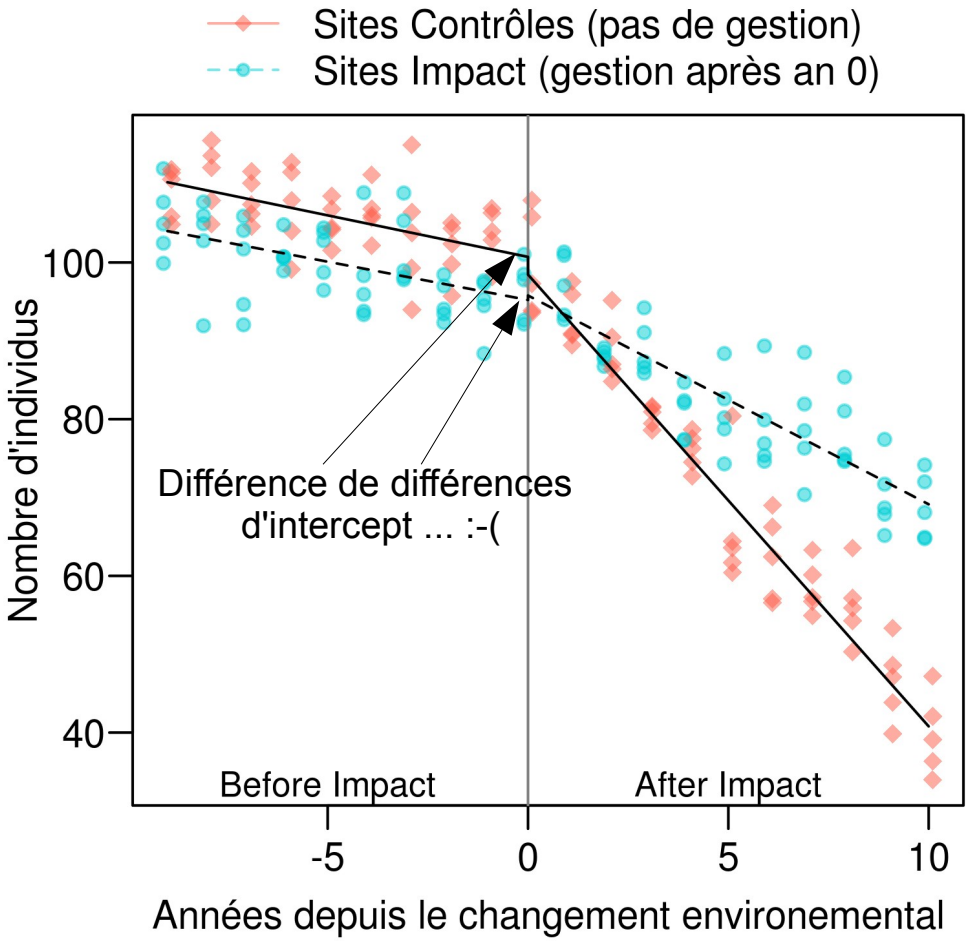


year:CIimpact :
La différence de pente avant l'Année 0 était seulement de 0.08 individus entre les sites contrôles et les sites "impact" qui à cette époque ne recevaient aucune mesure de gestion particulière. La perte annuelle d'individuas avant l'année 0 était donc de -1.05 pour les sites contrôles et de -0.97 pour les futur sites gérés.

Interactions

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	100.71122	1.23124	81.796	< 2e-16	***
year	-1.05920	0.23063	-4.593	7.91e-06	***
BAafter	-2.26006	1.88781	-1.197	0.23271	
CIimpact	-5.49685	1.74124	-3.157	0.00185	**
year:BAafter	-4.70638	0.32616	-14.429	< 2e-16	***
year:CIimpact	0.08212	0.32616	0.252	0.80149	
BAafter:CIimpact	**2.83589**	**2.66977**	**1.062**	**0.28947**	
year:BAafter:CIimpact	3.01824	0.46127	6.543	5.33e-10	***



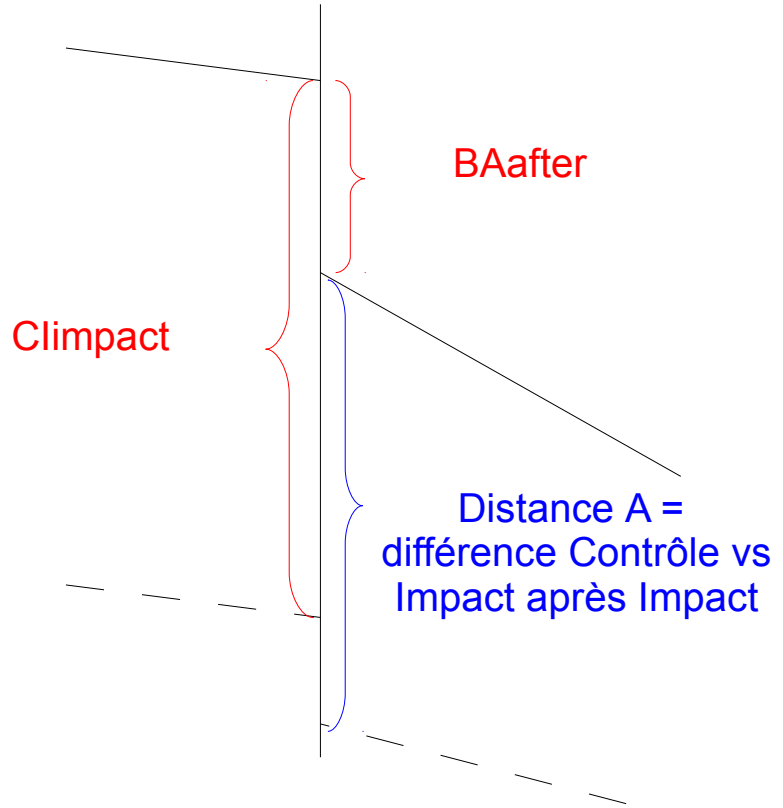


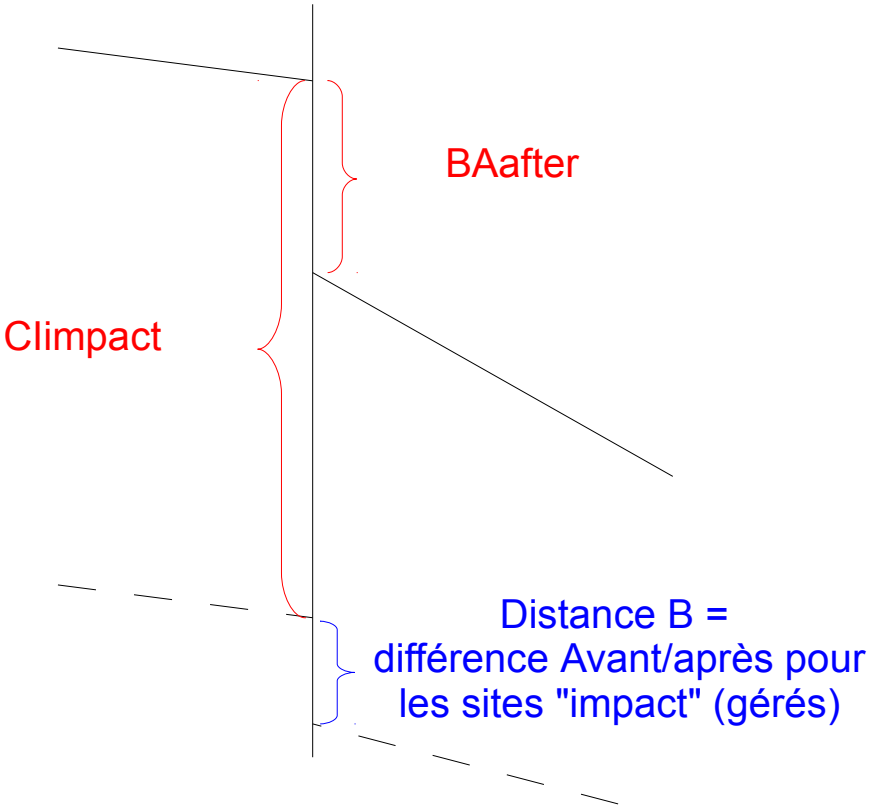
BAafter:CIimpact
Peut-être le plus difficile à cerner et dans ce cas précis sans grande signification biologique...

Ce paramètre permet de tester la question suivante : est-ce que la différence entre les sites contrôle et les sites impact est la même avant et après le début de la gestion ? Ou de manière équivalente, est-ce que les différences avant/après sont les mêmes pour les contrôles et les sites impact ?

Interactions

Deux interprétations équivanlentes de l'interaction BA x CI



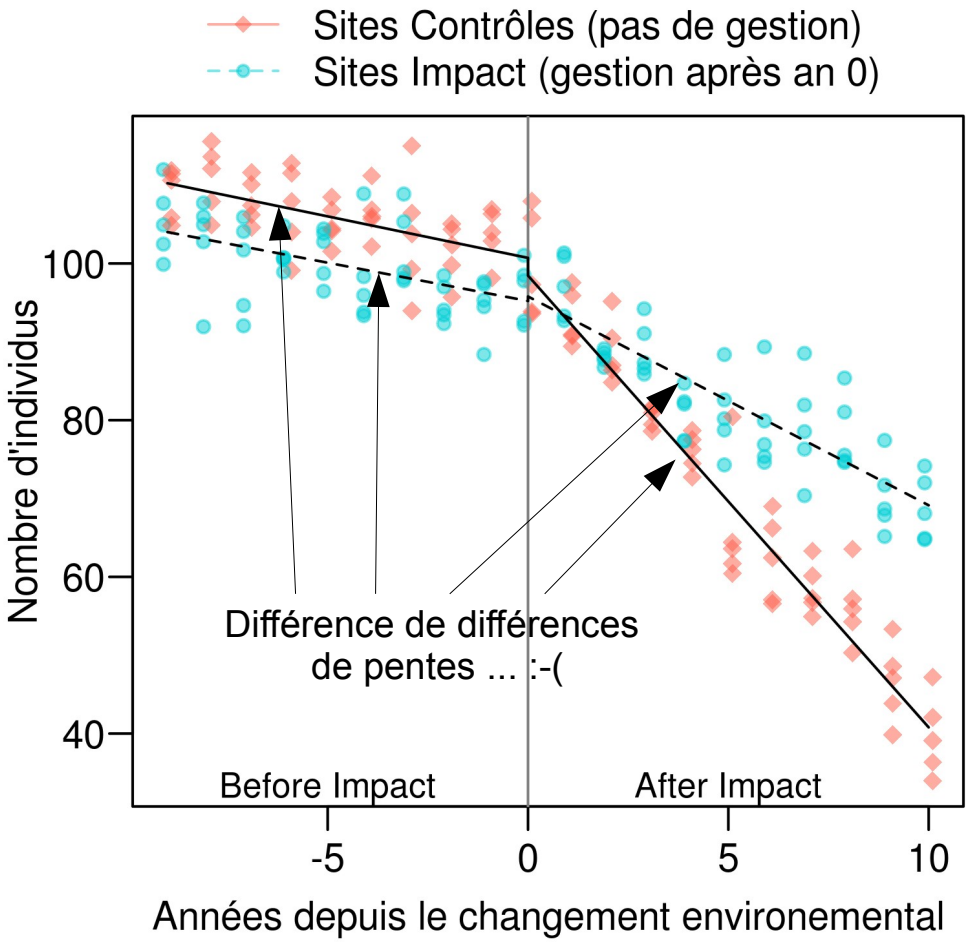


BAafter :CImpact = CImpact - Distance A

BAafter :CImpact = BAafter - Distance B

Interactions

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	100.71122	1.23124	81.796	< 2e-16	***
year	-1.05920	0.23063	-4.593	7.91e-06	***
BAafter	-2.26006	1.88781	-1.197	0.23271	
CIimpact	-5.49685	1.74124	-3.157	0.00185	**
year:BAafter	-4.70638	0.32616	-14.429	< 2e-16	***
year:CIimpact	0.08212	0.32616	0.252	0.80149	
BAafter:CIimpact	2.83589	2.66977	1.062	0.28947	
year:BAafter:CIimpact	**3.01824**	**0.46127**	**6.543**	**5.33e-10**	***



year :BAafter:CIimpact
Cette interaction permet de répondre à la question suivante (qui est la plus importante ici) : est-ce qu'il y a des différences de pentes entre avant et après les gestion, est-ce que ces différences sont du même ordre entre les sites contrôles et les sites gérés (la réponse étant clairement non) ?
Il estime la valeur suivante :
(pente impact après - pente impact avant) - (pente contrôle après - pente contrôle avant).
Pour obtenir la pente du groupe impact, à partir de l'an 0 on doit donc faire :
-1.06 - 4.71 + 0.08 + 3.02 = -2.67

Interactions

Exemple simulé d'interaction triple : BACI design complet

L'interprétation de ces paramètres peut paraître compliquée mais ils testent directement toutes les question d’intérêt pour une telle étude

Interactions

Exemple simulé d'interaction triple : BACI design complet

En changeant la paramétrisation du modèle, on peut cependant obtenir un modèle parfaitement identique mais dont les paramètres estiment directement les 4 intercepts et les 4 pentes plutôt que des différences.
L'interprétation est plus facile mais les tests sont en général sans intérêt.
Mais on peut utiliser les tests post-hoc pour tester n'importe quelle hypothèse sur cette base.

Interactions

Autre paramétrisation pour un modèle identique :

```
> mod2 <- lm( nb ~ -1 + CI:BA+ year:BA:CI, data = d)
> summary(mod2)
                          Estimate Std. Error t value Pr(>|t|)
CIcontrol:BAbefore        100.7112     1.2312  81.796  < 2e-16 ***
CIimpact:BAbefore          95.2144     1.2312  77.332  < 2e-16 ***
CIcontrol:BAafter          98.4512     1.4310  68.797  < 2e-16 ***
CIimpact:BAafter           95.7902     1.4310  66.937  < 2e-16 ***
CIcontrol:BAbefore:year    -1.0592     0.2306  -4.593 7.91e-06 ***
CIimpact:BAbefore:year     -0.9771     0.2306  -4.237 3.52e-05 ***
CIcontrol:BAafter:year     -5.7656     0.2306 -24.999  < 2e-16 ***
CIimpact:BAafter:year      -2.6652     0.2306 -11.556  < 2e-16 ***
```

4 intecepts

4 pentes

```
> anova(mod, mod2)
Analysis of Variance Table

Model 1: nb ~ year * BA * CI
Model 2: nb ~ -1 + CI:BA + year:BA:CI
  Res.Df    RSS Df  Sum of Sq F Pr(>F)
1    192 4212.8
2    192 4212.8  0 1.0914e-11
```

Les 2 modèles sont bien identiques : mêmes résidus, mêmes degrés de liberté

Interactions

On peut recalculer les mêmes différences que dans le premier modèle avec des General Linear Hypothesis (post-hoc tests)

```
> C <- rbind("Pente Before Control" = c(0, 0, 0, 0, 1, 0, 0, 0),
+       "Pente Control After - Pente Control Before " = c(0, 0, 0, 0, -1, 0, 1, 0),
+       "Pente Before Impact - Pente Before Control" = c(0, 0, 0, 0, -1, 1, 0, 0),
+       "Pente(AfterImpact-BeforeImpact)-(AfterControl-BeforeControl)" =
+           c(0, 0, 0, 0, 1, -1, -1, 1))
> C
                                                [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
Pente Before Control                               0    0    0    0    1    0    0    0
Pente Control After - Pente Control Before         0    0    0    0   -1    0    1    0
Pente Before Impact - Pente Before Control         0    0    0    0   -1    1    0    0
Pente(AI-BI)-(AC-BC)                               0    0    0    0    1   -1   -1    1
```

Interactions

On peut recalculer les mêmes différences que dans le premier modèle avec des General Linear Hypothesis (post-hoc tests)

```
> library(multcomp)
> mod2mc <- glht(mod2, linfct = C)
> summary(mod2mc, test =  adjusted("none"))

Linear Hypotheses:
                                                      Estimate Std. Error t value Pr(>|t|)
Pente Before Control == 0                             -1.05920    0.23063  -4.593 7.91e-06 ***
Pente Control After - Pente Control Before  == 0      -4.70638    0.32616 -14.429  < 2e-16 ***
Pente Before Impact - Pente Before Control == 0        0.08212    0.32616   0.252    0.801
Pente(AI-BI)-(AC-BC) == 0                              3.01824    0.46127   6.543 5.33e-10 ***
```

p valeurs non ajustées pour la comparaison

```
> summary(mod)
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)        100.71122    1.23124  81.796  < 2e-16 ***
year                -1.05920    0.23063  -4.593 7.91e-06 ***
BAafter             -2.26006    1.88781  -1.197  0.23271
CIimpact            -5.49685    1.74124  -3.157  0.00185 **
year:BAafter        -4.70638    0.32616 -14.429  < 2e-16 ***
year:CIimpact        0.08212    0.32616   0.252  0.80149
BAafter:CIimpact     2.83589    2.66977   1.062  0.28947
year:BAafter:CIimpact 3.01824   0.46127   6.543 5.33e-10 ***
```

Interactions

Exemple simulé d'interaction triple : BACI design complet

Représentation graphique

```
X1 <- cbind(1,
            c(-9:0, 0:10),
            rep(c(0,1), times = c(10,11)),
            0,
            c(-9:0, 0:10) * rep(c(0,1), times = c(10,11)),
            0,
            0,
            0)

X2 <- cbind(1,
            c(-9:0, 0:10),
            rep(c(0,1), times = c(10,11)),
            1,
            c(-9:0, 0:10) * rep(c(0,1), times = c(10,11)),
            c(-9:0, 0:10) * 1,
            rep(c(0,1), times = c(10,11)) * 1,
            c(-9:0, 0:10) * rep(c(0,1), times = c(10,11)) *1)


pred1 <- X1 %*% coef(mod)
pred2 <- X2 %*% coef(mod)
```

Interactions

Exemple simulé d'interaction triple : BACI design complet

Représentation graphique

```
dev.new(10/2.54, 10/2.54)
par(mar = c(3,3,4,1), mgp = c(1.85, 0.6, 0), las = 1, cex = 0.9)

plot(y = d$nb, x = (d$year - (as.numeric(d$CI)-1.5)*0.2),
     pch = c(18, 20)[as.numeric(d$CI)],
     col = c("#FF5D4980", "#00CFD680")[as.numeric(d$CI)],
     cex = 1.1, xlab = "Années depuis le changement environnemental",
     ylab = "Nombre d'individus")
abline(v=0, col = "grey50")
mtext(c("Before Impact", "After Impact"), 1, -1, at = c(-5,5) , cex = 0.8)


lines(y = pred1, x = X1[,2], lty = 1)
lines(y = pred2, x = X2[,2], lty = 2)

legend(x = "top", inset = -0.20, xpd = NA, bty = "n", lty = 1:2, pch = c(18,20),
       cex = 1, pt.cex = 1, col = c("#FF5D4980", "#00CFD680"),
       legend = c("Sites Contrôles (pas de gestion)", "Sites Impact (gestion après an 0)"))
```

Formules de modèles : syntaxe

Il existe de nombreux raccourcis permettant d'éviter d'écrire un modèle en développant tous ses termes.

Les opérateurs suivants ont tous une signification particulière quand ils sont utilisés dans une formule de modèle :

```
~ 1 + : - * ^ . I() / %in%
```

Le chiffre 1 correspond à l'intercept. Il est présent par défaut dans tous les modèles et ne doit pas être explicitement indiqué.
Les deux modèles suivants sont donc équivalents :

```
y ~ 1 + A + B
y ~ A + B
```

Le signe + sépare les effets additifs
et le signe : indique les interactions entre les termes

```
y ~ A + B + A:B
```

Formules de modèles : syntaxe

L'opérateur * indique tous les effets principaux et toutes les interactions possibles (de tous niveaux) entre les termes qu'il sépare.

Par exemple le modèle suivant :

```
y ~ A * B * C * D
```

est équivalent à :

```
y ~ A + B + C + D + A:B + A:C + A:D + B:C + B:D + C:D
    + A:B:C + A:B:D + B:C:D + A:B:C:D
```

Et le modèle suivant :

```
y ~ A + B * C * D
```

est équivalent à :

```
y ~ A + B + C + D + B:C + B:D + C:D + B:C:D
```

Attention, dans de nombreux autres programmes/langages statistiques, l'opérateur `*` décrit les interactions (`:` dans R)

Formules de modèles : syntaxe

L'opérateur - permet d'enlever un effet.

Par exemple le modèle suivant :

```
y ~ A * B * C * D - B:C:D - A:B:C:D
```

est équivalent à :

```
y ~ 1 + A + B + C + D + A:B + A:C + A:D + B:C + B:D +
    C:D  + A:B:C + A:B:D
```

Et les modèles suivants (équivalents) sont des modèles sans intercept :

```
y ~ -1 + A + B
y ~ A + B -1
```

Formules de modèles : syntaxe

L'opérateur `()^n` estime toutes les interaction au n-1 ème niveau pour les termes qui se trouvent entre les parenthèses.

Par exemple le modèle suivant estime les effets principaux et toutes les interactions doubles (interactions de premier niveau):

```
y ~ (A + B + C + D)^2
```

est équivalent à :

```
y ~ A + B + C + D + A:B + A:C + A:D + B:C + B:D + C:D
```

Et le modèle suivant :

```
y ~ (A + B + C + D)^3
```

est équivalent à :

```
y ~ A + B + C + D + A:B + A:C + A:D + B:C + B:D + C:D
    + A:B:C + A:B:D + B:C:D
```

ou encore :

```
y ~ A * B * C * D - A:B:C:D
```

Formules de modèles : syntaxe

L'opérateur `.` identifie "toutes les autres variables"

Par exemple le modèle suivant on estime y en fonction de la racine carrée de A et toutes les autres variables présentes dans le jeu de données d (sans interaction)

```
lm(y ~ sqrt(A) + ., data=d)
```

équivalent à :

```
y ~ sqrt(A) + B + C + D
```

Et le modèle suivant :

```
lm(y ~ sqrt(A) + (.)^2, data=d)
```

est équivalent à :

```
y ~ sqrt(A) + (B + C + D)^2
y ~ sqrt(A) + B + C + D + B:C + B:D + C:D
```

Formules de modèles : syntaxe

La fonction `update()` permet de modifier un modèle existant et fait intensivement appel aux opérateurs . , + et -

Par exemple dans le cas suivant, on estime d'abord un modèle avec toutes les interactions. puis on le modifie en enlevant l'interaction quadruple `A:B:C:D`

```
mod <- lm(y ~ A * B * C * D, data=d)
update(mod, ~. - A:B:C:D)
```

Dans le cas suivant, on ajoute toutes les interaction doubles :

```
mod <- lm(y ~ A + B + C + D, data=d)
update(mod, ~. + (.)^2)
```

Formules de modèles : syntaxe

Facteurs croisés vs facteurs hiérarchisés

Les opérateurs `/` et `%in%` permettent de spécifier des facteurs hiérarchisés ("nested factors")

Facteurs croisés

Exemple : dans 3 sites différents, on a appliqué les pesticides A, B et C.
Le traitement A du site 1 est bien comparable au traitement A du site 2.
On dit que les facteurs site et pesticide sont croisés et on les spécifiera "normalement"

```
y ~ site + pesticide + site:pesticide
```

Formules de modèles : syntaxe

Facteurs hiérarchisés :

Exemple : dans 3 sites différents on a positionné 3 quadrats que l'on appelle A, B, C.

Le quadrat A du site 1 n'a pas grand chose à voir avec le quadrat A du site 2. On pourrait très bien renommer le quadrat A du site 2 quadrat "B", çà ne changerait rien.

Les niveaux du facteur quadrat ne doivent être considérés que à l'intérieur de chaque niveau du facteur site. On dit que le facteur quadrat est hiérarchisé au facteur site.

Ces facteurs hiérarchisés doivent alors être spécifiés d'une manière particulière

(NB : en général les facteurs hiérarchisés sont considérés comme des facteurs aléatoires et sont en général spécifiés dans des modèles mixtes)

```
lm(y ~ site + quadrat %in% site)
```

ou (équivalent)

```
lm(y ~ site/quadrat)
```

Formules de modèles : syntaxe

L'opérateur `I()` permet d'utiliser les autres opérateurs de manière classique au sein d'une formule.

Par exemple, on veut modéliser y en fonction de la somme de A et B, du rapport entre A et C et du carré de D :

```
y ~ I(A + B) + I(A/C) + I(D^2)
```

Dans certains cas, il est plutôt conseillé de créer de nouvelles variables avec un nouveau nom et de n'utiliser `I()` que pour les analyses exploratoires:

```
d$AplusB <- d$A + d$B
d$ACratio <- d$A / d$C
d$Dsq <- d$D^2
y ~ AplusB + ACratio + Dsq
```

Comparaison de modèles emboîtés

Pour tester globalement des variables qualitatives à plus de 2 niveaux il faut utiliser des comparaisons de modèles emboîtés.

Typiquement, les logiciels de statistiques vous donnent un "tableau d'analyse de la variance" en vous permettant de choisir entre plusieurs "Sum of Square" : type I, type II, type III

Ces 3 approches sont toutes correctes mais testent des hypothèses parfois totalement différentes (pour une même variable explicative).

Il est parfois assez difficile de se savoir quelles hypothèses sont testées exactement en particulier en présence d'interactions et il faut être particulièrement prudent avec les interprétations de tels tableaux.

Comparaison de modèles emboîtés

```
> mod <- lm( tomato ~ fertilizer + variety + fertilizer:variety, data=d)
> summary(mod)
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)          10.5183     0.4439  23.695  < 2e-16 ***
fertilizer            0.4915     0.1812   2.712   0.0075 **
variety2              0.3163     0.6278   0.504   0.6151
variety3              4.5727     0.6278   7.284 1.96e-11 ***
fertilizer:variety2  -0.6366     0.2563  -2.484   0.0141 *
fertilizer:variety3   1.0556     0.2563   4.119 6.40e-05 ***
---
```

Si on utilise drop1 comme on l'a vu précédemment, il ne teste que l'interaction du niveau le plus élevé :

```
> drop1(mod, test = "F")
                   Df Sum of Sq    RSS    AIC F value    Pr(>F)
<none>                          472.94 184.25
fertilizer:variety  2     146.1 619.03 220.63  22.242 3.822e-09 ***
```

Comparaison de modèles emboîtés

Pour obtenir les SS de type I :

```
> anova(mod)
                     Df  Sum Sq Mean Sq F value    Pr(>F)
fertilizer            1  119.49  119.49  36.383 1.302e-08 ***
variety               2 1732.82  866.41 263.805 < 2.2e-16 ***
fertilizer:variety    2  146.10   73.05  22.242 3.822e-09 ***
Residuals           144  472.94    3.28
-
```

Pour obtenir les SS de type II :

```
> library(car)
> Anova(mod)
                     Sum Sq  Df F value    Pr(>F)
fertilizer           119.49   1  36.383 1.302e-08 ***
variety             1732.82   2 263.805 < 2.2e-16 ***
fertilizer:variety   146.10   2  22.242 3.822e-09 ***
Residuals            472.94 144
```

NB : dans ce cas, Type I et Type II donnent les mêmes résultats car on a un design parfaitement balancé et des variables explicatives parfaitement indépendantes

Comparaison de modèles emboîtés

Pour obtenir les SS de type III :

```
> library(car)
> Anova(mod, type = "III")
Anova Table (Type III tests)

Response: tomato
                     Sum Sq  Df  F value    Pr(>F)
(Intercept)        1843.91   1 561.4360 < 2.2e-16 ***
fertilizer           24.15   1   7.3545  0.007504 **
variety             217.37   2  33.0919 1.496e-12 ***
fertilizer:variety  146.10   2  22.2421 3.822e-09 ***
Residuals           472.94 144
```

Ou alternativement :

```
> drop1(mod, .~., test = "F")

                   Df Sum of Sq    RSS    AIC F value    Pr(>F)
<none>                          472.94 184.25
fertilizer          1    24.154 497.09 189.72  7.3545  0.007504 **
variety             2   217.366 690.30 236.97 33.0919 1.496e-12 ***
fertilizer:variety  2   146.098 619.03 220.63 22.2421 3.822e-09 ***
```

NB : Type II et Type III donnent les mêmes résultats quand il n'y a pas d'interaction (ici les résultats sont donc différents).
L'interaction du plus haut niveau donne toujours les mêmes résultats avec les 3 méthodes

Comparaison de modèles emboîtés

Il est de manière générale déconseillé d'utiliser ces tableaux à moins de comprendre quelle hypothèse est évaluée par chaque test.

En général, si l'interaction AxB est significative, il n'y a que peu d'intérêt à tester si A ou B sont significatifs. Le fait que A soit significatif va dépendre des valeurs de B et/ou de la position de l'intercept.
Si AxB est significatif, A et B sont importants pour prédire y

En général, il vaut mieux respecter la règle de marginalité :
dans les comparaisons de modèles, si on teste un effet principal (main effect) alors il faut ignorer également les interactions qui comprennent cet effet.

Les tests de type II respectent normalement cette règle de marginalité alors que les tests de type III ne le font pas.
--> de manière générale il faut éviter les tests de Type III
(qui sont souvent les tests par défaut dans d'autres logiciels)

Mais il existe des cas où on peut à bon escient ne pas respecter cette règle.

Comparaison de modèles emboîtés

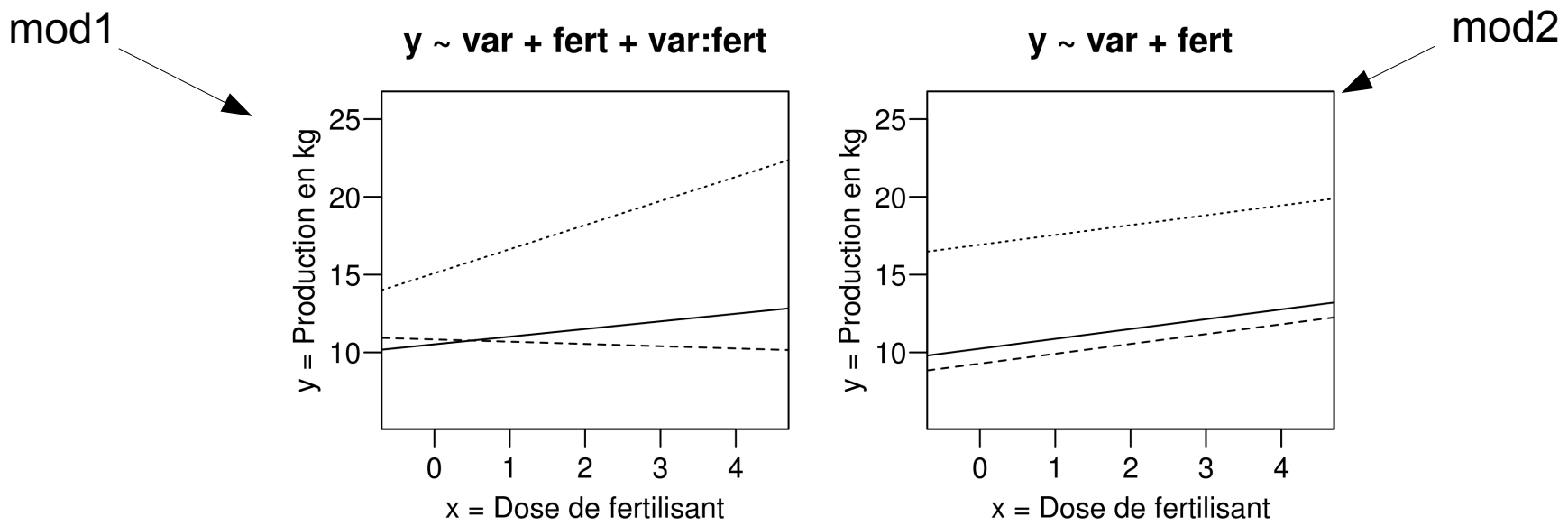
Dans R la pratique recommandée en général et qui permet de maîtriser les hypothèses que l'on teste consiste - quand il y a des interactions - à construire soi-même les modèles correspondant à chaque hypothèse et à les comparer avec la fonction `anova`

Quand il n'y a pas d'interactions, on peut sans souci utiliser `drop1` ou `Anova` (résultats identiques)

Comparaison de modèles emboîtés

Test de l'interaction (mêmes résultat pour toutes les méthodes) :

```
> mod1 <- lm( tomato ~ fertilizer + variety + fertilizer:variety, data=d)
> mod2 <- lm( tomato ~ fertilizer + variety , data=d)
> anova(mod1,mod2)
Analysis of Variance Table

Model 1: tomato ~ fertilizer + variety + fertilizer:variety
Model 2: tomato ~ fertilizer + variety
  Res.Df    RSS Df Sum of Sq      F    Pr(>F)
1    144 472.94
2    146 619.03 -2    -146.1 22.242 3.822e-09 ***
```



Comparaison de modèles emboîtés

Test de la variété. On peut tester l'effet variété de au moins deux manières différentes :
Test de Type II (respectant la règle de marginalité)

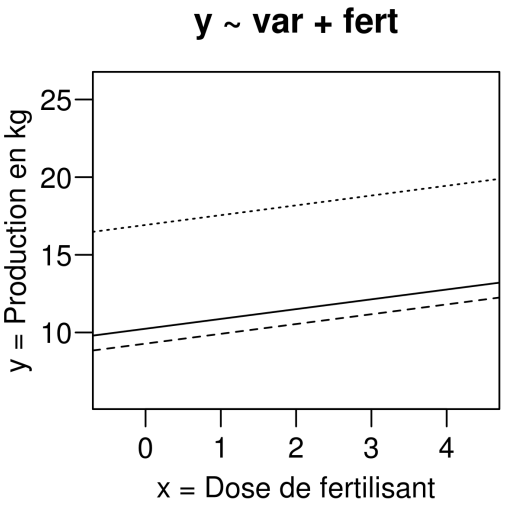
```
> mod1 <- lm(tomato ~ fertilizer + variety , data=d)
> mod2 <- lm(tomato ~ fertilizer , data=d)
> anova(mod1,mod2)

  Res.Df     RSS Df Sum of Sq      F    Pr(>F)
1    146  619.03
2    148 2351.85 -2   -1732.8 204.34 < 2.2e-16 ***
```

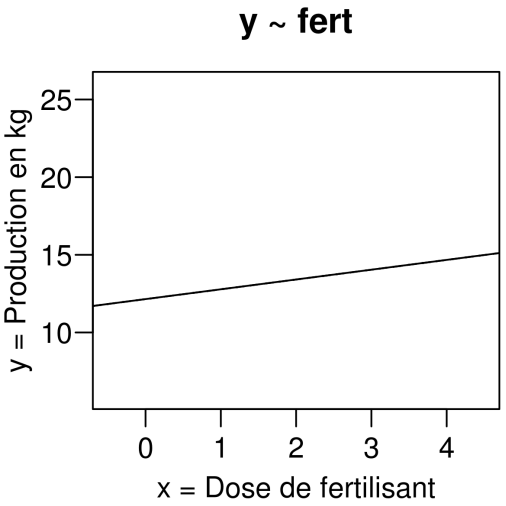
Test de Type III (ne respectant pas la règle de marginalité) :

```
> mod1 <- lm( tomato ~ fertilizer + variety + fertilizer:variety, data=d)
> mod2 <- lm(tomato ~ fertilizer + fertilizer:variety, data=d)
> anova(mod1,mod2)

  Res.Df    RSS Df Sum of Sq      F    Pr(>F)
1    144 472.94
2    146 690.30 -2   -217.37 33.092 1.496e-12 ***
```

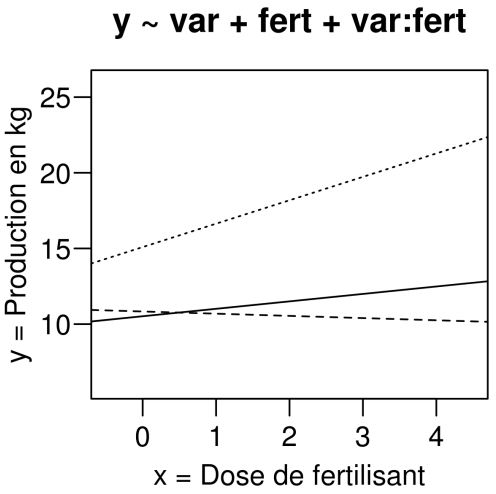
Comparaison de modèles emboîtés

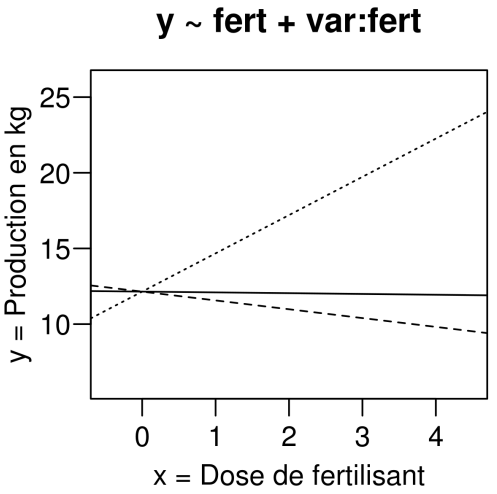
Ces deux tests ne testent pas la même hypothèse :





Est-ce qu'il y a une différence entre les variétés quelque soit la dose ?
-> la réponse ne change pas si on modifie l'intercept (pex si on centre les x)





Est-ce qu'il y a une différence entre les variétés à l'intercept, càd dans ce cas précis quand la dose de fertilisant est 0 ?

Comparaison de modèles emboîtés

Test de l'effet "fertilisant". On peut tester l'effet "fertilisant" de plusieurs manières également. Pex 2 approches différentes qui respectent toutes les deux la règle de marginalité :

```
> mod1 <- lm(tomato ~ fertilizer + variety , data=d)
> mod2 <- lm(tomato ~ variety , data=d)
> anova(mod1,mod2)

  Res.Df    RSS Df Sum of Sq      F    Pr(>F)
1    146 619.03
2    147 738.53 -1   -119.49 28.183 4.026e-07 ***
>
```

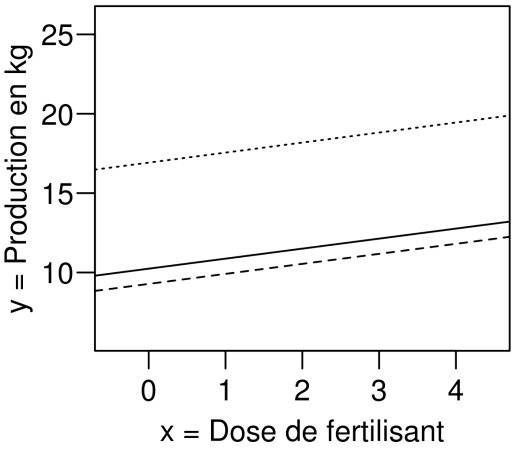
```
> mod1 <- lm(tomato ~ fertilizer + variety + fertilizer:variety , data=d)
> mod2 <- lm(tomato ~  variety  , data=d)
> anova(mod1,mod2)

  Res.Df    RSS Df Sum of Sq      F    Pr(>F)
1    144 472.94
2    147 738.53 -3   -265.59 26.956 6.759e-14 ***
```

NB : encore une fois, tout ceci n'a pas beaucoup de sens ici puisqu'on a une interaction très significative

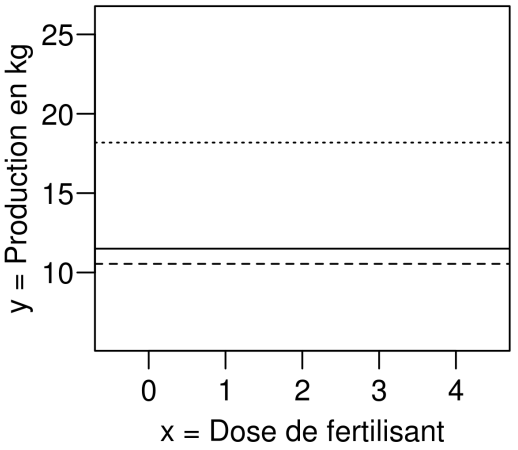
Comparaison de modèles emboîtés

Ces deux tests ne testent pas la même hypothèse mais c'est une hypothèse proche quand l'interaction est faible :

y ~ var + fert

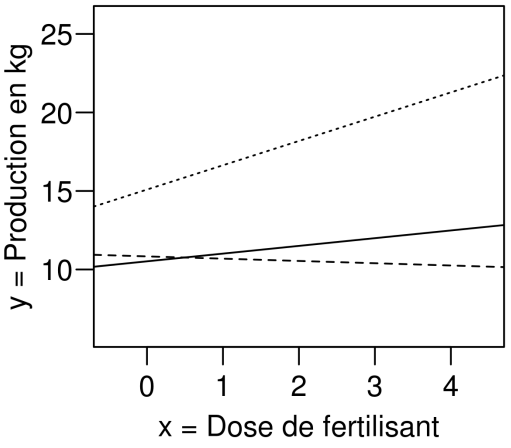


y ~ var

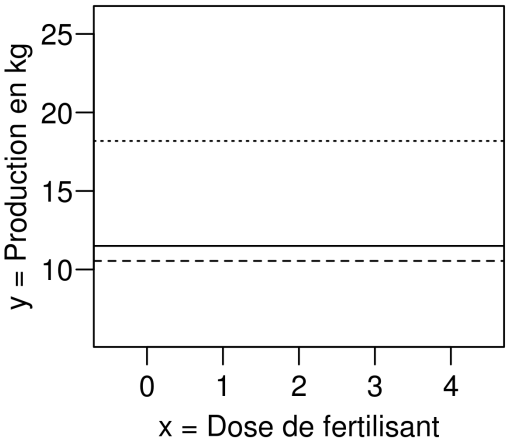


Est-ce que la pente est nulle pour l'ensemble des variétés par rapport à un modèle avec une pente commune pour toutes les variétés ?

y ~ var + fert + var:fert



y ~ var



Est-ce que la pente est nulle pour chaque variété par rapport à un modèle avec une pente différente?

Comparaison de modèles emboîtés

Test de l'effet "fertilisant" de type III : il teste en fait si la pente de la variété de référence est nulle tout en permettant aux autres variétés d'avoir des pentes différentes (c'est donc le même test que dans le summary du modèle).

```
> mod <- lm( tomato ~ fertilizer + variety + fertilizer:variety, data=d)
> summary(mod)
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)          10.5183     0.4439  23.695  < 2e-16 ***
fertilizer            0.4915     0.1812   2.712   0.0075 **
variety2              0.3163     0.6278   0.504   0.6151
variety3              4.5727     0.6278   7.284 1.96e-11 ***
fertilizer:variety2  -0.6366     0.2563  -2.484   0.0141 *
fertilizer:variety3   1.0556     0.2563   4.119 6.40e-05 ***

> Anova(mod, type = "III")
                    Sum Sq  Df  F value    Pr(>F)
(Intercept)        1843.91   1 561.4360 < 2.2e-16 ***
fertilizer           24.15   1   7.3545  0.007504 **
variety             217.37   2  33.0919 1.496e-12 ***
fertilizer:variety  146.10   2  22.2421 3.822e-09 ***
Residuals           472.94 144

> 2.712^2
[1] 7.354944
```

Comparaison de modèles emboîtés

Le résultat va du test de type III pour le fertilisant va donc dépendre du niveau de référence de la variété ce qui n'est pas le cas avec les tests de type II !

```
> d2 <- d
> d2$variety <- relevel(d2$variety, ref = "2")
> mod <- lm( tomato ~ fertilizer + variety + fertilizer:variety, data=d)
> mod2 <- lm( tomato ~ fertilizer + variety + fertilizer:variety, data=d2)
>
> Anova(mod, type = "III")
                    Sum Sq  Df  F value    Pr(>F)
(Intercept)        1843.91   1 561.4360 < 2.2e-16 ***
fertilizer           24.15   1   7.3545  0.007504 **
variety             217.37   2  33.0919 1.496e-12 ***
fertilizer:variety  146.10   2  22.2421 3.822e-09 ***
Residuals           472.94 144

> Anova(mod2, type = "III")
                    Sum Sq  Df  F value    Pr(>F)
(Intercept)        1956.48   1 595.7105 < 2.2e-16 ***
fertilizer            2.11   1   0.6415    0.4245
variety             217.37   2  33.0919 1.496e-12 ***
fertilizer:variety  146.10   2  22.2421 3.822e-09 ***
Residuals           472.94 144
```

Comparaison de modèles emboîtés

Le résultat va du test de type III pour le fertilisant va donc dépendre du niveau de référence de la variété ce qui n'est pas le cas avec les tests de type II !

```
> Anova(mod)

                    Sum Sq  Df F value    Pr(>F)
fertilizer          119.49   1  36.383 1.302e-08 ***
variety            1732.82   2 263.805 < 2.2e-16 ***
fertilizer:variety  146.10   2  22.242 3.822e-09 ***
Residuals           472.94 144

> Anova(mod2)

                    Sum Sq  Df F value    Pr(>F)
fertilizer          119.49   1  36.383 1.302e-08 ***
variety            1732.82   2 263.805 < 2.2e-16 ***
fertilizer:variety  146.10   2  22.242 3.822e-09 ***
Residuals           472.94 144
```

Comparaison de modèles emboîtés

Attention également à la paramétrisation du modèle.
Pour faire le test de type III pour le fertilisant, on pourrait être tenté de comparer les deux modèles suivants qui sont en fait identiques mais paramétrés autrement :

```
> mod1 <- lm(tomato ~ fertilizer + variety + fertilizer:variety , data=d)
> mod2 <- lm(tomato ~  variety + fertilizer:variety  , data=d)
> anova(mod1,mod2)
Analysis of Variance Table

Model 1: tomato ~ fertilizer + variety + fertilizer:variety
Model 2: tomato ~ variety + fertilizer:variety
  Res.Df    RSS Df Sum of Sq F Pr(>F)
1    144 472.94
2    144 472.94  0         0
```

Si vous respectez le principe de marginalité les paramètres auront toujours la même signification dans les différents modèles

Comparaison de modèles emboîtés

Conclusions

Prudence quand il y a des interactions en particulier quand on ne respecte pas la règle de marginalité !

Si vous voulez un tableau d'analyse de la variance synthétique, utilisez de préférence les comparaisons de type II (fonction Anova)

Encore mieux :
construisez vous-même les modèles correspondant aux hypothèses que vous voulez tester et comparez les avec anova()

Si une interaction est significative, tester ses effets principaux est souvent (mais pas toujours...) inutile.

Conditions d'application des modèles

Conditions d'application des modèles

On discutera en détail tous ces points dans la dernière partie. Ici on va se concentrer sur ces 3 points et en particulier sur la linéarité

Les présupposés du modèle
("model assumptions")
par ordre d'importance :

1) adéquation/validité
2) linéarité - additivité
3) indépendance des résidus
4) homogénéité de la variance des résidus
5) distribution normale des résidus

Autre points à garder en tête :
- influence des données extrêmes (outliers)
- indépendance des variables explicatives
- faut-il centrer, standardiser les données ?
- overfitting : sélection de modèle nécessaire ?
- Quel type de tests (type I, II, III, ...) ?
- l'erreur de mesure des X doit être négligeable

Distribution normale des résidus

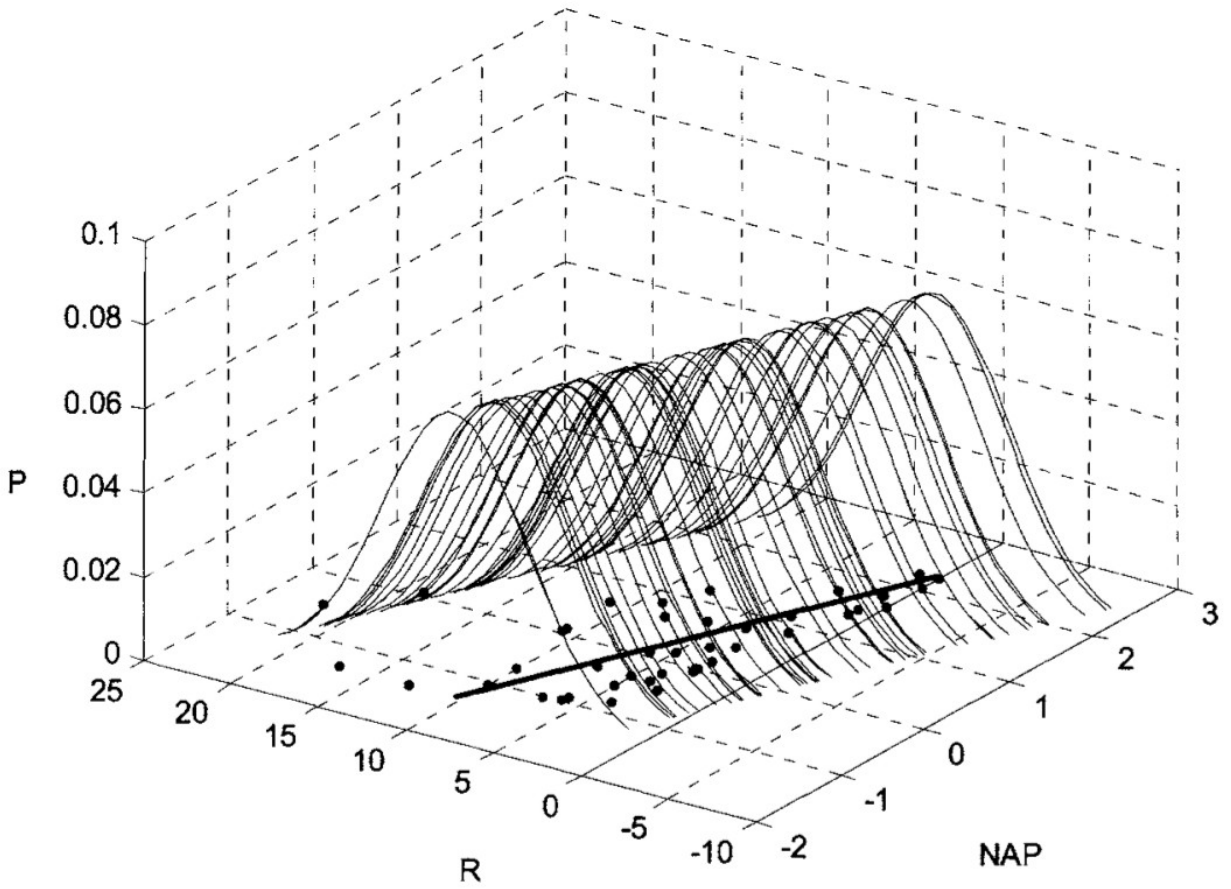
C'est le critère le plus connu, c'est aussi celui qui est le moins important et pour lequel le modèle est le plus robuste

La non-normalité n'affecte pas l'estimation des paramètres (pour les modèles estimés avec la méthode des moindre carrés) mais peut dans des cas extrêmes affecter les inférences classiques.

Dans ce cas on peut de toute façon s'en sortir avec des tests par permutation ou par du bootstrap.

Distribution normale des résidus

Pour chaque valeur de chaque variable explicative, la distribution des résidus devrait approximativement suivre une distribution normale.



Distribution normale des résidus

C'est souvent impossible à vérifier par manque de répétitions pour chaque valeur de x.
On se contente en général de regarder les résidus de manière globale (pour toutes les valeur de x groupées).

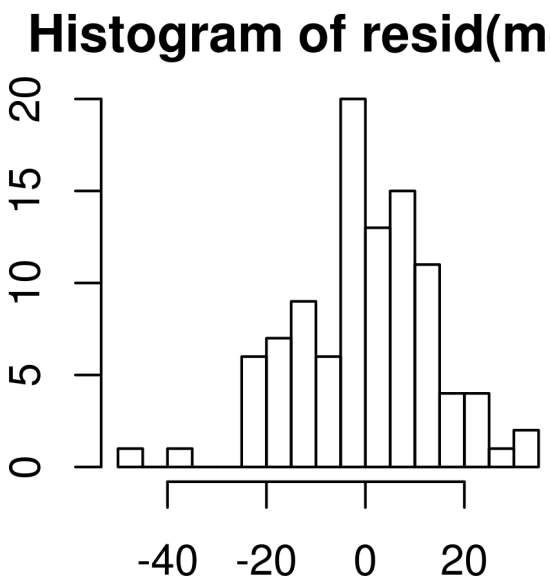
Typiquement on trace deux types de graphiques :
qqplot (quantile - quantile plot) et histogramme des résidus

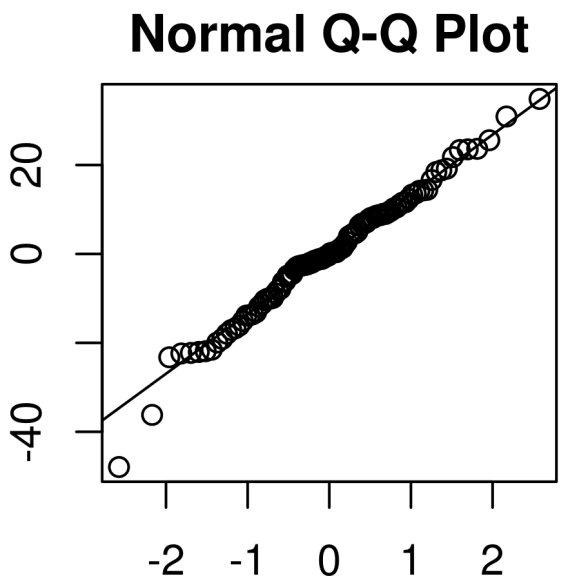
le qq plot normal compare les résidus observés avec des résidus issus de simulations de lois normales ayant les mêmes caractéristiques (moyenne, variance).
On peut aussi utiliser des qqplot pour d'autres distributions

Si la distribution est normale, les points s'alignent sur la droite.

Distribution normale des résidus

```
> n <- 100
> x1 <- runif(n, 0, 20)
> x2 <- runif(n, 0, 20)
> y <- x1 + x2 +  rnorm(n, 0, 15)
> mod <- lm(y ~ x1 + x2)
>
> par(mfrow=c(1,2), mar = c(2,2,2,1))
> hist(resid(mod), breaks=15)
> qqnorm(resid(mod))
> qqline(resid(mod))
```

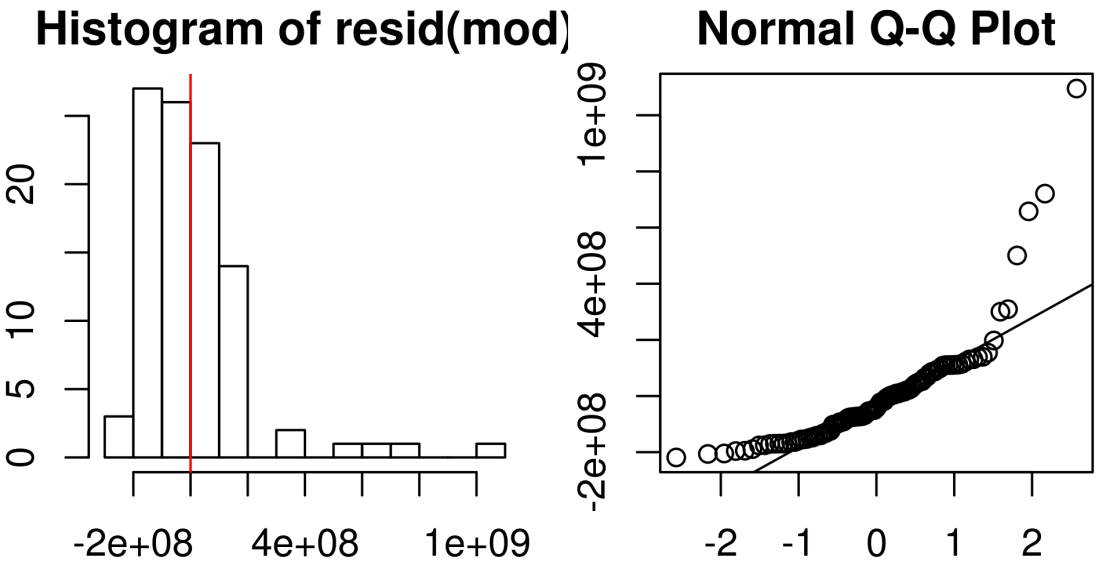




Distribution normale des résidus

```
lambda <- exp(0.1*x1 + x2)
y <- rpois(n, lambda)
mod <- lm(y ~ x1 + x2)

par(mfrow=c(1,2), mar = c(2,2,2,1))
hist(resid(mod), breaks=15)
abline(v=0, col ="red")
qqnorm(resid(mod))
qqline(resid(mod))
```

données avec une distribution de Poisson



Distribution normale des résidus

Ce sont bien les résidus qui doivent avoir globalement une distribution normale, PAS la variable dépendante Y !

Si Y a une distribution normale , les résidus auront souvent une distribution normale.

Mais on peut avoir des données Y pas du tout normales et des résidus suivant parfaitement une distribution normale !

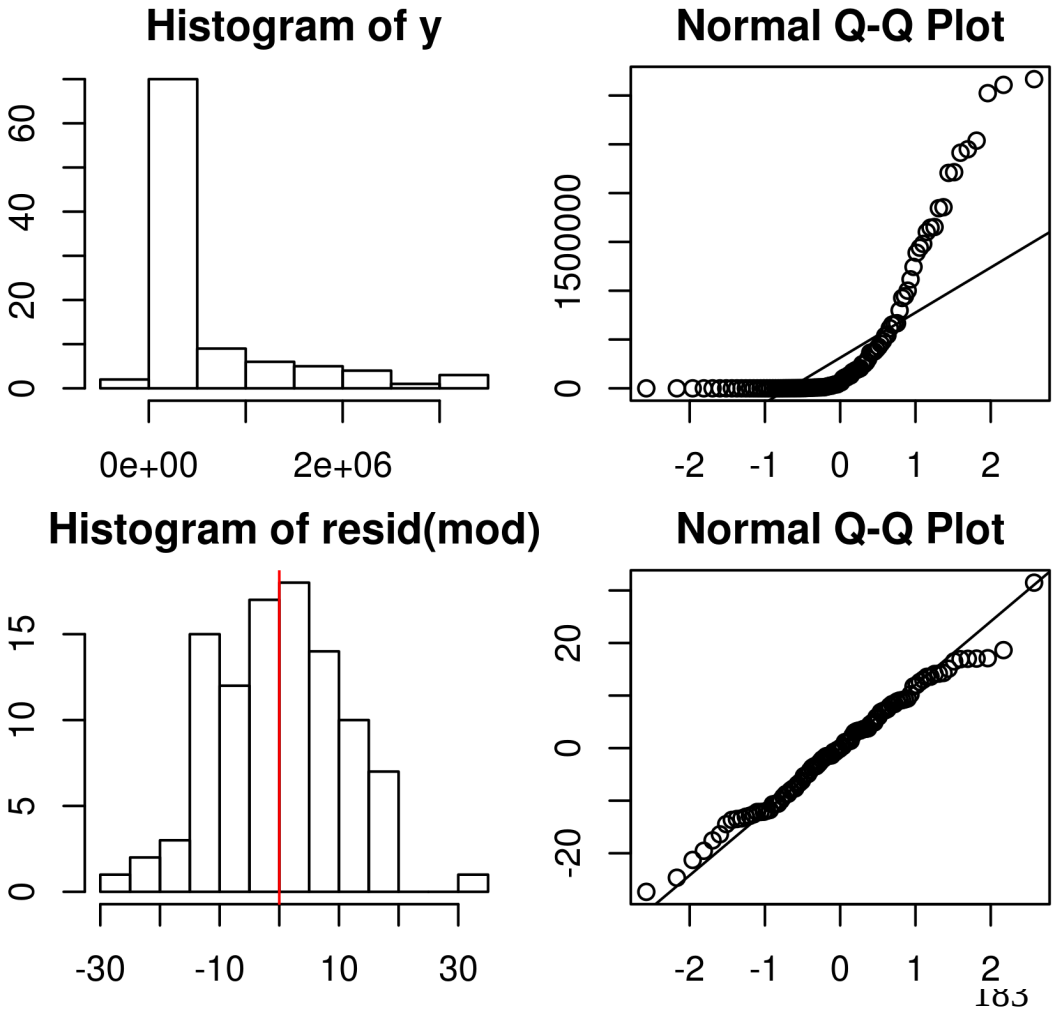
Les variables explicatives ne doivent pas avoir de distribution normale du tout.
Mais il est parfois souhaitable de limiter l'influence des valeurs extrêmes dans les X
(lorsque la distribution des x est très dissymétrique).

NB : en réalité les Y doivent bien suivre une distribution normale pas globalement mais pour chaque valeur de x ce qui est en général impossible à vérifier.

Distribution normale des résidus

Exemple de données Y pas du tout normales mais dont les résidus du modèle la modélisant suivent parfaitement une distribution normale.

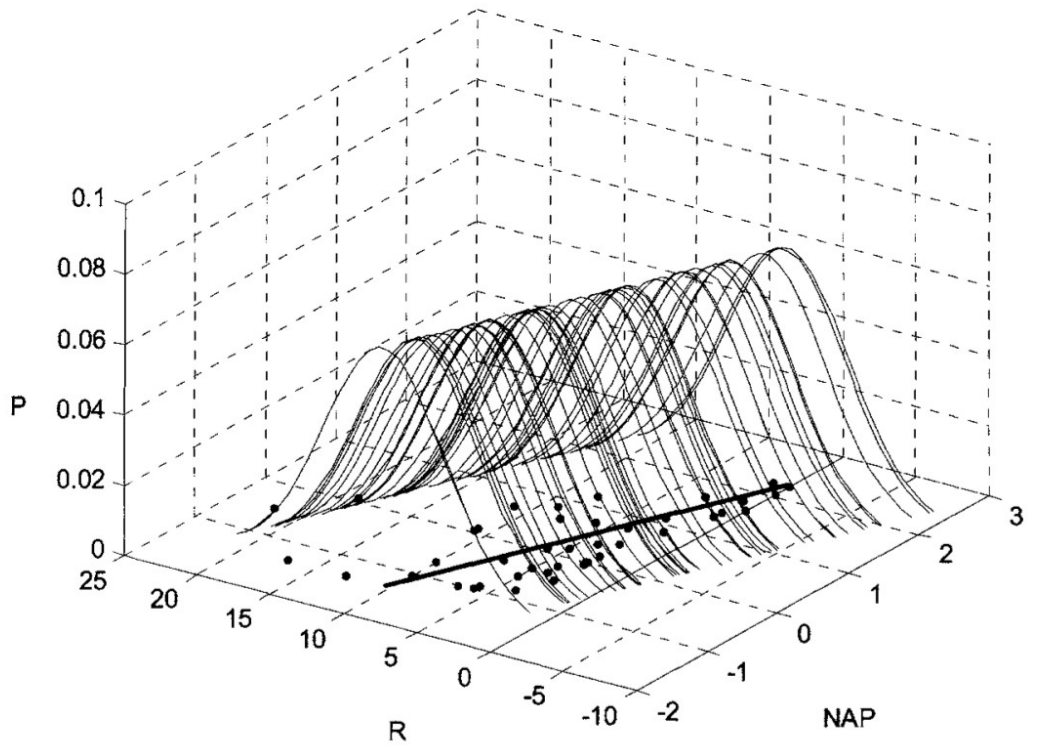
```
n <- 100
x1 <- runif(n, 0, 20)
y <- x1^5 + rnorm(n,0,10)
mod <- lm(y ~ I(x1^5))

par(mfrow=c(2,2), mar = c(2,2,2,1))
hist(y, breaks = 10)
qqnorm(y)
qqline(y)

hist(resid(mod), breaks=15)
abline(v=0, col ="red")
qqnorm(resid(mod))
qqline(resid(mod))
```



Homogénéité de la variance des résidus

Pour chaque valeur de chaque variable explicative, la distribution des résidus devrait approximativement suivre une distribution normale de variance constante

Le modèle estime en effet une seule valeur unique pour la variance résiduelle.



Homogénéité de la variance des résidus

L'hétéroscédasticité (variances non homogène) n'a pas d'effet sur l'estimation des paramètres mais peut dans certains cas extrêmes affecter les inférences.

Cette règle est plus importante que la normalité mais il faut en général une forte hétéroscédasticité pour rencontrer des problèmes

Comme pour la normalité, on ne peut en général pas vérifier directement ce présupposé.

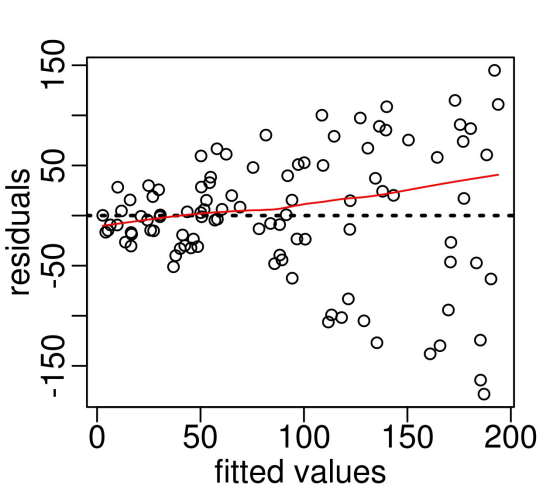
En général on trace un graphique des résidus en fonction des valeurs prédites et/ou de chaque variable explicative.
Il faut essayer d'éviter les cas où la variance résiduelle augmente avec la moyenne (forme triangulaire du nuage de points)

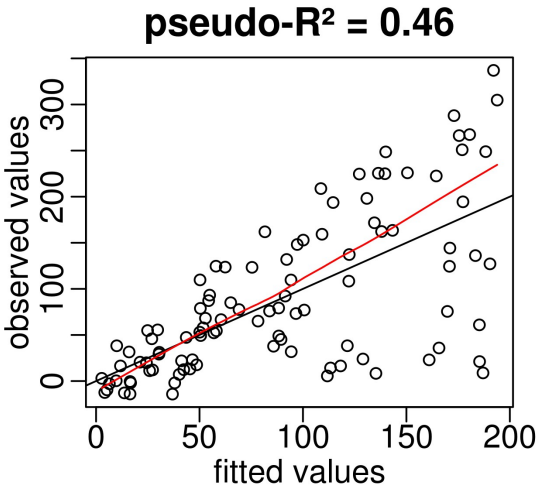
Les fonctions `diagplot()` et `diagplot2()` dans mytoolbox.R
Tracent ces graphiques ainsi que les qqnorm et histogramme

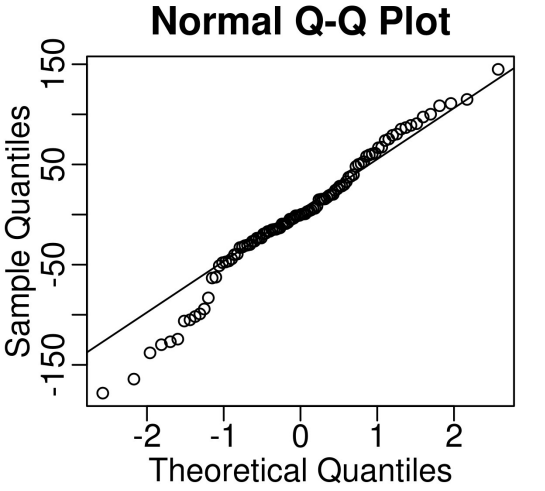
Homogénéité de la variance des résidus

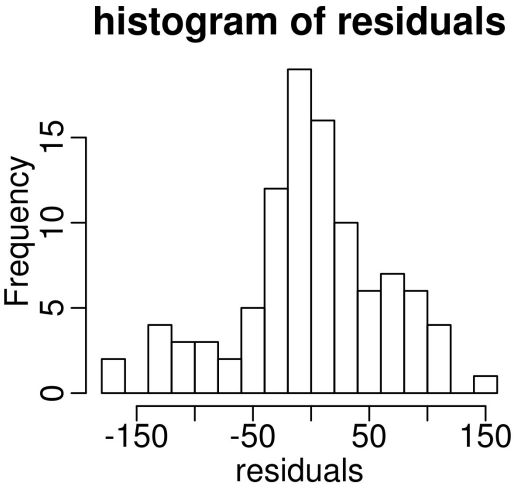
Exemple où le nuage de résidus a une forme triangulaire : la variabilité augmente quand les valeurs prédites augmentent : A éviter !

```
n <- 100
x1 <- runif(n, 0, 20)
x2 <- runif(n, 0, 20)
y <- x1 * x2 +  rnorm(n, 0, 15)

mod <- lm(y ~ x1 )
diagplot(mod)
```





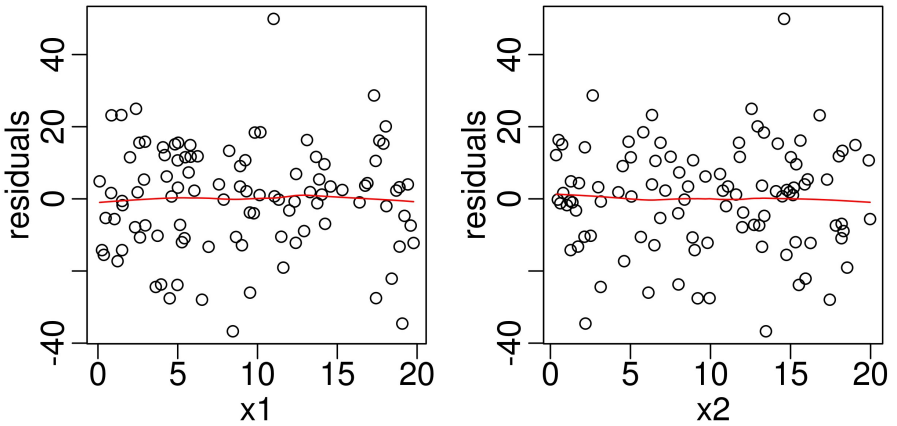


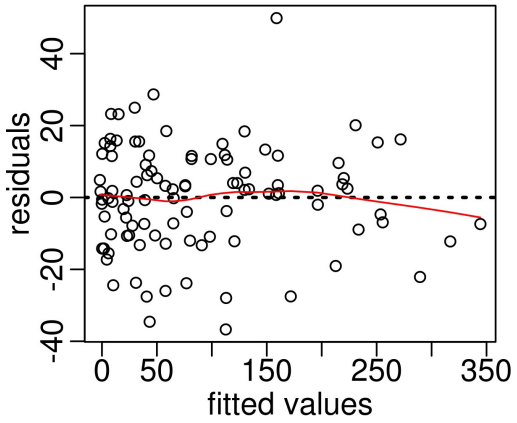


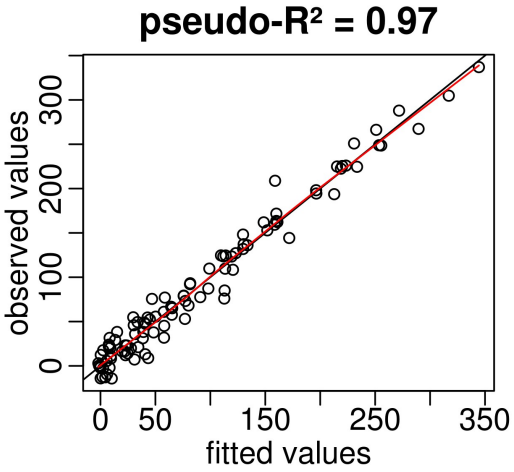
Homogénéité de la variance des résidus

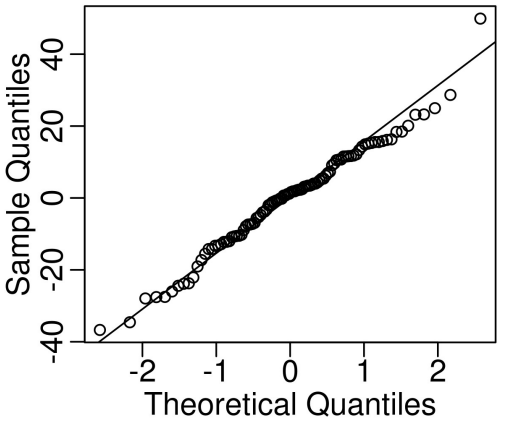
Le problème est totalement résolu en ajoutant une deuxième variable explicative et l'interaction

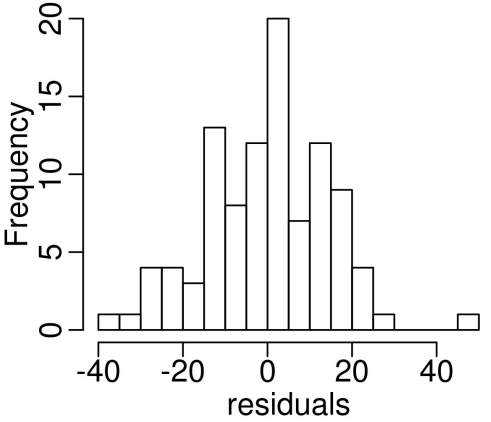
```
mod <- lm(y ~ x1*x2 )
diagplot(mod)
diagplot2(mod)
```











Homogénéité de la variance des résidus

A propos des tests de normalité et d'homoscedasticité

Ces tests sont en général inutiles

Si vous avez beaucoup de données, ils vont mettre en évidence des écarts très faibles à la normalité et l'homoscedasticité.
Or on a vu que les modèles linéaires sont robustes à des écarts de faible à moyenne importance.

Si vous avez très peu de données, vous pourriez "rater" des différences (mais c'est vrai aussi avec des plots de résidus)

Les graphiques de résidus sont amplement suffisants et ils sont bien plus informatifs

Homogénéité de la variance des résidus

A propos des tests de normalité et d'homoscedasticité

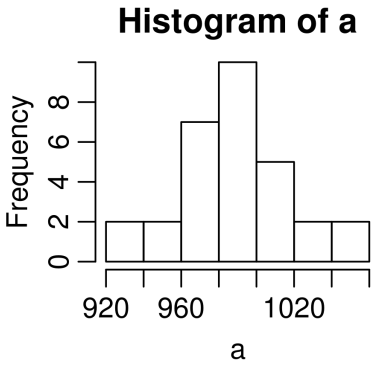
```
set.seed(11)
a <- rpois(30, 1000)
a10 <- rep(a,10)

par(mfrow = c(2,1))
hist(a, breaks = 6)
hist(a10, breaks = 6)

> shapiro.test(a)

     Shapiro-Wilk normality test

data:  a
W = 0.98191, p-value = 0.8737

> shapiro.test(a10)

     Shapiro-Wilk normality test

data:  a10
W = 0.97301, p-value = 2.016e-05
```

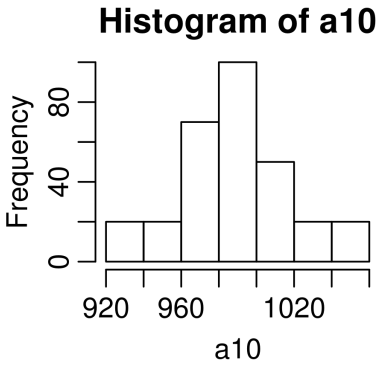
a = 20 obs avec une distribution de poisson de moyenne = 1000

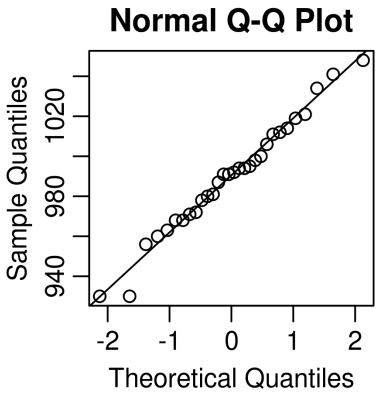
a10 = a répété 10 fois --> la distribution est exactement la même !

a ne diffère pas significativement d'une distribution normale

a10 diffère significativement d'une distribution normale mais on a plus de données donc a10 pose encore moins de problèmes que a







Comme toujours il est délicat de baser son jugement en regardant seulement une p-valeur. Il faut aussi regarder les données...
Si les résidus d'un modèle ont une telle distribution vous pouvez probablement sans grand danger utiliser un modèle gaussien, quoiqu'en dise le test de shapiro...

Linéarité

Un modèle linéaire est une combinaison linéaire de paramètres c'est à dire un modèle ou les paramètres à estimer ne se trouvent ni en exposant ni multiplié ou divisé par un autre paramètre.

Un modèle comme y ~ x + x^2 est donc un modèle linéaire bien que la relation entre x et y ne soit pas une ligne droite ...

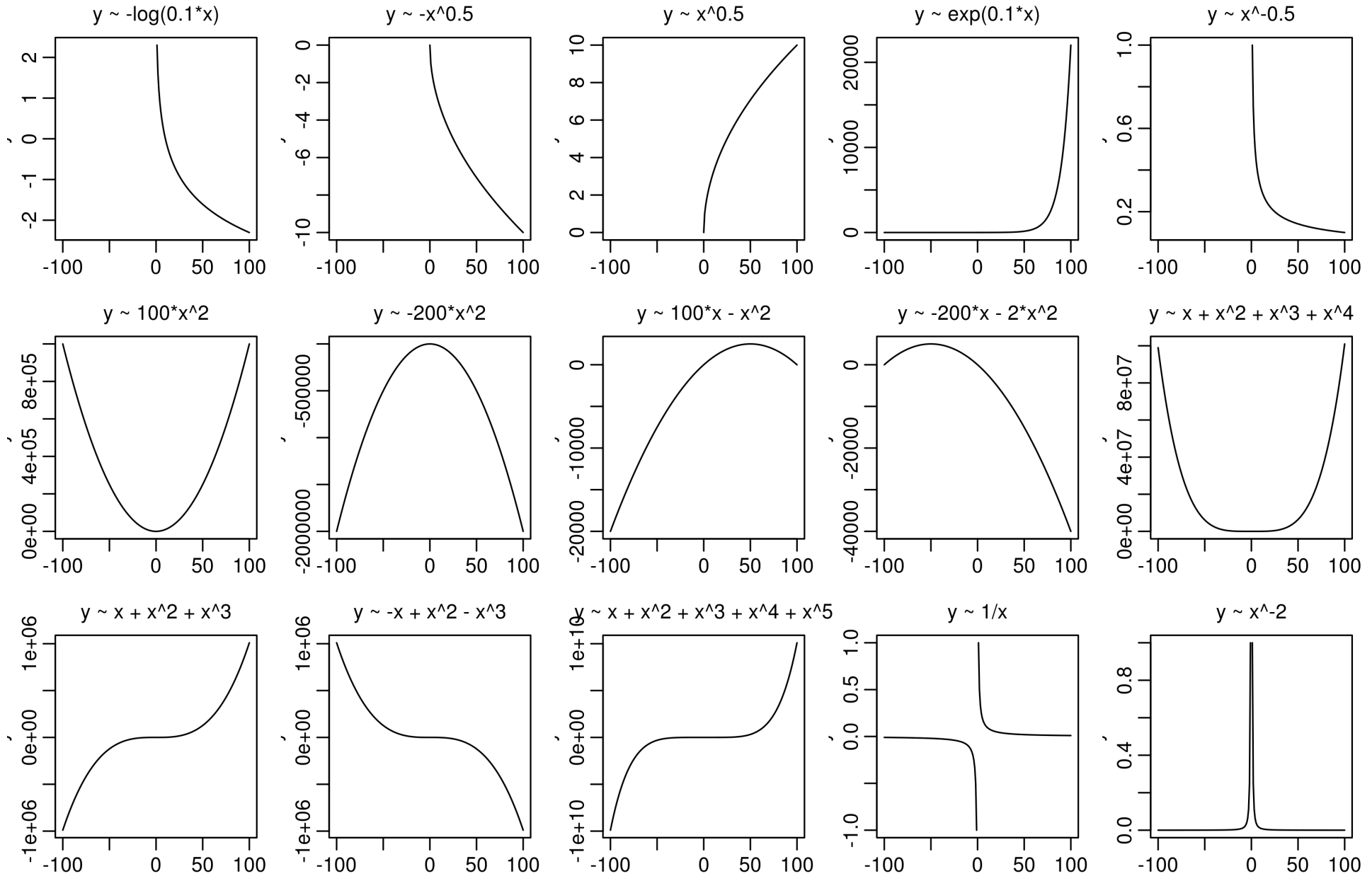
Cette définition est peu intuitive ...
Par facilité, on parlera par la suite de problèmes de linéarité quand la relation entre y et x ne suit pas une ligne droite.

Dans ce cas le modèle ajuste mal les données et c'est un problème bien plus important que la non normalité ou l'hétéroscedasticité.

La non linéarité affecte directement l'estimation des paramètres, la forme du modèle et les prédictions.
Les inférences sont correctes mais sur un mauvais modèle...

Linéarité

Avec un modèle linéaire, on peut modéliser ce genre de relations :



Linéarité

Comment détecter les problèmes de non linéarité ?

On peut faire des graphiques de la variable dépendante en fonction de chaque variable explicative. (pex scatterplot matrix)

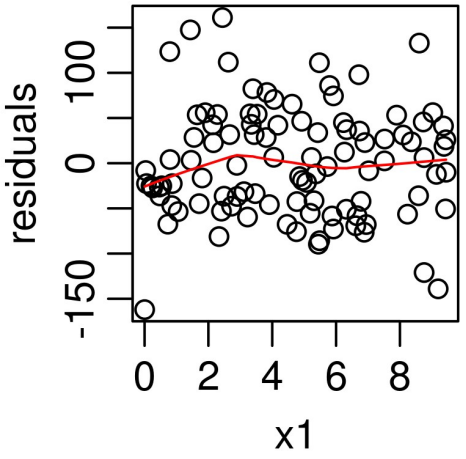
Mais ce n'est pas suffisant car souvent le bruit masque des problèmes de non linéarité.

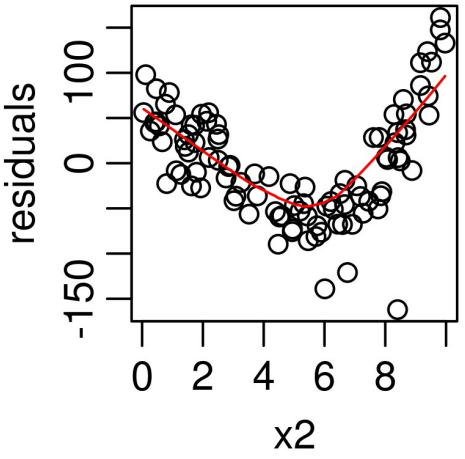
L'idéal est ici aussi de faire des graphiques des résidus en fonction des valeurs prédites et en fonction de chaque variable explicative pex avec `diagplot` et `diagplot2`

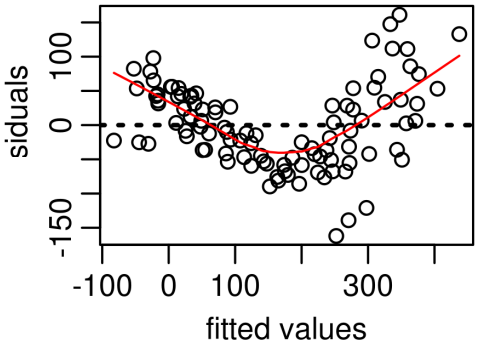
Si le nuage de point n'est pas uniformément réparti de part et d'autre du 0, le modèle surestime ou sous-estime les données dans une partie de l'espace du modèle.

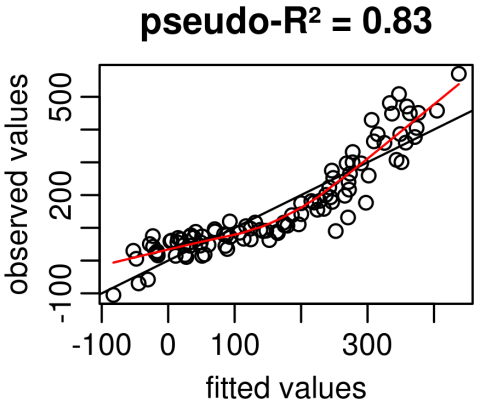
Linéarité

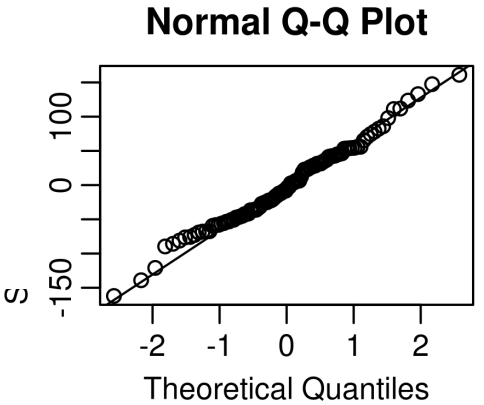
Comment détecter les problèmes de non linéarité ?

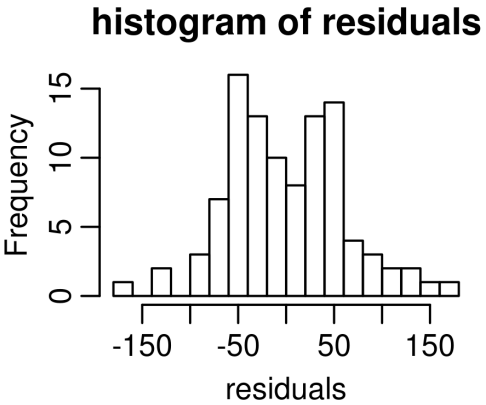












Linéarité

Quelles solutions aux problèmes de non linéarité ?

Il existe de nombreuses possibilités (cfr fin de ce module).

On va explorer ici
les transformations de variables et les régressions polynomiales.

NB : on transforme aussi souvent les variables pour d'autres raisons que la linéarisation (ea homogénéisation des variances, normalisation des résidus etc...)

On peut à la fois transformer les Y et les X

Les transformations les plus fréquentes sont des transformations log, sqrt, ^2, ^3, ^1/3, 1/x, etc...

Linéarité

Régression polynomiale

Une régression polynomiale a par exemple la forme suivante :

```
y ~ x1 + x1^2 + x2 + x2^2 + x2^3
```

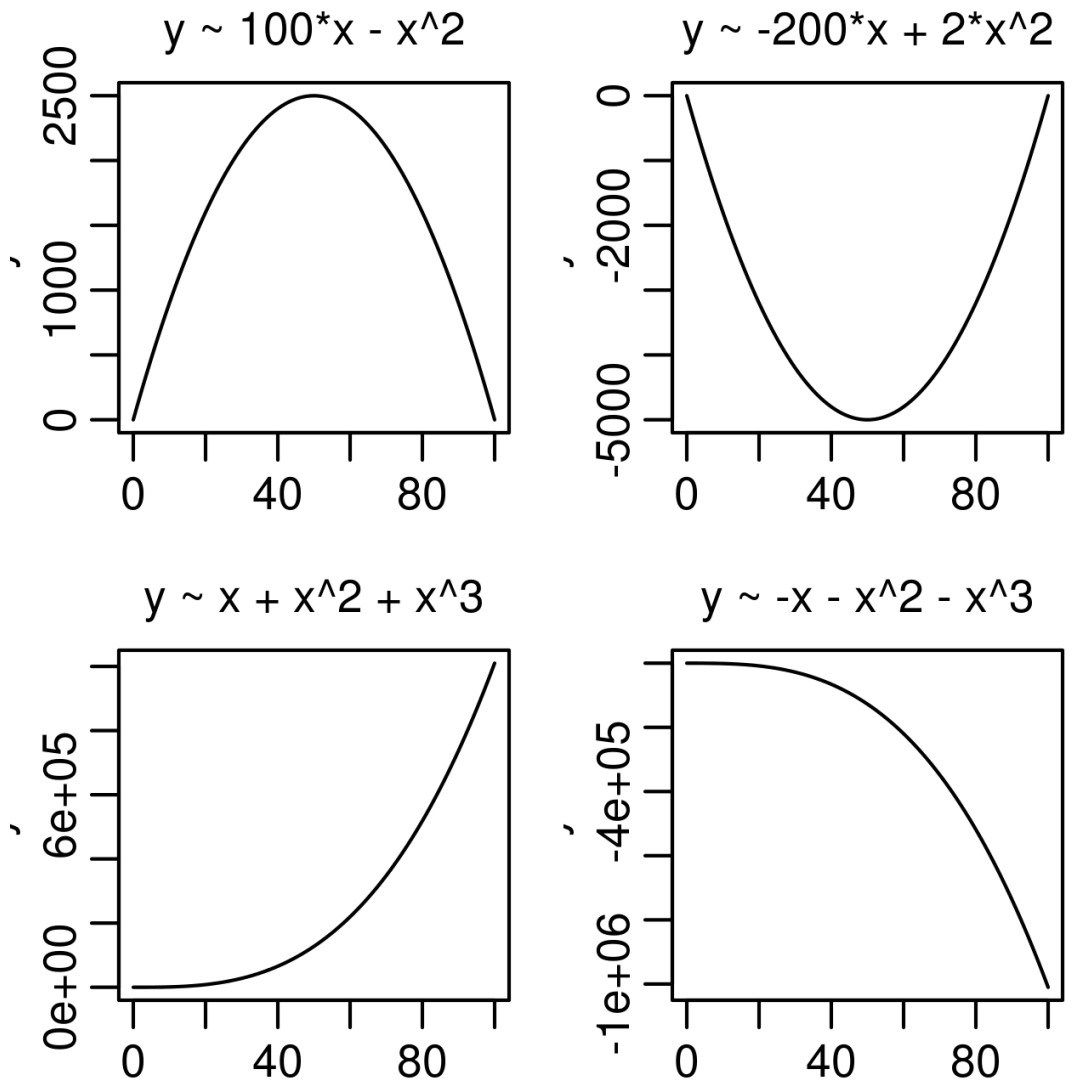
On dit qu'il s'agit d'une polynomiale d'ordre 2 pour x1 et d'ordre 3 pour x2. On appelle parfois x1^2 un "terme quadratique" et x2^3 un "terme cubique".

La régression polynomiale d'ordre 2 a la forme d'un U ou d'un U inversé. Contrairement à un simple modèle y ~ x^2, l'optimum ou le minimum peut se trouver ailleurs que sur le 0.

La polynomiale d'ordre 3 ou plus peut avoir des formes assez variées en fonction des paramètres et de la position de l'intercept

Linéarité

Régression polynomiale



Régression polynomiale

Une régression polynomiale a par exemple la forme suivante :

$$y \sim x1 + x1^2 + x2 + x2^2 + x2^3$$

On dit qu'il s'agit d'une polynomiale d'ordre 2 pour x1 et d'ordre 3 pour x2. On appelle parfois x1^2 un "terme quadratique" et x2^3 un "terme cubique".

La régression polynomiale d'ordre 2 a la forme d'un U ou d'un U inversé. Contrairement à un simple modèle y ~ x^2, l'optimum ou le minimum peut se trouver ailleurs que sur le 0.

La polynomiale d'ordre 3 ou plus peut avoir des formes assez variées en fonction des paramètres

Linéarité

Comment choisir les transformations ?

En général l'expérience guide les choix de transformation...
Voici quelques règles d'aide à la décision qui n'ont rien d'absolu...

Si il y a un pattern dans le graphique résidus vs valeurs prédites mais pas dans les graphiques des résidus vs variables explicatives, transformez plutôt la variable dépendante Y.
Si les patterns se trouvent dans les plots résidus vs variables explicatives, transformez plutôt les variables explicatives x

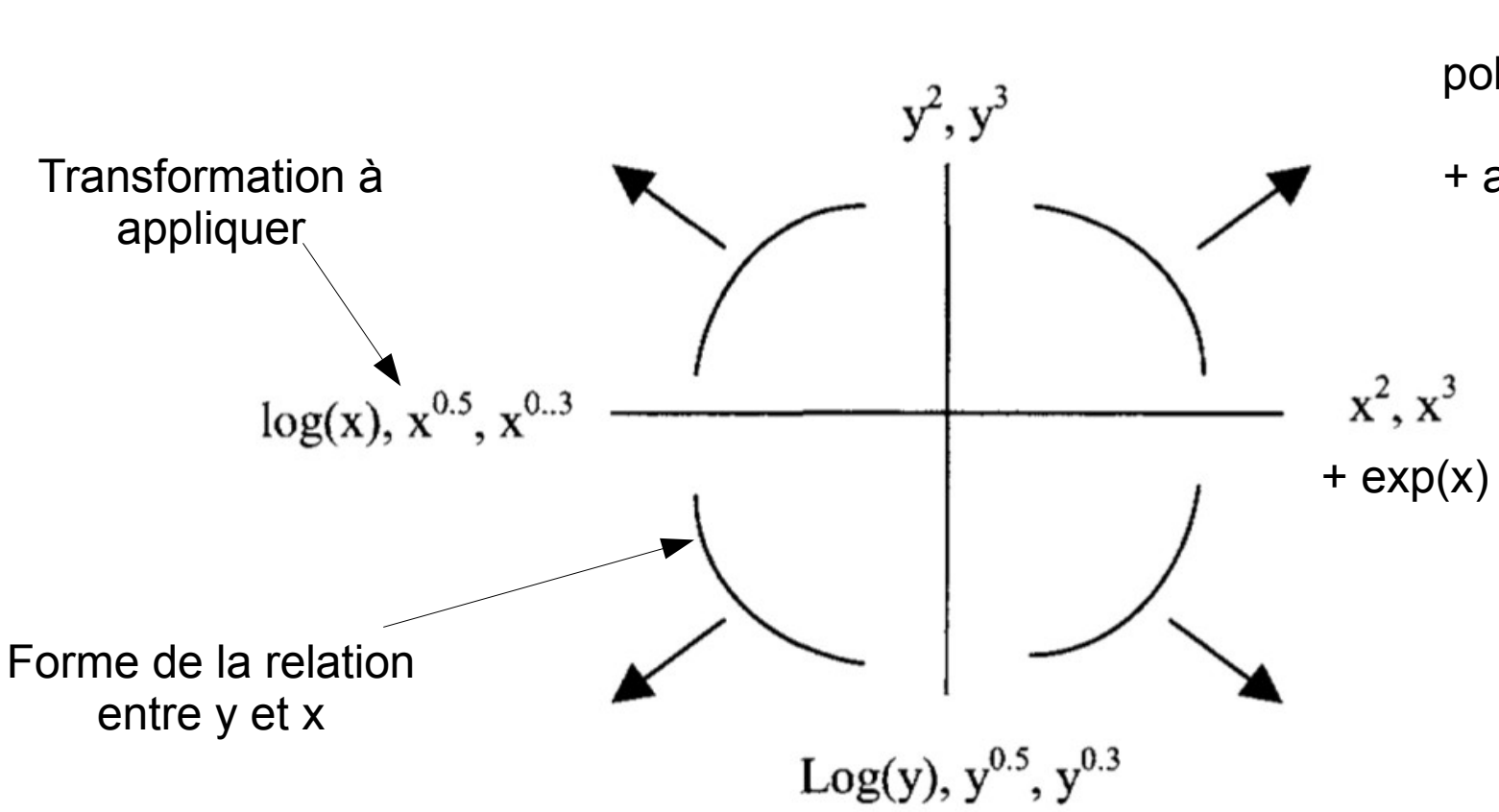
Si il n'y a PAS de relation positive ou négative nette entre Y et x, et qu'il y a un pattern en U ou U inversé, utilisez une polynomiale d'ordre 2.

Si il y a une relation positive ou négative et un pattern dans les résidu, aidez vous de la "Mosteller and Tukey's bulging rule"

Linéarité

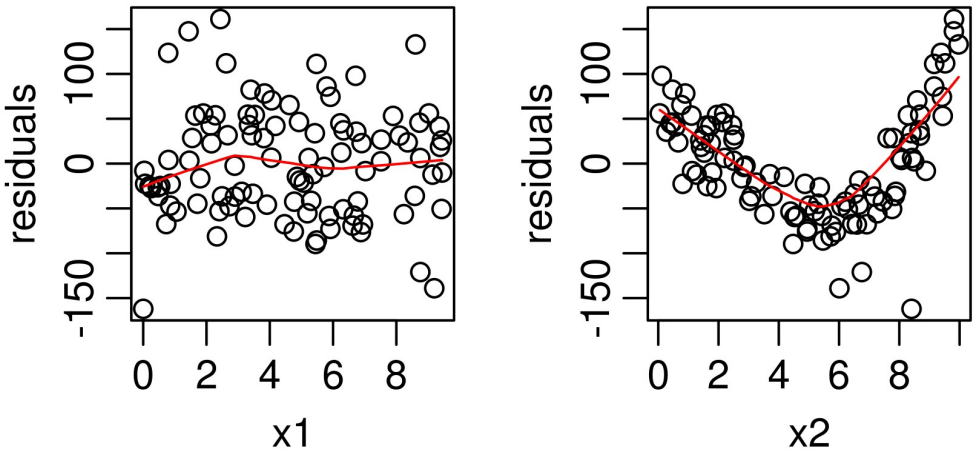
Comment choisir les transformations ?

"Mosteller and Tukey's bulging rule"
On procède par essais-erreurs en vérifiant l'effet de la transformation sur les graphiques de résidus

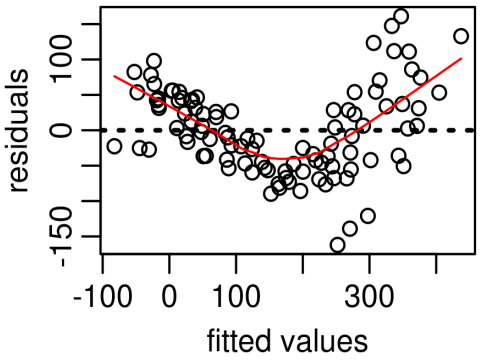


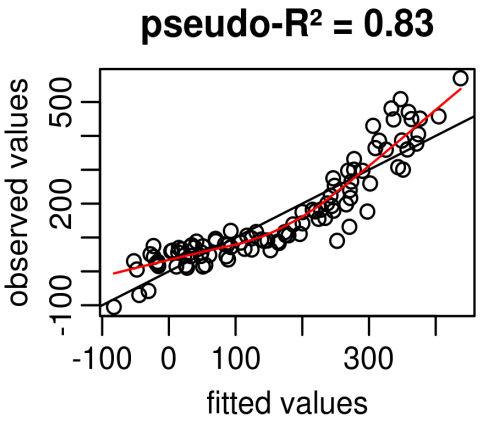
Linéarité

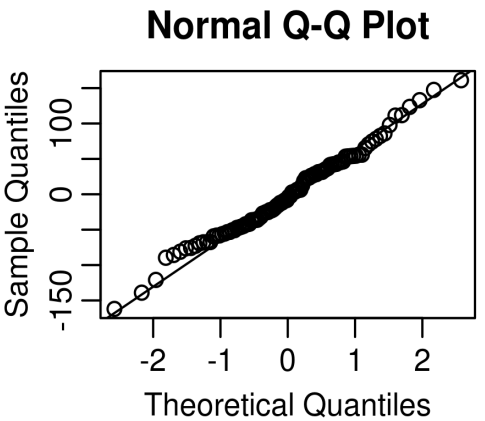
Comment choisir les transformations ?

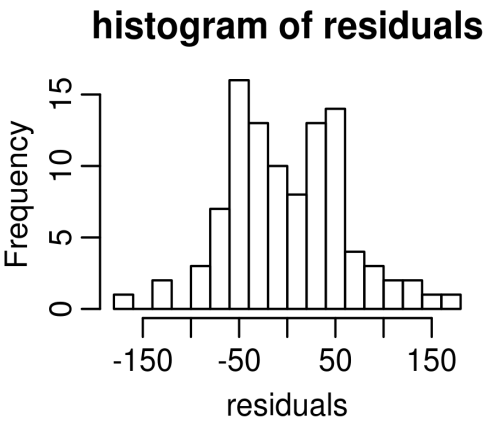



```
x1 <- runif(100, 0, 10)
x2 <- runif(100, 0, 10)
y <- 30* log(x1) +
     0.5 * x2^3 +
     rnorm(100, 0, 20)

mod <- lm(y ~ x1 + x2)
diagplot(mod)
diagplot2(mod)
```



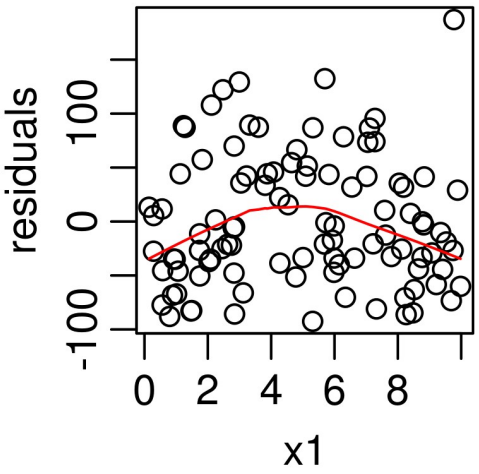


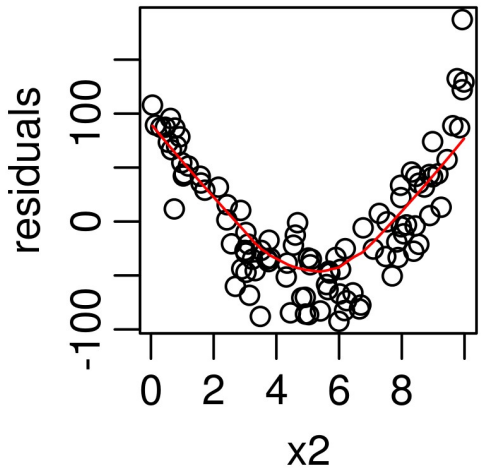


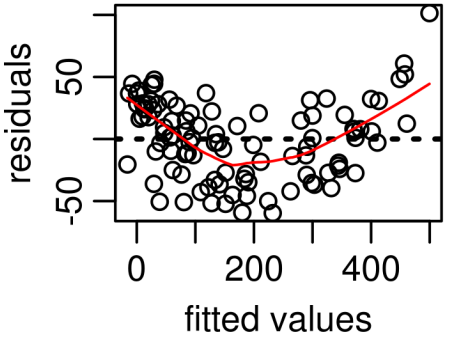


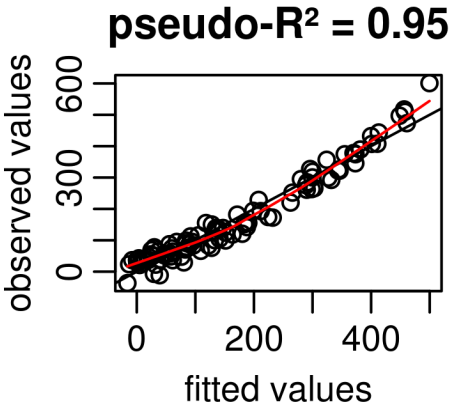
Linéarité

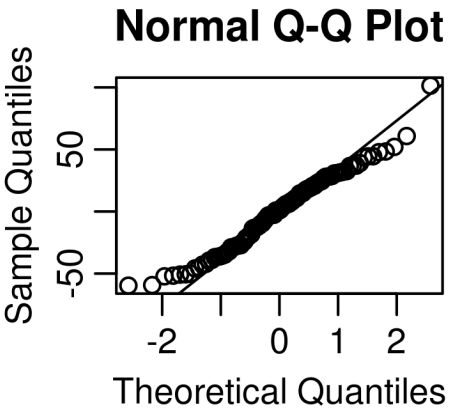
```
mod <- lm(y ~ x1 + I(x2^2))
diagplot(mod)
diagplot2(mod)
```

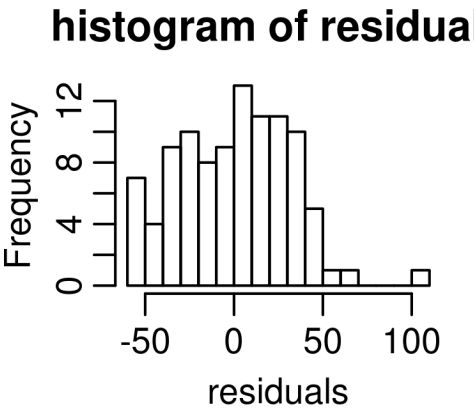






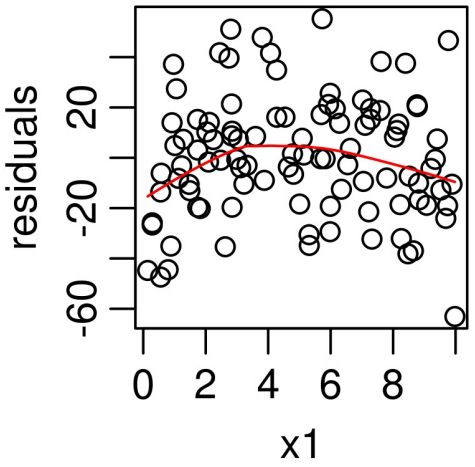


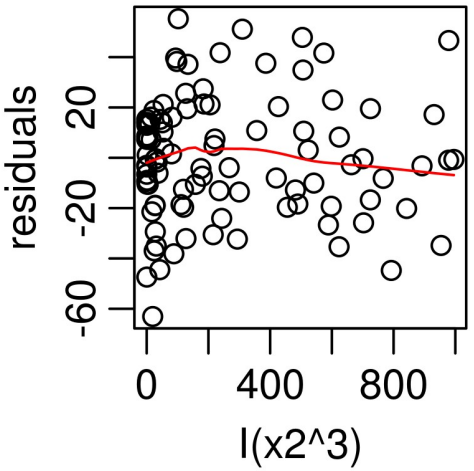


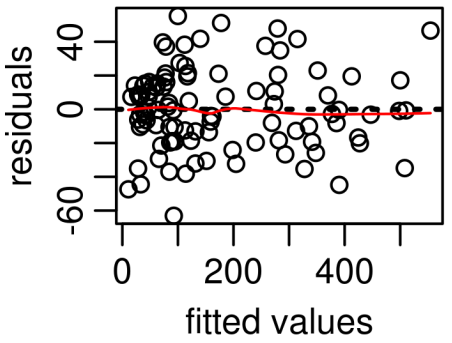


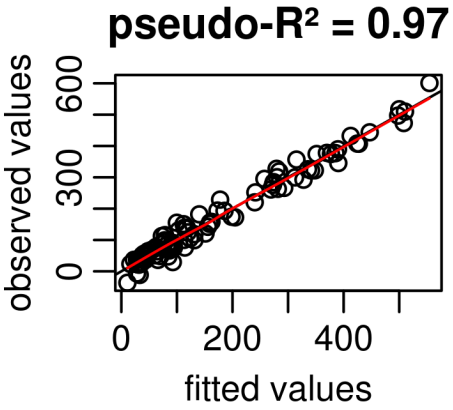
Linéarité

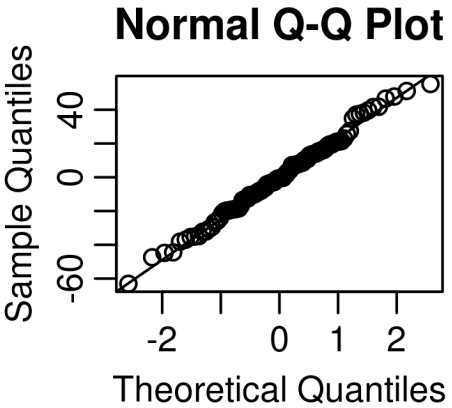
```
mod <- lm(y ~ x1 + I(x2^3))
diagplot(mod)
diagplot2(mod)
```

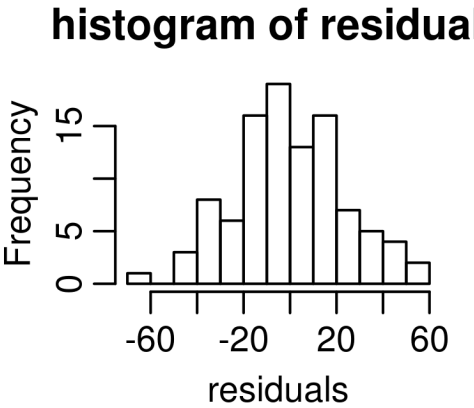






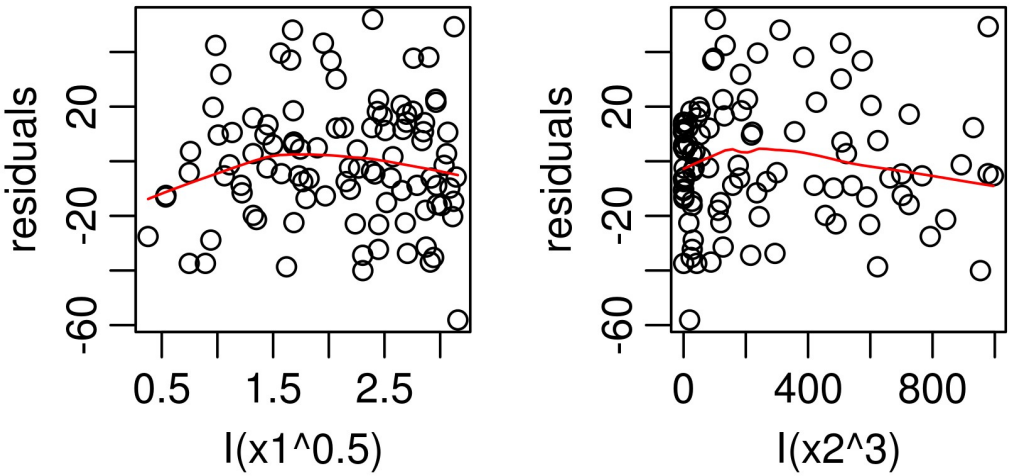


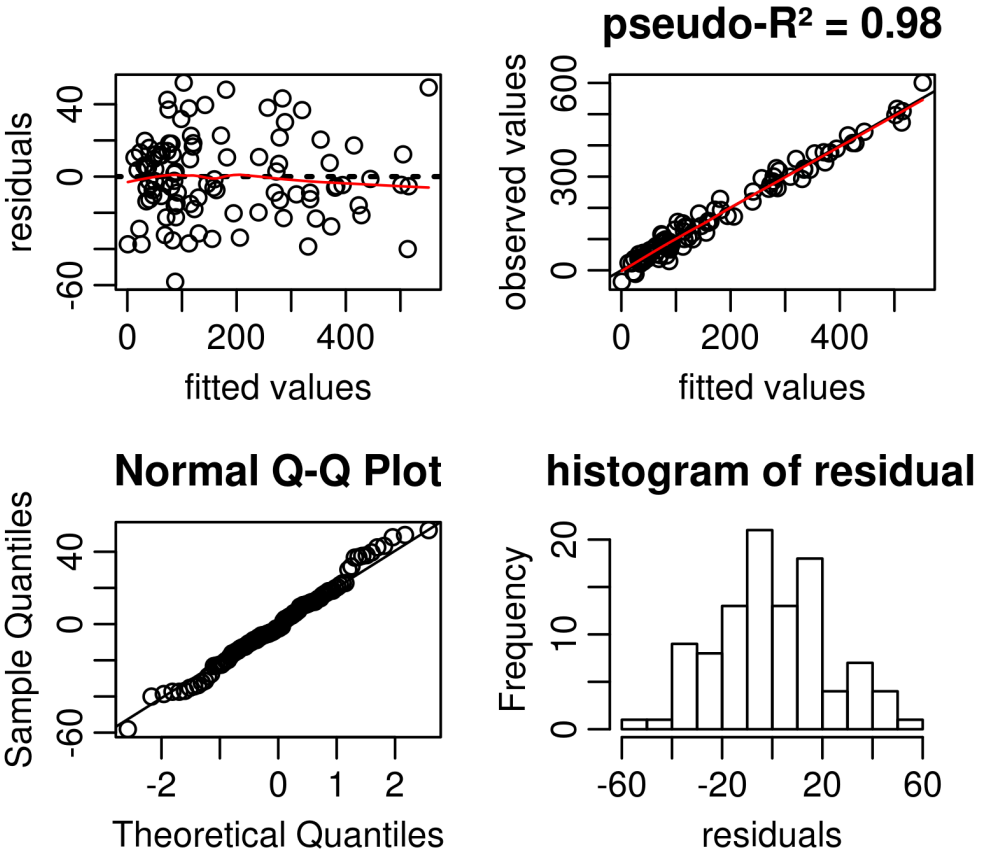




Linéarité

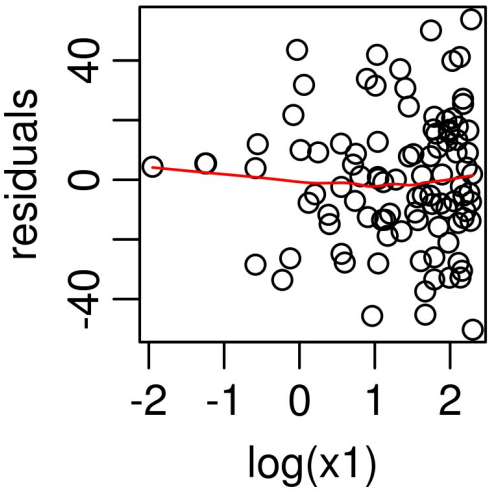
```
mod <- lm(y ~ I(x1^0.5) + I(x2^3))
diagplot(mod)
diagplot2(mod)
```

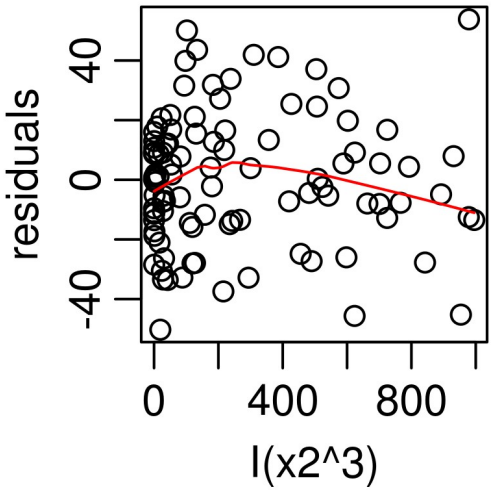


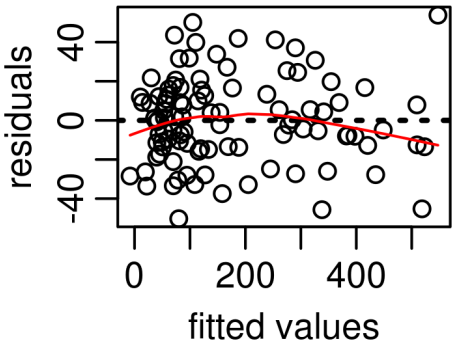


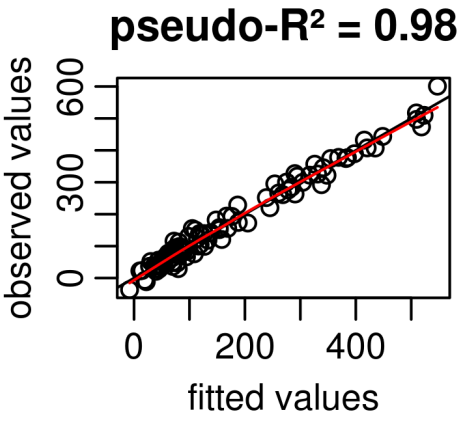
Linéarité

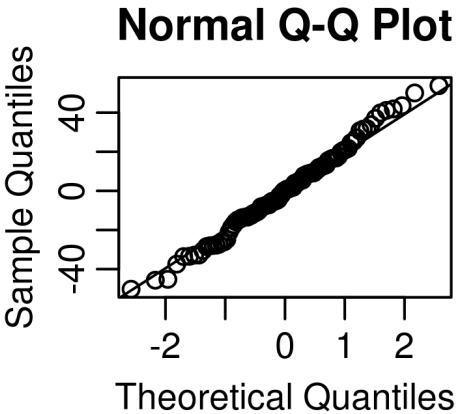
```
mod <- lm(y ~ log(x1) + I(x2^3))
diagplot(mod)
diagplot2(mod)
```

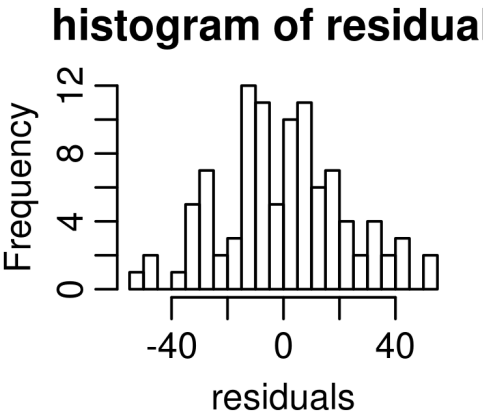












Linéarité

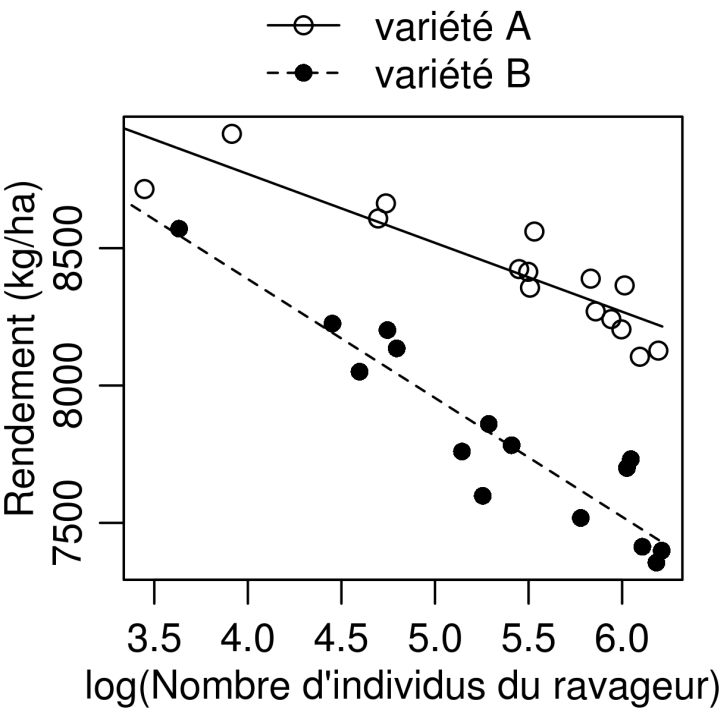
Représentation graphique des relations transformées

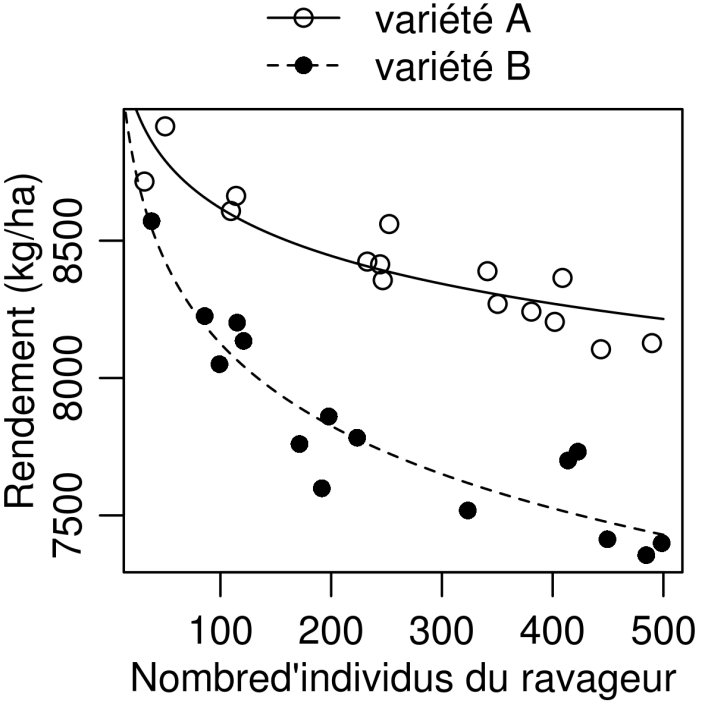
Il faut idéalement représenter les données sur l'échelle d'origine, non transformée pour faciliter l'interprétation.

Pour ce faire, il faut faire les prédictions sur l'échelle transformée puis, juste avant de tracer le modèle, utiliser la transformation inverse des valeurs prédites ou des x.

transformation	inverse
log(x), log10(x)	exp(x), 10^x
sqrt(x) = x^0.5	x^2
x^(1/3), x^0.25	x^3, x^4
1/x = x^-1	1/x = x^-1

```r
d <- data.frame(
    ravageur = runif(n, 0, 500),
    var = rep(c("varA", "varB"), each = n/2))
d$logravageur <- log(d$ravageur)
X <- model.matrix(~ logravageur*var, data=d)
B <- c(10000, -300, 200, -150)
d$rendement <- X %*% B + rnorm(n, 0, 100)
mod <- lm(rendement ~ log(ravageur) * var, data=d)

X1 <- cbind(1, log(0:500), 0, 0)
X2 <- cbind(1, log(0:500), 1, 1*log(0:500))
pred1 <- X1 %*% coef(mod)
pred2 <- X2 %*% coef(mod)

par(mfrow= c(2,1), mar = c(3,3,3,1),
    mgp = c(1.7, 0.7, 0))
plot(rendement~ logravageur, data=d,
     pch = c(1,16)[as.numeric(d$var)],
     ylab = "Rendement (kg/ha)",
     xlab = "log(Nombre d'individus du ravageur)")
lines(pred1 ~ X1[,2], lty = 1)
lines(pred2 ~ X2[,2], lty = 2)

legend(x = "top", inset = -0.3, xpd = NA,
        bty = "n", lty = 1:2, pch = c(1,16),
        legend = c("variété A", "variété B"))

plot(rendement~ ravageur, data=d,
     pch = c(1,16)[as.numeric(d$var)],
     ylab = "Rendement (kg/ha)",
     xlab = "Nombre d'individus du ravageur")
lines(pred1 ~ exp(X1[,2]), lty = 1)
lines(pred2 ~ exp(X2[,2]), lty = 2)

legend(x = "top", inset = -0.3, xpd = NA,
        bty = "n", lty = 1:2, pch = c(1,16),
        legend = c("variété A", "variété B"))
```





Generalized Linear Models

Programme

Part 2 : Generalized Linear Model (GLM)

La partie concernant les variables explicatives change peu.
On va s'intéresser aux Y et aux distributions des résidus

Généralités

Y = données de comptage (distribution de Poisson)

Y binaire ou % (régression logistique - distribution binomiale)

Tables de contingence (distribution de Poisson)

Surdispersion

Generalized Linear Models

Les Generalized Linear Models (GLM) sont une généralisation des General Linear Models (LM) vus jusqu'à présent.

Le côté "X" (= variables explicatives = "linear predictor") de l'équation ne change pas

Deux choses en plus :

- la distribution qui peut être autre que normale
- une fonction de lien établissant la relation entre la variable dépendante et la combinaison de variables explicatives (=linear predictor)

La méthode d'estimation n'est pas la même (Maximum de vraisemblance vs Moindres carrés)

Generalized Linear Models

Chaque GLM a donc 3 composantes :

Facultatifs

1) Distribution

Y ~ Distribution(paramètres de position, autres paramètres)

2) Fonction de lien (link function)

g(paramètre de position) = $g(E(Y)) = g(\hat{Y})$ = linear predictor

Espérance de Y
= valeurs prédites

3) Linear predictor

Notation Matricielle → $X\beta$

Notation Classique → $= \beta_0 + \beta_1 * x_1 + \beta_2 * x_2 + ...$

Pour chaque fonction de lien, il existe une fonction de lien inverse $g^{-1}()$ qui permet de lier directement Y au "linear predictor".
Les différent composants peuvent donc se résumer comme :

Y ~ Distribution($g^{-1}(X\beta)$, autres paramètres)

Generalized Linear Models

GLM à distribution Gaussienne (=Normale)

Ce sont les modèles vus jusqu'à présent mais estimés avec la méthode du maximum de vraisemblance (comme tous les GLM) au lieu de la méthode des moindre carrés.

Les GLM gaussiens donnent les mêmes résultats que les LM moyennant l'hypothèse que les résidus ont réellement une distribution gaussienne.

Les LM n'ont pas besoin de cette hypothèse pour l'estimation des paramètres, mais uniquement pour l'inférence.

C'est une des raisons pour lesquelles il est recommandé d'utiliser `lm()` et non pas `glm()` quand vous voulez estimer des modèles à distribution normale
On le fera ici juste pour la comparaison ...

Generalized Linear Models

GLM à distribution Gaussienne (=Normale)

On l'utilise pour modéliser des variables continues.

Distribution : $Y \sim \text{Normale}(\mu, \sigma^2)$
Fonction de lien : identité : $\mu = E(Y) = \hat{Y}$ = linear predictor
Linear predictor : $X\beta$

La fonction identité ne change rien et donc l'inverse non plus, on peut donc résumer ce modèle comme :

$$Y \sim \text{Normale}(X\beta, \sigma^2)$$

Ce qui est strictement identique (dans le cas gaussien) à :

$$Y \sim X\beta + Normale(0, \sigma^2)$$

Generalized Linear Models

GLM à distribution de Poisson

On l'utilise pour modéliser des entiers positifs.
Typiquement les données de comptage suivent une distribution de Poisson. Lorsque les nombres comptés sont grands, cette distribution tend vers une distribution normale.

On l'utilisera également pour modéliser des tables de contingence (association entre variables qualitatives)

Distribution : $Y \sim \text{Poisson}(\lambda)$

Fonction de lien log : $\log(\lambda) = log(E(Y)) = log(\hat{Y})$ = linear predictor

Linear predictor : $X\beta$

L'inverse la fonction log() est exp() :

$Y \sim \text{Poisson}(\exp(X\beta))$

GLM à distribution de Poisson

Dans une distribution de Poisson, on estime pas la variance, elle est fixée comme étant égale à la moyenne λ :

$$var(Y) = \lambda$$

Dans certains cas (fréquemment en fait) , la variable étudiée ne suit pas une exactement une distribution de Poisson dans le sens où sa variance est plus élevée que celle prévue par la loi de Poisson.

$$var(Y) = \varphi\lambda$$

Le paramètre φ est appelé paramètre de surdispersion ("overdispersion") ou "scale parameter".

Pour la distribution de Poisson il est fixé à 1
Le non respect de cette hypothèse a des conséquence dramatiques sur les inférences.

Generalized Linear Models

GLM à distribution Binomiale

On l'utilise pour modéliser une autre forme de données de comptage:
1) une proportion p de la forme nombre de succès/nombre d'essais
2) des données binaires qui sont un cas particulier
du cas précédent où le nombre d'essais N = 1
Lorsque N est grand et que p est proche de 0.5, cette distribution tend vers une distribution normale.

Distribution : Y ~ Binomiale (p, N)

Fonction de lien : logit : logit(p) = $log(p/(1-p))$ = linear predictor

Linear predictor : $X\,\beta$

Nom officieux

L'inverse la fonction logit() est invlogit() :

Y ~ Binomiale ($\exp(X\,\beta) / (1 + \exp(X\,\beta)$, N)

GLM à distribution Binomiale

Ici aussi la variance est une fonction de la moyenne avec un paramètre de surdispersion φ fixé à 1 :

$$\text{Var}(Y) = \varphi\, Np(1-p)$$

La surdispersion aura également des conséquences importantes sur les inférences.
Pour des données binaires, φ est toujours à peu près = 1

Autres distributions pour les GLM

Les distributions Gaussienne, de Poisson et Binomiale permettent de modéliser une grande variété de problèmes courants et ce sont de loin les plus utilisées.
On se concentrera sur ces distributions ici.

Certaines autres distributions sont proches des distribution de Poisson et Binomiale mais avec un paramètre de dispersion en plus : distribution Négative Binomiale, Beta Binomiale, ...

D'autres distributions sont adaptées à d'autres types de donnes comme les analyses de survie (temps écoulé jusqu'au décès) : (distribution Gamma, de Weibull, etc...)

Autres distributions pour les GLM

Les modèles multinomiaux sont une généralisation des modèles Binomiaux permettant de modéliser des variables qualitatives avec plus de 2 niveaux.
On peut souvent obtenir des résultats similaires avec des analyses de tables de contingence avec une distribution de Poisson

Il existe aussi la possibilité d'utiliser des "mixtures" de distributions.

Par exemple les "Zero Inflated Poisson models" (ZIP) utilisent une combinaison de distribution Binomiale et de Poisson pour modéliser des données de comptage avec un excès de 0 par rapport à la distribution de Poisson.

Autres fonctions de lien

Pour une même distribution, il existe parfois la possibilité d'utiliser d'autres fonctions de lien que la fonction par défaut ("canonic link").

Par exemple pour un GLM binomial les fonctions probit, cloglog, log et cauchit sont disponibles en plus de la fonction canonique logit.

La logit et la probit donnent en général des résultats très similaires mais la logit est plus pratique à utiliser et interpréter.
D'autres fonctions de lien peuvent parfois donner une meilleure estimation des données mais dans la grande majorité des cas, on se contente de la fonction canonique par défaut qui est généralement la plus adaptée.

Autres fonctions de lien

Les fonctions de lien sont en général choisies de façon à ce que leur inverse ne donne que des valeurs possibles pour une distribution donnée.

Par exemple la fonction exp ne peut donner que des valeurs positives pour la distribution de poisson (comptage) et l'inverse de la logit donne uniquement des valeurs comprises entre 0 et 1 pour la binomiale (proportions).

Les fonctions de lien correspondent aussi typiquement à la relation qu'on est susceptible d'observer avec certains types de données (pex : courbe sigmoïde pour des données binomiales)

Autres fonctions de lien

Liste des familles et des fonctions de lien disponibles pour la fonction glm() de R. Les fonction de lien par défaut sont indiquées parla lettre C.

Les familles quasi, quasibinomial et quasipoisson ne sont pas à proprement parler des distribution, mais des méthodes permettant d'estimer le degré de surdispersion dans certains modèles. Taper?family pour plus d'info.

family	identity	log	logit	probit	cloglog	inverse	1/mu^2	sqrt	cauchit
gaussian	C	x				x			
poisson	x	C						x	
binomial		x	C	x	x				x
Gamma	x	x				C			
inverse.gaussian	x	x				x	C		
quasi	C	x	x	x	x	x	x	x	
quasibinomial		x	C	x	x				x
quasipoisson	x	C						x	

Autres fonctions de lien

Si on veut savoir à quoi correspond une fonction de lien et son inverse, on peut taper par exemple :

```
> make.link("cloglog")
$linkfun
function (mu)
log(-log(1 - mu))
<environment: namespace:stats>

$linkinv
function (eta)
pmax(pmin(-expm1(-exp(eta)), 1 - .Machine$double.eps), .Machine$double.eps)
<environment: namespace:stats>
(...)
```

= -exp(-exp(eta))+1 sauf quand on est très proche de 0 ou 1

On peut utiliser les deux slots directement comme une fonction :

```
> make.link("cloglog")$linkfun(0.4)
[1] -0.671727
> make.link("cloglog")$linkinv(-0.671727)
[1] 0.4
> -exp(-exp(-0.671727))+1
[1] 0.4
```

GLM gaussien

On simule un jeu de données avec une distribution Normale (=Gaussienne)
Deux manières différentes mais strictement équivalentes de générer les y

```
n <- 10
beta0 <- 25
beta1 <- 1.5
beta2 <- -2
sigma <- 5

fertilizer <- rep (0:4, each=n)
set.seed(1)
mildew <- runif(n*5,0,4)
set.seed(2)

tomato <- beta0 + beta1*fertilizer + beta2*mildew + rnorm(n*5,0,sigma)

set.seed(2)
tomato2 <- rnorm(n = n*5,
                 mean = beta0 + beta1*fertilizer - beta2*mildew,
                 sd = sigma)

> all(tomato == tomato2)
[1] TRUE
```

$$y = \alpha + \beta * x + \epsilon$$

$$\epsilon \sim Normale(0, \sigma^2)$$

$$y \sim Normale(\alpha + \beta * x, \sigma^2)$$

GLM gaussien

```
> modlm <- lm(tomato ~ fertilizer + mildew)
> summary(modlm)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  25.3986     2.0847  12.183 3.77e-16 ***
fertilizer    0.9925     0.5705   1.740   0.0885 .
mildew       -1.5484     0.7484  -2.069   0.0441 *
---
Residual standard error: 5.7 on 47 degrees of freedom
Multiple R-squared:  0.1301,  Adjusted R-squared:  0.09304
F-statistic: 3.513 on 2 and 47 DF,  p-value: 0.03785
```

```
> modglm <- glm(tomato ~ fertilizer + mildew,
                                  family = gaussian)
> summary(modglm)

            Estimate Std. Error t value Pr(>|t|)
(Intercept)  25.3986     2.0847  12.183 3.77e-16 ***
fertilizer    0.9925     0.5705   1.740   0.0885 .
mildew       -1.5484     0.7484  -2.069   0.0441 *
---
(Dispersion parameter for gaussian family taken to be 32.49223)

    Null deviance: 1755.4  on 49  degrees of freedom
Residual deviance: 1527.1  on 47  degrees of freedom
AIC: 320.85

Number of Fisher Scoring iterations: 2
```

```
> summary(modlm)$sigma
[1] 5.700195
> summary(modglm)$dispersion
[1] 32.49223
> sqrt(summary(modglm)$dispersion)
[1] 5.700195
```

Plus de R² mais une "Null Deviance" et "Residual Deviance" (voir plus loin)

GLM de Poisson

On a suivi pendant 15 ans l'évolution d'une population sur un site en comptant le nombre d'individus dans 5 quadrats.
Typiquement ce genre de données de comptage suit une distribution de Poisson.

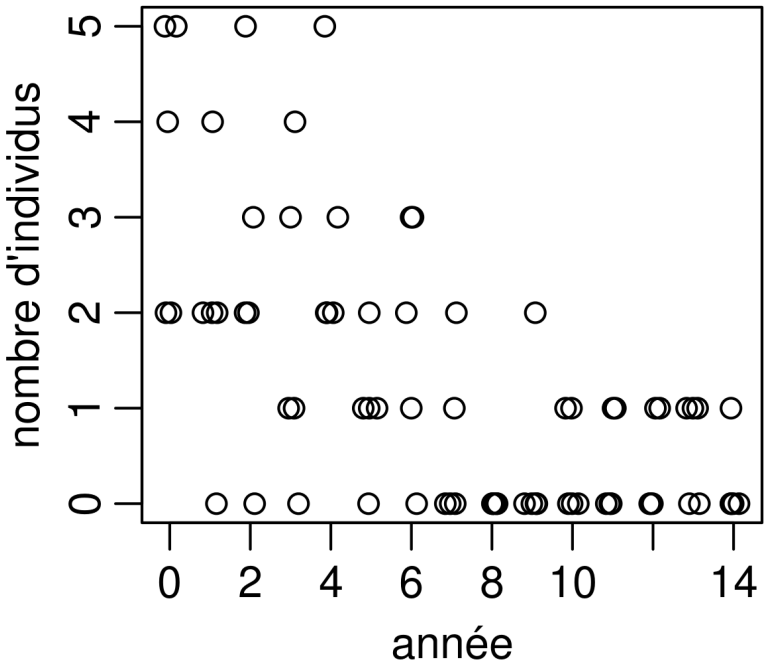
```
n <-5
x <- rep(0:14, each = n)
lambda <- 1 - 0.15 * x
```

$$\lambda = \alpha + \beta * x$$

```
set.seed(123)
y <- rpois(n*15, exp(lambda))
```

$$y \sim Poisson\left(g^{-1}(\lambda)\right) = Poisson\left(e^{\alpha+\beta*x}\right)$$

```
plot(y ~ jitter(x),
     ylab = "nombre d'individus",
     xlab = "année")
```



GLM de Poisson

Résultats et interprétation

```
> modglm <- glm(y ~ x, family = poisson)
> summary(modglm)

Call:
glm(formula = y ~ x, family = poisson)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.3668  -1.0030  -0.5046   0.7459   2.1631

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   1.2208     0.1527   7.993 1.32e-15 ***
x            -0.1908     0.0290  -6.578 4.77e-11 ***
---

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 131.832  on 74  degrees of freedom
Residual deviance:  79.884  on 73  degrees of freedom
AIC: 193.15

Number of Fisher Scoring iterations: 5
```

log link par défaut

L'année 0, il y avait en moyenne exp(1.2208) = 3.39 individus par quadrat.

Un an plus tard, il y avait exp(1.2208 - 0.1908*1) = 2.80 individus par quadrat.

10 ans plus tard : exp(1.2208 - 0.1908*10) = 0.50 individus par quadrat

soit une diminution de (0.47 - 3.39)*100/3.39 = - 86.14 %

GLM de Poisson

Résultats et interprétation

La relation n'est donc pas linéaire au cours du temps mais peut s'interpréter de manière multiplicative

```
> modglm <- glm(y ~ x, family = poisson)
> summary(modglm)
             Estimate Std. Error z value Pr(>|z|)
(Intercept)    1.2208     0.1527   7.993 1.32e-15 ***
x             -0.1908     0.0290  -6.578 4.77e-11 ***
```

Un an plus tard, il y avait
exp(1.2208 - 0.198*1)
= exp(1.2208) * exp(-0.198)

Autrement dit, en un an, le nombre moyen est multiplié par exp(-0.198) = 0.82 soit une diminution de 18 %
par rapport à l'année précédente

Et en effet :
(2.78 - 3.39)*100/3.39 = -17.994 %

Et au bout de 10 ans :

```
> 1-exp(-0.198*10)
[1] 0.8619308
```

```
> slopes <- round(seq(-0.4,0.4,0.05),2)
> data.frame(
+     slopes = slopes,
+     expslopes = round(exp(slopes),2),
+     pct = 100*(round(exp(slopes),2)-1)
+     )
   slopes expslopes pct
1   -0.40      0.67 -33
2   -0.35      0.70 -30
3   -0.30      0.74 -26
4   -0.25      0.78 -22
5   -0.20      0.82 -18
6   -0.15      0.86 -14
7   -0.10      0.90 -10
8   -0.05      0.95  -5
9    0.00      1.00   0
10   0.05      1.05   5
11   0.10      1.11  11
12   0.15      1.16  16
13   0.20      1.22  22
14   0.25      1.28  28
15   0.30      1.35  35
16   0.35      1.42  42
17   0.40      1.49  49
```

```
> Intercepts <- 0:7
> data.frame(Intercepts =
                  Intercepts,
+            expIntercepts =
                  exp(Intercepts))
  Intercepts expIntercepts
1          0      1.000000
2          1      2.718282
3          2      7.389056
4          3     20.085537
5          4     54.598150
6          5    148.413159
7          6    403.428793
8          7   1096.633158
```

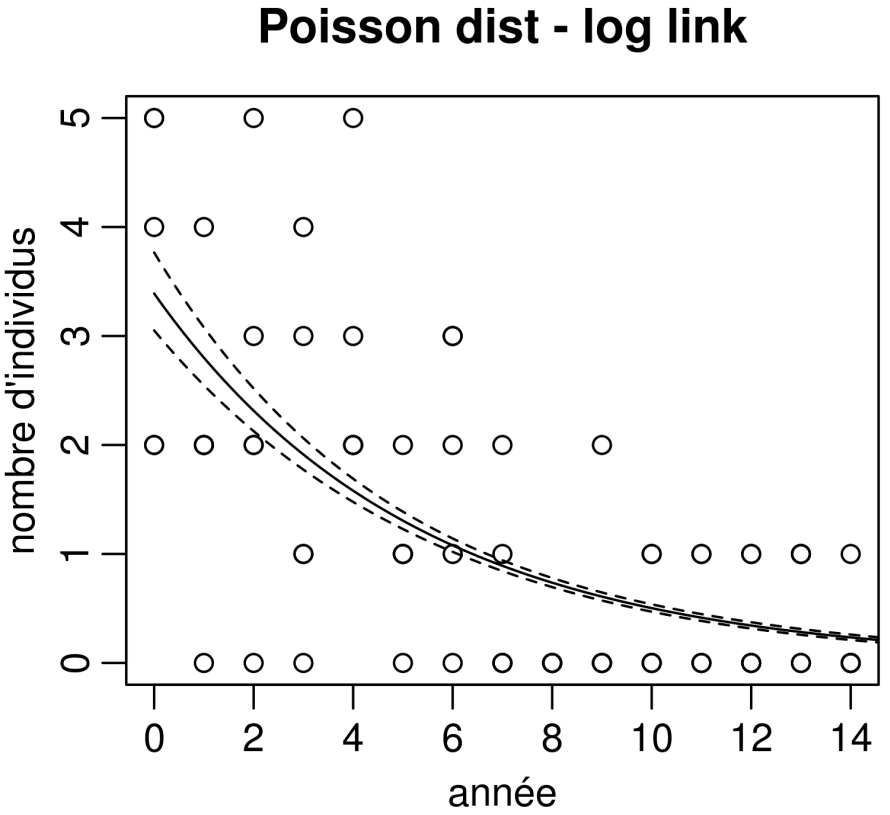
GLM de Poisson

Représentation graphique

Les prédictions et erreurs standards sont sur l'échelle log. Il faut les rétro-transformer pour obtenir les valeurs sur l'échelle d'origine.
Attention : Il ne faut PAS rétro-transformer les erreurs standard elles-mêmes mais bien les bornes : valeurs prédites +- se

NB : la fonction predict() transforme les erreurs standards et ne donnera pas les mêmes résultats...

```
X <- cbind(1, seq(0, 15, 0.1))
pred <- X %*% coef(modglm)
se <- sqrt(diag(X %*% vcov(mod) %*% t(X)))
lwr <- exp(pred - se)
upr <- exp(pred + se)
pred <- exp(pred)

plot(y ~x, ylab = "nombre d'individus",
     xlab = "année",
     main = "Poisson dist - log link")
lines(pred, x = seq(0, 15, 0.1))
lines(lwr, x = seq(0, 15, 0.1), lty = 2)
lines(upr, x = seq(0, 15, 0.1), lty = 2)
```



(log) Likelihood - Deviance - AIC

Résultats et interprétation

```
> modglm <- glm(y ~ x, family = poisson)
> summary(modglm)

Call:
glm(formula = y ~ x, family = poisson)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.3668  -1.0030  -0.5046   0.7459   2.1631

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   1.2208     0.1527   7.993 1.32e-15 ***
x            -0.1908     0.0290  -6.578 4.77e-11 ***
---

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 131.832  on 74  degrees of freedom
Residual deviance:  79.884  on 73  degrees of freedom
AIC: 193.15

Number of Fisher Scoring iterations: 5
```

(log) Likelihood - Deviance - AIC

Log Vraisemblance (Log Likelihood)

La vraisemblance est la probabilité d'obtenir les données observées (les y) sachant un modèle (modèle linéaire avec distribution de Poisson et fonction de lien log dans notre cas) et ses paramètres (l'intercept et la pente dans notre cas).
Cette probabilité est données par la fonction de densité de probabilité de la distribution choisie (dpois dans notre cas).

$$L = p(y|\beta, X) = \prod_{i=1}^{n} Poisson(y_i|e^{X_i\beta})$$

y sachant (|) beta et X

Produit de la probabilité pour chaque y observé

En pratique, l'algorithme utilisé par glm() cherche de manière itérative les paramètres beta pour lesquels la vraisemblance est maximisée.

(log) Likelihood - Deviance - AIC

Log Vraisemblance (Log Likelihood)

On travaille presque toujours avec le log de la vraisemblance pour des raisons pratiques ea parce que la vraisemblance est souvent une valeur très petite et parce que :

$$\log(a/b) = \log(a) - \log(b)$$
$$\log(a*b) = \log(a) + \log(b)$$

La log-vraisemblance est toujours plus petite que 0 (car la vraisemblance est toujours comprise entre 0 et 1) et plus elle s'approche de 0, mieux le modèle prédit notre jeu de données

La (log) vraisemblance est difficile à interpréter directement mais elle va servir de base pour comparer des modèles entre eux

(log) Likelihood - Deviance - AIC

Log Vraisemblance (Log Likelihood)

En pratique dans R, on peut extraire la log-vraisemblance avec `logLik()` et recalculer cette log-vraisemblance avec les fonctions de densité des différentes lois :

```
> modglm <- glm(y ~ x, family = poisson)

> (Beta <- coef(modglm))
(Intercept)           x
  1.2207514  -0.1907899
> X <- cbind(1, x)

> logLik(modglm)
'log Lik.' -94.57749 (df=2)

> log(prod(dpois(x = y, lambda = exp(X %*% Beta))))
[1] -94.57749
> sum(log(dpois(x = y, lambda = exp(X %*% Beta))))
[1] -94.57749
```

Note :
dpois est la fonction de densité de probabilité, elle donne la probabilité d'un quantile pour une loi de Poisson
ppois est la fonction de probabilité cumulée, elle donne la surface sous la fonction de Poisson jusqu'à un quantile donné

$$L = p(y|\beta, X) = \prod_{i=1}^{n} Poisson(y_i|e^{X_i\beta})$$

(log) Likelihood - Deviance - AIC

Deviance

La déviance est une statistique qui compare la vraisemblance du modèle d'intérêt avec la vraisemblance d'un modèle saturé c'est à dire comprenant autant de variables explicatives que de données et prédisant donc parfaitement les données
(comme lorsque de le $R^2 = 1$)

$$Deviance = 2\log\frac{L_{modèle\,saturé}}{L_{modèle}} = 2*(\log(L_{modèle\,saturé}) - \log(L_{modèle}))$$

La déviance est donc une statistique de qualité d'ajustement du modèle analogue à la somme des carrés des résidus dans les modèles linéaires classiques.

(log) Likelihood - Deviance - AIC

Deviance

Plus la déviance est petite (proche de 0) mieux le modèle prédit les données (mais sans prendre en compte le risque de sur-ajustement des données - overfitting).

Lorsqu'on ajoute une variable explicative sans aucun lien avec la variable dépendante (bruit aléatoire) la déviance diminuera en moyenne de 1 unité.

Si la variable est informative, la déviance diminuera de plus d'une unité en moyenne.

(log) Likelihood - Deviance - AIC

Null Deviance - Residual Deviance

Le summary du modèle donne deux déviances :
la "Residual Deviance" correspond à la déviance du modèle tel que défini précédemment.
(attention à ne pas confondre avec les "deviance residuals"...)

La "Null deviance" correspond à la déviance d'un modèle sans variables explicatives (juste un intercept le cas échéant).

Si le modèle est très mauvais, la différence entre la Residual Deviance et la null deviance est à peu près égale à la différence de nombre de paramètres entre les deux (données par la différence de degrés de liberté)

(log) Likelihood - Deviance - AIC

Null Deviance - Residual Deviance

```
> modglm <- glm(y ~ x, family = poisson)
> summary(modglm)

Call:
glm(formula = y ~ x, family = poisson)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.3668  -1.0030  -0.5046   0.7459   2.1631

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   1.2208     0.1527   7.993 1.32e-15 ***
x            -0.1908     0.0290  -6.578 4.77e-11 ***
---

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 131.832  on 74  degrees of freedom
Residual deviance:  79.884  on 73  degrees of freedom
AIC: 193.15

Number of Fisher Scoring iterations: 5
```

(log) Likelihood - Deviance - AIC

Test de Rapport de vraissemblance

Likelihood Ratio Test = LRT

La statistique utilisée pour calculer la déviance peut être utilisée pour comparer n'importe quels modèles emboîtés et suit asymptotiquement une distribution Chi carré avec pour degrés de liberté la différence de nombre de paramètres entre les deux modèles.

$$LRT = 2*(\log(L_{modèle\,1}) - \log(L_{modèle\,2})) = Deviance_{modèle\,1} - Deviance_{modèle\,2}$$

(log) Likelihood - Deviance - AIC

Test de Rapport de vraisemblance

$$LRT = 2*(\log(L_{modèle\,1}) - \log(L_{modèle\,2})) = Deviance_{modèle\,1} - Deviance_{modèle\,2}$$

```
> modglm <- glm(y ~ x, family = poisson)
> nullmod <- glm(y ~ 1, family = poisson)

> # anova(nullmod, modglm, test = "Chisq")
> anova(nullmod, modglm, test = "LRT") # equivalent
Analysis of Deviance Table

Model 1: y ~ 1
Model 2: y ~ x
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1        74    131.832
2        73     79.884  1   51.948  5.7e-13 ***


> (dfnullmod <- length(coef(nullmod)))
[1] 1
> (dfmodglm <- length(coef(modglm)))
[1] 2
> LR <- (deviance(modglm) - deviance(nullmod))

> 1-pchisq(q = abs(LR), df = abs(dfmodglm - dfnullmod))
[1] 5.699885e-13
```

(log) Likelihood - Deviance - AIC

AIC = Aikaike Information Criterion

NB : On reverra plus loin en détail l'utilisation et la philosophie derrière les AIC

La déviance ne tient pas compte du problème de sur-ajustement des données (overfitting) quand on a trop de variables explicatives/paramètres à estimer par rapport au nombre de données.

L'AIC est un critère d'information basé sur la vraisemblance qui pénalise les modèles avec trop de paramètres à estimer.
Il existe aussi une version corrigée pour les petits échantillons : l'AICc

$$AIC = -2 \log(L) + 2k$$

$$AICc = AIC + \frac{2k(k+1)}{n-k-1}$$

Avec :
k = nombre de paramètres
n = nombre de données
L = vraissemblance du modèle

NB : On devrait toujours utiliser l'AICc dont la valeur tend vers celle de l'AIC quand n est grand

(log) Likelihood - Deviance - AIC

AIC = Aikaike Information Criterion

L'AIC seul est inutile. Il s'agit d'une mesure relative

Pour un jeu de données et un ensemble de modèles pour ce jeu de données, ceux qui ont un AIC plus petits sont "meilleurs que les autres" (voir + loin pour + de détails)

Les AIC permettent de comparer également des modèles non emboîtés (ie dont les paramètres ne sont pas un sous-ensemble des paramètres de l'autre modèle)

ATTENTION :
Les AIC (comme la déviance) ne peuvent comparer que des modèles basés exactement sur le même jeu de données du côté des variables dépendantes (Y).
On ne peut donc pas comparer des modèles avec des transformations différentes des Y ou avec des nombres de données différentes (attentions aux NA dans les x!)

(log) Likelihood - Deviance - AIC

Pseudo R^2

Le R^2 des modèles linéaires est basé sur la somme des carrés des résidus. Il n'y a donc pas à proprement parler de R^2 dans les GLM qui sont basés sur le Maximum de Vraisemblance

De nombreux équivalents du R^2 ont été inventés pour les GLM. Ils donnent souvent des valeurs très différentes et sont parfois difficiles à interpréter.

Prudence donc quand on vous présente un R^2 pour un GLM. Il faut savoir comment il a été calculé.

(log) Likelihood - Deviance - AIC

Pseudo R²

Un Pseudo-R² qui nous semble parfois utile et d'interprétation à peu près identique au R² classique est :
le R² d'une régression entre les valeurs prédites et les valeurs observées.

récupère la variable dépendante dans les modèles mixtes

```
pseudoRsq <- function(mod, dec=3) {

    {if (class(mod)[1] %in% "mer") y <- mod@frame[,1] else y <- mod$model[,1] }

    pseudoR <- round(summary(lm(y ~ fitted(mod)))$r.squared, dec)
    return(pseudoR)
}
```

Ce Pseudo-R² a les mêmes défauts que le R² classique.
(Les deux valeurs sont identiques pour des modèles linéaires)

De plus il ne représente pas toujours bien la qualité des modèles, en particulier pour des données binaires où le pseudo R² est presque toujours petit sans que le modèle soit spécialement mauvais.

GLM de Poisson

Inférence

```
> modglm <- glm(y ~ x, family = poisson)
> summary(modglm)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   1.2208     0.1527   7.993 1.32e-15 ***
x            -0.1908     0.0290  -6.578 4.77e-11 ***
```

Test de Wald : rapport entre un paramètre et son erreur standard : suit asymptotiquement une loi normale.

Paramètre +- 1.96 * se donne l'intervalle de confiance à 95 %

```
> drop1(modglm, test = "Chisq")

       Df Deviance    AIC    LRT Pr(>Chi)
<none>      79.884 193.16
x       1  131.832 243.10 51.948  5.7e-13 ***
```

Test de rapport de vraisemblance : équivalent au test F de l'Anova. Suit asymptotiquement une distribution Chi carré.

L'approximation est normalement meilleure que dans le test de wald

Le test de Wald et les comparaisons de modèles emboîtés ne donnent plus les même résultats contrairement aux modèles linéaires classiques

On peut aussi utiliser la fonction anova pour comparer deux modèles emboîtés

GLM de Poisson

Inférence

NB : toutes les inférences paramétriques pour les GLM sont donc approximatives en particulier quand on a peu de données.

C'est une tendance générale : plus on va vers des modèles complexes (modèles mixtes, GAM,...) moins les inférences sont fondées sur des bases mathématiques solides.

Il faut se rappeler qu'il ne 'agit que d'un outil d'aide à la décision (est-ce que j'ai suffisamment de données pour estimer un paramètre?)

"*All models are false but some models are useful*" (G. Box)

"*It is better to be able to say something approximate about the right model rather than something very precise about the wrong model*" (S.Wood)

GLM de Poisson

Différentes fonction de lien + modèles linéaires

```
> modglm <- glm(y ~ x, family = poisson)
> modglm2 <- glm(y ~ x ,
              family = poisson(link = "sqrt"))

> modglm3 <- glm(y ~ x ,
              family = poisson(link = "identity"))
Erreur : impossible de trouver un jeu de coefficients
correct : prière de fournir des valeurs initiales
De plus : Message d'avis :
In log(y/mu) : production de NaN
> modglm3 <- glm(y ~ x ,
     family = poisson(link = "identity"), start=c(3,-0.2))

> modlm <- glm(y ~ x)
> modlm2 <- glm(log(y+1) ~ x)
```

Modèle de Poisson avec une fonction de lien racine carrée

Le lien identité n'est à priori pas très adapté ea car il peut prédire des valeurs négatives.
Il faut ici lui fournir des valeurs de départ sans quoi l'algorithme n'arrive pas à estimer les paramètres

Modèles linéaires classiques
Une approche fréquente pour ce genre de données (mais pas forcément conseillée) est de transformer les y avec un log. Comme il y a des 0 on est obligé d'utiliser log(y+1)

GLM de Poisson

Différentes fonction de lien + modèles linéaires

```
> deviance(modglm) ; deviance(modglm2) ; deviance(modglm3) ; deviance(modlm)  ;
deviance(modlm2)

[1] 79.88419
[1] 82.33936
[1] 87.91184
[1] 93.07524
[1] 16.719
```

ATTENTION ! On ne peut pas comparer la déviance ou les AIC du dernier modèle car les Y ne sont pas les mêmes (transformation log)

```
> AIC(modglm) ; AIC(modglm2) ; AIC(modglm3) ; AIC(modlm) ; AIC(modlm2)

[1] 193.155
[1] 195.6101
[1] 201.1826
[1] 235.0348
[1] 106.2701
```

```
> pseudoRsq(modglm) ; pseudoRsq(modglm2) ; pseudoRsq(modglm3) ; pseudoRsq(modlm) ;
pseudoRsq(modlm2)

[1] 0.433
[1] 0.428
[1] 0.392
[1] 0.392
[1] 0.38
```

On ne peut pas non plus directement comparer le pseudo-R^2 tel quel car pour le dernier modèle, les valeurs prédites le sont sur l'échelle log alors que pour tous les autres on travaille sur l'échelle d'origine. Mais on peut calculer le pseudo-R^2 après rétro-transformation qui lui est comparable :

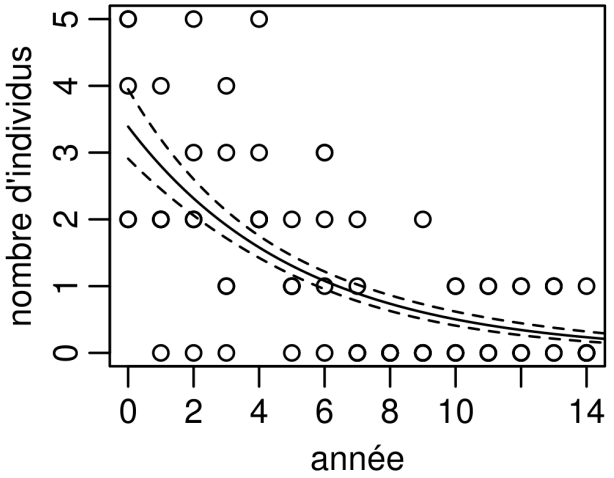
```
> summary(lm(I(exp(modlm2$model[,1])-1) ~ I(exp(fitted(mod))-1)))$r.squared
[1] 0.4120851
```

GLM de Poisson

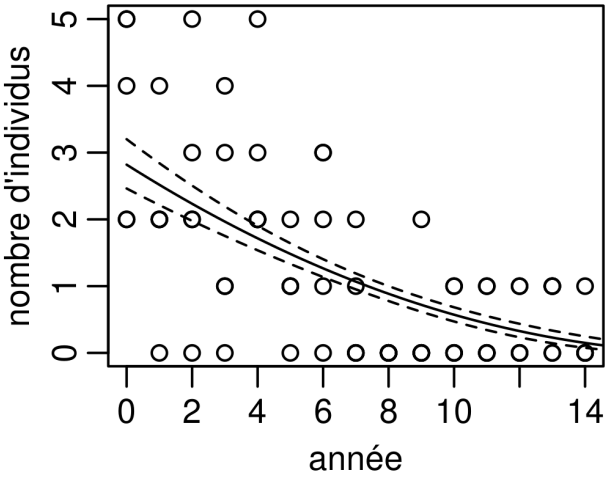
Différentes fonction de lien + modèles linéaires

Représentation graphique des 4 premiers modèles

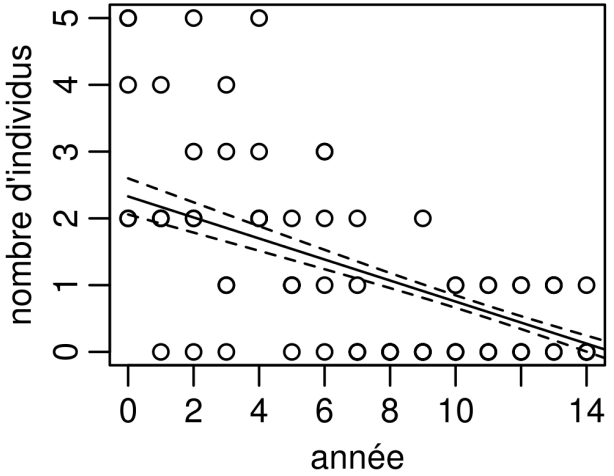
Poisson dist - log link



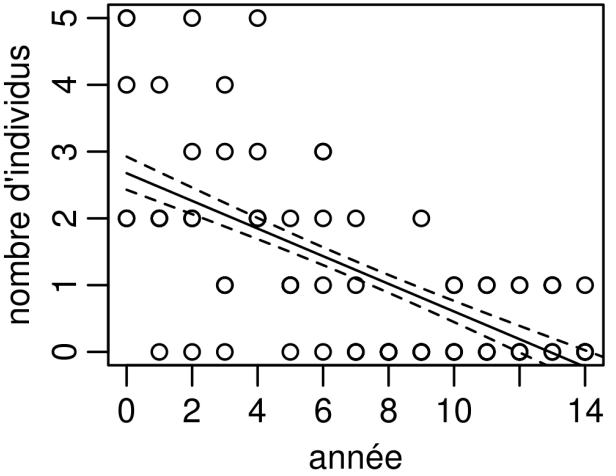
Poisson dist - sqrt link



Poisson dist - identity link



Normal dist



Code pour les graphiques de la dia précédente

```
par(mfrow = c(2,2), mar = c(3,3,3,1),
     mgp = c(1.75, 0.6, 0))

modglm <- glm(y ~ x , family = poisson)
X <- cbind(1, seq(0, 15, 0.1))
pred <- X %*% coef(modglm)
se <- sqrt(diag(X %*% vcov(modglm) %*% t(X)))
lwr <- exp(pred - se)
upr <- exp(pred + se)
pred <- exp(pred)

plot(y ~x, ylab = "nombre d'individus",
     xlab = "année",
     main = "Poisson dist - log link")
lines(pred, x = seq(0, 15, 0.1))
lines(lwr, x = seq(0, 15, 0.1), lty = 2)
lines(upr, x = seq(0, 15, 0.1), lty = 2)


modglm2 <- glm(y ~ x ,
     family = poisson(link = "sqrt"))
pred <- X %*% coef(modglm2)
se <- sqrt(diag(X %*% vcov(modglm2) %*% t(X)))
lwr <- (pred - se)^2
upr <- (pred + se)^2
pred <- (pred)^2

plot(y ~x, ylab = "nombre d'individus",
     xlab = "année",
     main = "Poisson dist - sqrt link")
lines(pred, x = seq(0, 15, 0.1))
lines(lwr, x = seq(0, 15, 0.1), lty = 2)
lines(upr, x = seq(0, 15, 0.1), lty = 2)
```

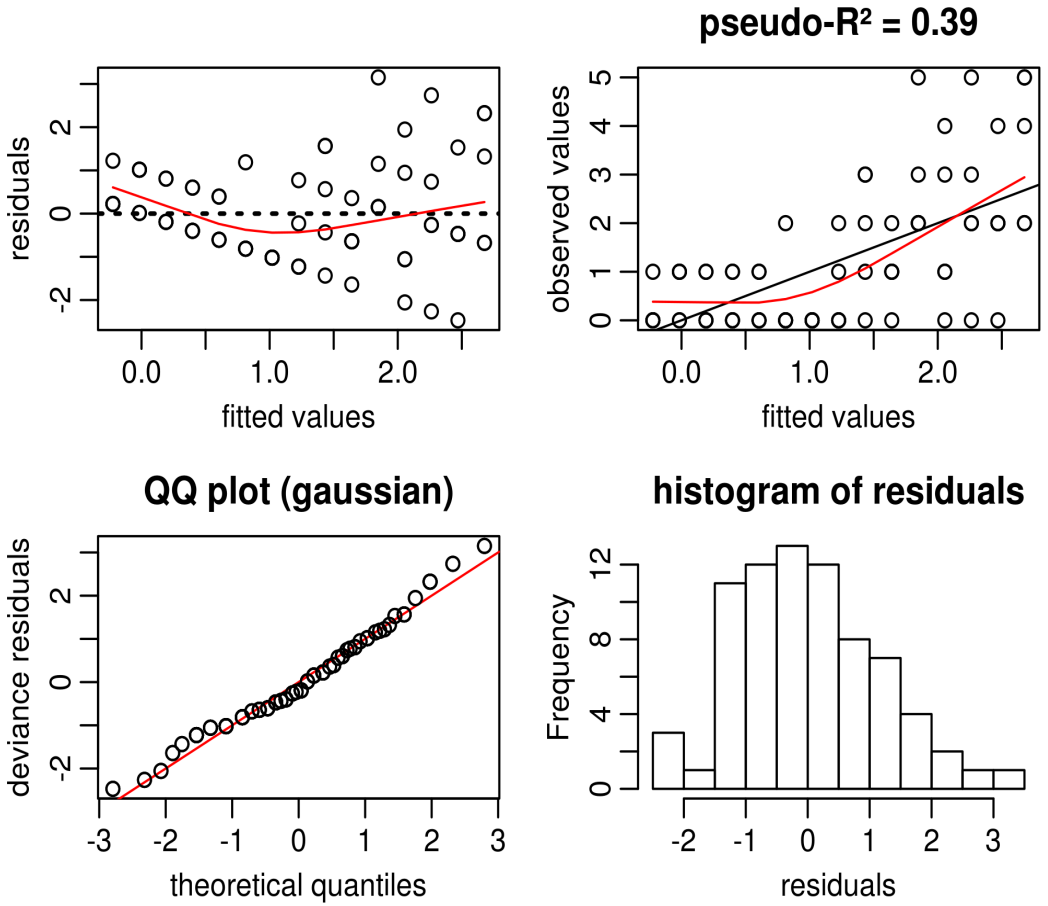
```
modglm3 <- glm(y ~ x ,
     family = poisson(link = "identity"),
     start=c(3,-0.2))
pred <- X %*% coef(modglm3)
se <- sqrt(diag(X %*% vcov(mod) %*% t(X)))
lwr <- (pred - se)
upr <- (pred + se)
pred <- (pred)

plot(y ~x, ylab = "nombre d'individus",
     xlab = "année",
     main = "Poisson dist - identity link")
lines(pred, x = seq(0, 15, 0.1))
lines(lwr, x = seq(0, 15, 0.1), lty = 2)
lines(upr, x = seq(0, 15, 0.1), lty = 2)


mod <- lm(y ~ x)
X <- cbind(1, seq(0, 15, 0.1))
predlm <- X %*% coef(mod)
selm <- sqrt(diag(X %*% vcov(mod) %*% t(X)))
lwrlm <- (predlm - selm)
uprlm <- (predlm + selm)
predlm <- (predlm)

plot(y ~x, ylab = "nombre d'individus",
     xlab = "année",
     main = "Normal dist")
lines(predlm, x = seq(0, 15, 0.1))
lines(lwrlm, x = seq(0, 15, 0.1), lty = 2)
lines(uprlm, x = seq(0, 15, 0.1), lty = 2)
```

GLM de Poisson

Différentes fonction de lien + modèles linéaires

Plots des résidus pour le modèle gaussien

```
> modlm <- glm(y ~ x)
> diagplot(modlm)
```



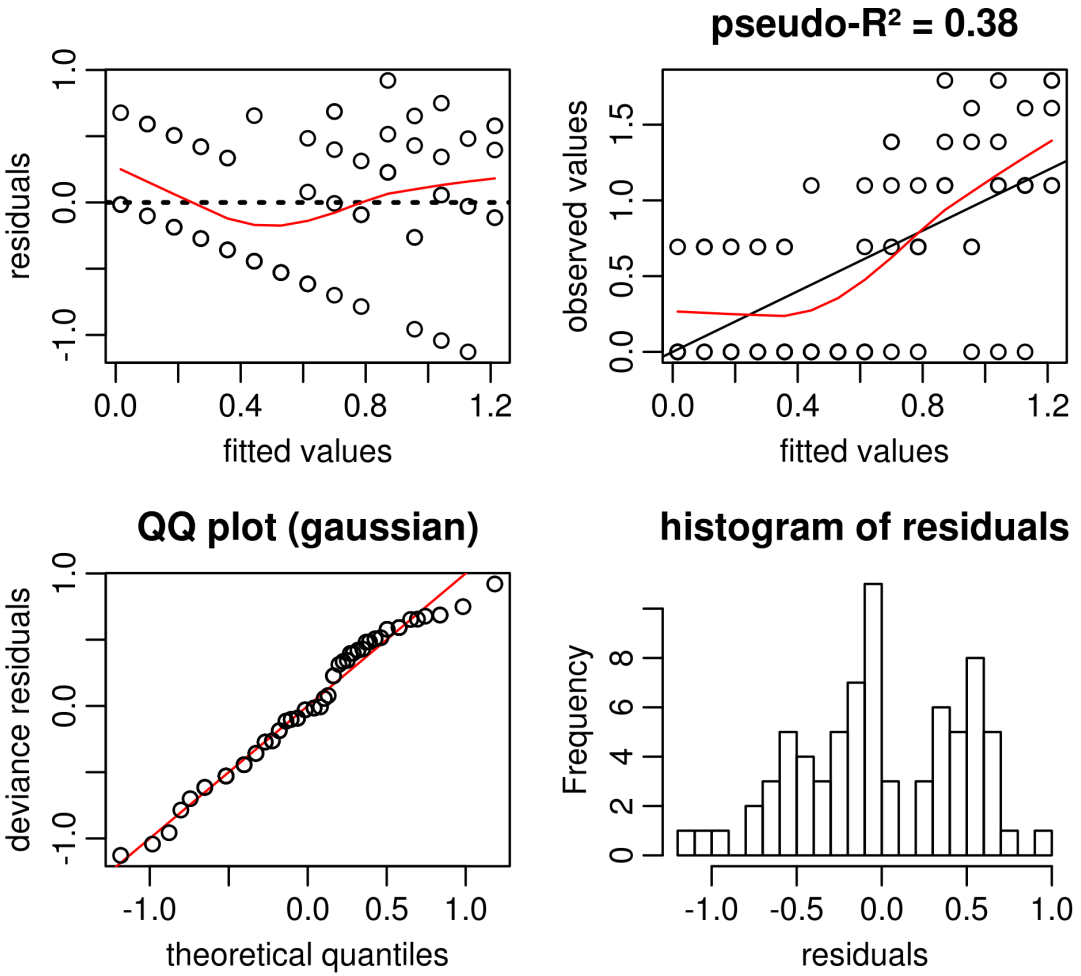
L'écart à la normalité des résidus n'est pas flagrant.
Mais on voit bien que la variance n'est pas du tout homogène

GLM de Poisson

Différentes fonction de lien + modèles linéaires

Plots des résidus pour le modèle gaussien transformé

```
> modlm2 <- glm(log(y+1) ~ x)
> diagplot(modlm2)
```



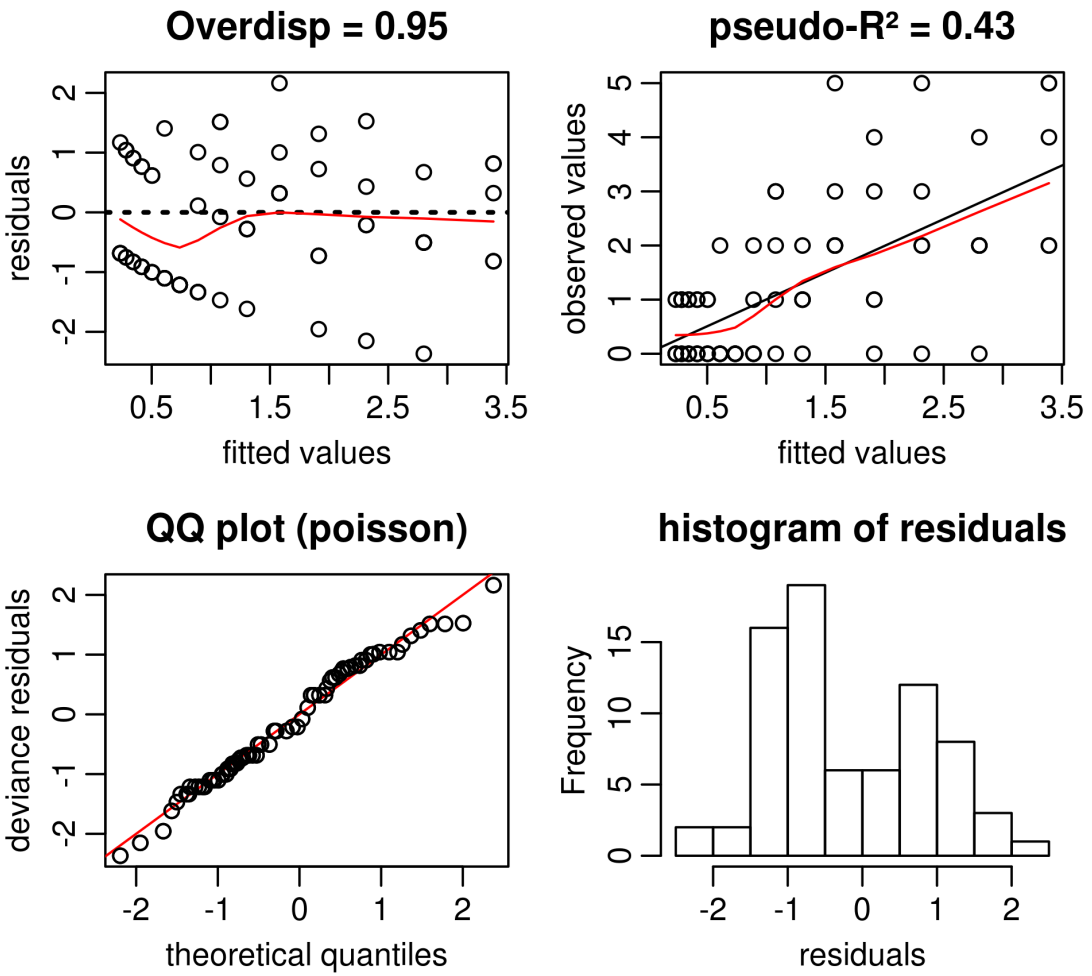
C'est déjà mieux mais ce n'est pas parfait (non linéarité?)

GLM de Poisson

Différentes fonction de lien + modèles linéaires

Plots des résidus pour le modèle de Poisson standard

```
> modglm <- glm(y ~ x, family = poisson)
> diagplot(modglm)
```



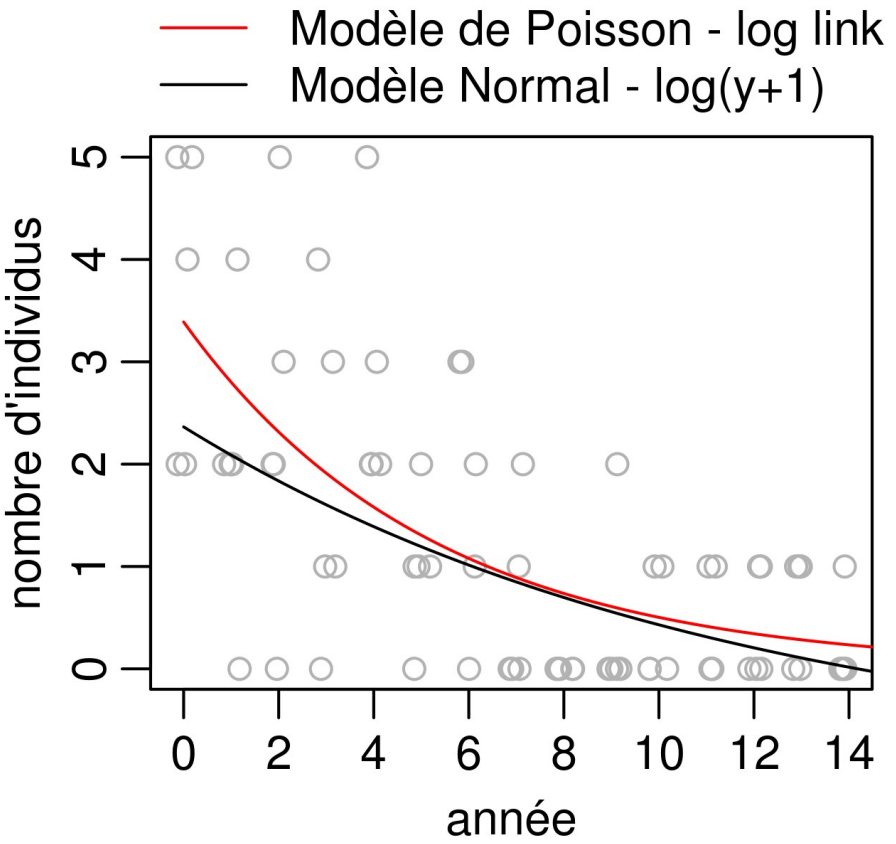
NB : les résidus ne sont pas exactement les mêmes.
Ce sont des résidus standardisés (cfr plus loin)

GLM de Poisson

Différentes fonction de lien + modèles linéaires

Comparaison modèle de Poisson vs modèle gaussien avec transformation des y en log(y+1)

```
mod <- lm(log(y+1) ~ x)
X <- cbind(1, seq(0, 15, 0.1))
predlm <- X %*% coef(mod)
selm <- sqrt(diag(X %*% vcov(mod) %*% t(X)))
lwrlm <- exp(predlm - selm)-1
uprlm <- exp(predlm + selm)-1
predlm <- exp(predlm)-1

modglm <- glm(y ~ x , family = poisson)
pred <- X %*% coef(modglm)
se <- sqrt(diag(X %*% vcov(mod) %*% t(X)))
lwr <- exp(pred - se)
upr <- exp(pred + se)
pred <- exp(pred)

dev.new(8.5/2.54, 8/2.54)
par(mfrow = c(1,1), mar = c(3,3,3,1),
     mgp = c(1.75, 0.6, 0))
set.seed(2)
plot(y ~ jitter(x), ylab = "nombre d'individus",
     xlab = "année", col = "grey70")
lines(pred, x = seq(0, 15, 0.1), col = "red")
lines(predlm, x = seq(0, 15, 0.1))
```




```
legend(x = "top", inset = -0.3, xpd = NA, bty = "n", lty = 1, col = c("red", "black") ,
       legend = c("Modèle de Poisson - log link", "Modèle Normal - log(y+1)"))
```

GLM binomial

Dans les GLM binomiaux, la variable dépendante est un nombre de succès sur un nombre d'essais (N) qui s'exprime souvent sous forme d'une proportion (ou%)
La variable dépendante peut aussi être sous forme binaire, cas particulier où N = 1

Exemple classique d'écotoxicologie :
On a exposé des mâles et des femelles d'une espèce d'insecte à 6 doses croissantes d'un pesticide. A chaque dose on a testé 20 mâles et 20 femelles (N = 20) et on a regardé combien étaient morts après 2 jours (= nombre de "succès")

Attention : toutes les données en % ou en proportion n'ont pas une distribution binomiale.
Pex : % de surface d'une feuille attaquée par un champignon
(= rapport de 2 variables quantitatives continues).
Dans ce cas on fait typiquement un modèle gaussien, on examine les résidus et si la distribution n'est pas gaussienne, on fait des transformations de variables pex : `asin(sqrt(y))`, `logit(y+n)`,... ou on utilise la donnée brute (sans ratio, pex surface de champignon) et un offset (pex surface de la feuille) comme moyen de standardisation.

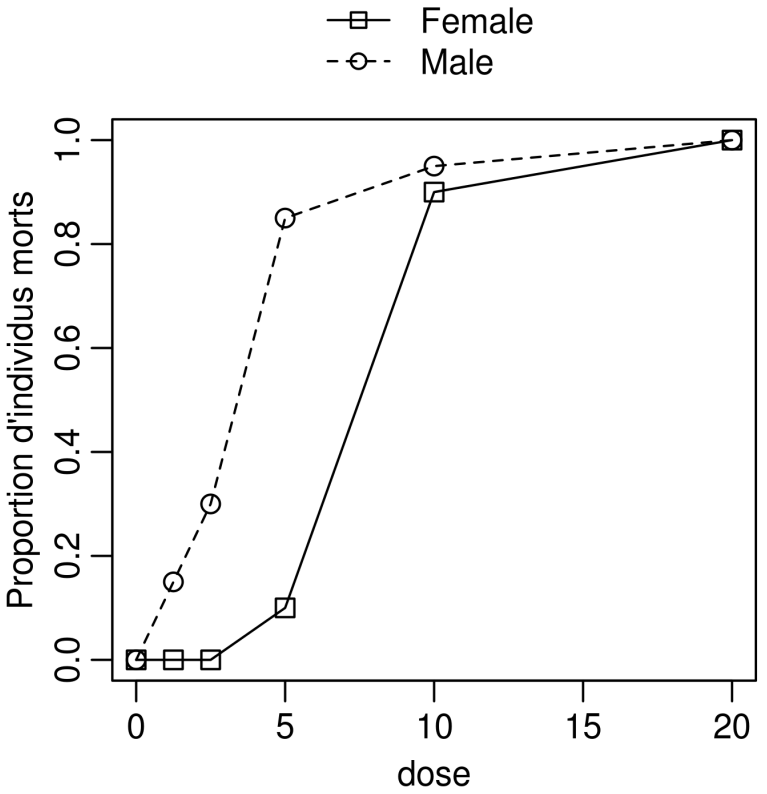
voir pex discussion ici : Warton, D.I., Hui, F.K., 2011. The arcsine is asinine: the analysis of proportions in ecology. Ecology 92, 3–10.

GLM binomial

Génération du jeu de données

```
nb = 20
d <- data.frame(
    sex = rep(c("M", "F"), each = 6),
    dose = rep(c(0, 1.25, 2.5, 5, 10, 20), 2),
    n = nb)

X <- cbind(1, d$sex, d$dose)
Beta <- c(-9, 3, 0.8)
linpred <- X %*% Beta
invlogit <- function(x) {return(exp(x)/(1+exp(x)))}
prop <- invlogit(linpred)

set.seed(2)
d$dead <- rbinom(12, p = prop, size = nb)
d$prop <- d$dead/d$n
```

```
> d
   sex  dose  n dead prop
1    M  0.00 20    0 0.00
2    M  1.25 20    3 0.15
3    M  2.50 20    6 0.30
4    M  5.00 20   17 0.85
5    M 10.00 20   19 0.95
6    M 20.00 20   20 1.00
7    F  0.00 20    0 0.00
8    F  1.25 20    0 0.00
9    F  2.50 20    0 0.00
10   F  5.00 20    2 0.10
11   F 10.00 20   18 0.90
12   F 20.00 20   20 1.00
```




```
dev.new(9/2.54, 9/2.54)
par(mar = c(3,3,3,1), mgp = c(1.75, 0.6, 0), cex = 0.9)
plot(prop ~ dose, data=d, pch = c(0,1)[as.numeric(d$sex)],
     cex = 1.2, ylab = "Proportion d'individus morts")
lines(prop ~ dose, data=d[d$sex == "F",], lty = 1)
lines(prop ~ dose, data=d[d$sex == "M",], lty = 2)
legend(x = "top", inset = -0.25, xpd = NA, bty = "n",
       lty = 1:2, pch = 0:1 ,
       legend = c("Female", "Male"))
```

GLM binomial

Modèle

Deux syntaxes possibles donnant des résultats identiques:

1) y = proportion de morts, et N dans l'argument "weights"

2) y = matrice à deux colonnes avec nombre de vivants et nombre de morts

```
> mod <- glm(prop ~ sex + dose, weights = n, data=d, family = binomial)
> mod2 <- glm(cbind(dead, n-dead) ~ sex + dose, data=d, family = binomial)
> summary(mod)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -6.6042     1.0208  -6.469 9.83e-11 ***
sexM          3.5689     0.7931   4.500 6.81e-06 ***
dose          0.8682     0.1236   7.025 2.15e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 236.7523  on 11  degrees of freedom
Residual deviance:   6.7239  on  9  degrees of freedom
AIC: 28.658

Number of Fisher Scoring iterations: 6
```

GLM binomial

Modèle

On peut vérifier qu'il n'y a pas d'interaction

```
> mod3 <- glm(prop ~ sex * dose, weights = n, data=d, family = binomial)
> anova(mod, mod3, test = "Chisq")
Analysis of Deviance Table

Model 1: prop ~ sex + dose
Model 2: prop ~ sex * dose
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1         9     6.7239
2         8     6.4366  1  0.28732   0.5919
```

GLM binomial

Interprétation (1)

Rappel : par défaut on travaille sur l'échelle logit
logit = log(p/(1-p) ; invlogit = exp(p) / (1+exp(p))

```
invlogit <- function(x) {return(exp(x)/(1+exp(x)))}
> mod <- glm(prop ~ sex + dose, weights = n, data=d, family = binomial)
> summary(mod)

            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -6.6042     1.0208  -6.469 9.83e-11 ***
sexM          3.5689     0.7931   4.500 6.81e-06 ***
dose          0.8682     0.1236   7.025 2.15e-12 ***
```

La proportion estimée de morts pour les femelles à une dose de 0 est de invlogit(-6.6042) = 0.0013
Pour les mâles cette proportion est de invlogit(-6.6042+3.5689) = 0.046
Lorsque la dose passe de 0 à 1, la proportion de mortalité passe de 0.0013 à invlogit(-6.6042 + 0.8682*1) = 0.0032 pour les femelles et de 0.046 à invlogit(-6.6042 + 3.5689+ 0.8682*1) = 0.10 pour les mâles.

GLM binomial

Interprétation (1)

NB : p/(1-p) représente un "odds ratio"
C'est la probabilité qu'un événement se produise / la probabilité qu'il ne se produise pas.
Lorsque ce ratio est > 1 il est plus probable que l'événement se produise
Lorsque ce ratio est <1, il est plus probable qu'il ne se produise pas

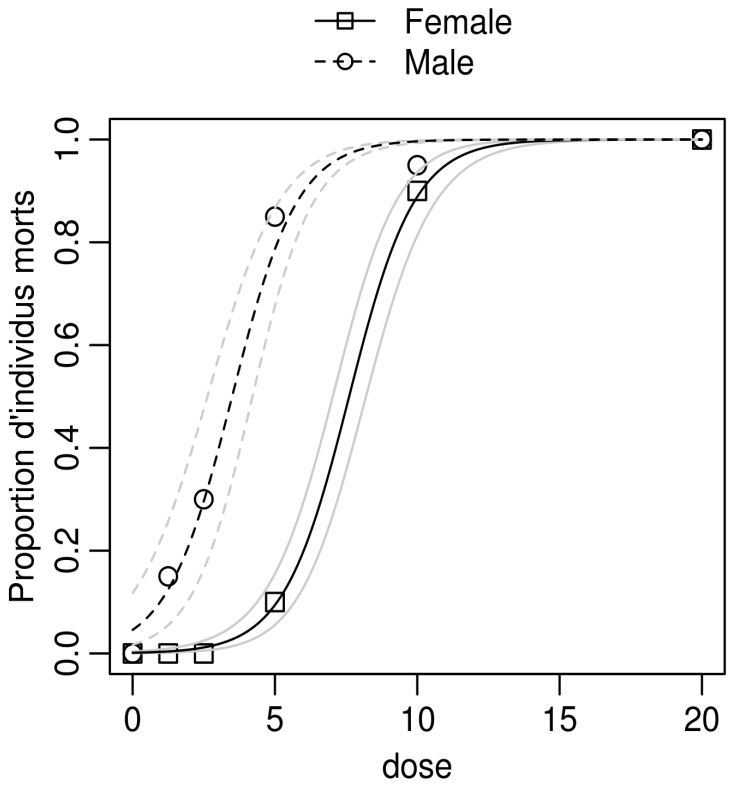
Sur l'échelle log, on a des valeurs positives lorsqu'un événement est plus probable et des valeurs négatives pour un événement moins probable et 0 quand on a autant de chances qu'il se produise que l'inverse (p = 0.5)

```
invlogit <- function(x) {return(exp(x)/(1+exp(x)))}
> mod <- glm(prop ~ sex + dose, weights = n, data=d, family = binomial)
> summary(mod)

             Estimate Std. Error z value Pr(>|z|)
(Intercept)  -6.6042     1.0208  -6.469 9.83e-11 ***
sexM          3.5689     0.7931   4.500 6.81e-06 ***
dose          0.8682     0.1236   7.025 2.15e-12 ***
```

GLM binomial

Représentation graphique

```
X <- cbind(1, 0, seq(0, 20, 0.1))
pred <- X %*% coef(mod)
se <- sqrt(diag(X %*% vcov(mod) %*% t(X)))
lwr <- invlogit(pred - se)
upr <- invlogit(pred + se)
pred <- invlogit(pred)

XM <- cbind(1, 1, seq(0, 20, 0.1))
predM <- XM %*% coef(mod)
seM <- sqrt(diag(X %*% vcov(mod) %*% t(X)))
lwrM <- invlogit(predM - se)
uprM <- invlogit(predM + se)
predM <- invlogit(predM)

dev.new(9/2.54, 9/2.54)
par(mar = c(3,3,3,1), mgp = c(1.75, 0.6, 0), cex = 0.9)
plot(prop ~ dose, data=d, pch = c(0,1)[as.numeric(d$sex)
    cex = 1.2, ylab = "Proportion d'individus morts")
lines(y = lwr, x = X[,3], lty = 1, col = "grey80")
lines(y = upr, x = X[,3], lty = 1, col = "grey80")
lines(y = pred, x = X[,3], lty = 1, col = "black")

lines(y = lwrM, x = X[,3], lty = 2, col = "grey80")
lines(y = uprM, x = X[,3], lty = 2, col = "grey80")
lines(y = predM, x = X[,3], lty = 2, col = "black")

legend(x = "top", inset = -0.25, xpd = NA, bty = "n", lty = 1:2, pch = 0:1 ,
       legend = c("Female", "Male"))
```



GLM binomial

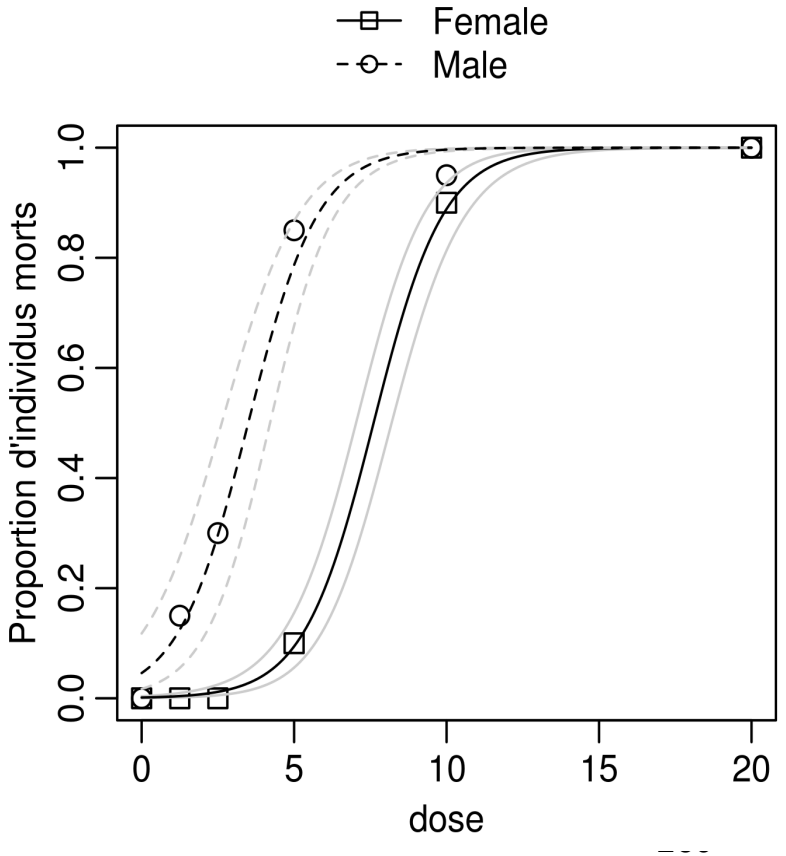
Interprétation (2)

Le signe peut s'interpréter directement : il y a plus de mortalité chez les mâles à la dose 0 (et aux autres doses) que chez les femelles et la mortalité augmente quand la dose augmente chez les femelles (et chez les mâles aussi puisqu'il n'y a pas d'interaction)

La sigmoide a une pente maximale pour p = 0.5.

A cet endroit la relation est à peu près linéaire avec une pente = à beta/4 (pour le lien logit uniquement!)

Soit ici :
0.8682 /4 = 0.21
Quand la proportion de morts est à 0.5, elle passe à 0.5 + 0.21 si on augmente la dose d'une unité



GLM binomial

Autres fonctions de lien

```
> mod <- glm(prop ~ sex + dose, weights = n, data=d, family = binomial)
> mprobit <- glm(prop ~ sex + dose, weights = n, data=d, family = binomial(link = "probit"))
> mcauchit <- glm(prop ~ sex + dose, weights = n, data=d, family = binomial(link = "cauchit"))
> mcloglog <- glm(prop ~ sex + dose, weights = n, data=d, family = binomial(link = "cloglog"))
Messages d'avis :
1: glm.fit: l'algorithme n'a pas convergé
2: glm.fit: des probabilités ont été ajustées numériquement à 0 ou 1
> mlog <- glm(prop ~ sex + dose, weights = n, data=d, family = binomial(link = "log"))
Erreur : impossible de trouver un jeu de coefficients correct : prière de fournir
des valeurs initiales
```

Typiquement, on devrait prendre le log de la dose quand on utilise le lien cloglog

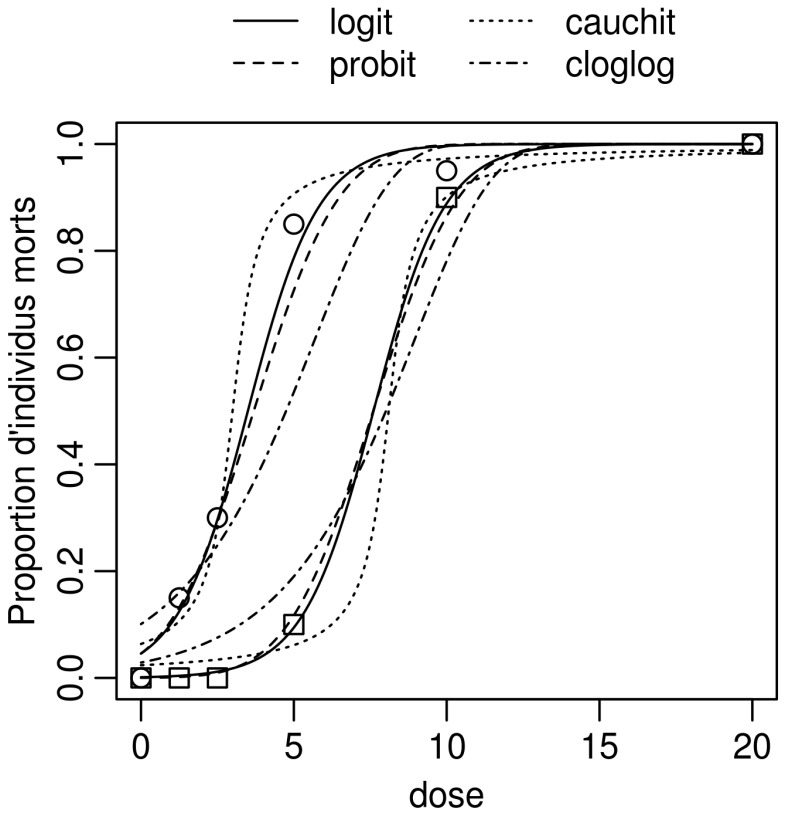
```
> compmod <- function(models) {
+     data.frame(
+         deviance = sapply(models, deviance),
+         AICc = sapply(models, aic)[2,],
+         QAICc = sapply(models, aic)[4,],
+         PsRsq = sapply(models, pseudoRsq),
+         disp = sapply(models, overdisp)[1,]
+         )
+ }
>
> compmod(list(mod, mprobit, mcauchit, mcloglog))
   deviance     AICc    QAICc PsRsq      disp
1  6.723949 31.65751 28.00706 0.996 1.5852427
2  9.083086 34.01665 21.99738 0.991 3.0202049
3  9.021243 33.95481 38.66909 0.994 0.6411044
4 26.159992 51.09356 27.71492 0.942 3.0065462
```

GLM binomial

Autres fonctions de lien

```
X <- cbind(1, 0, seq(0, 20, 0.1))
pred <- invlogit(X %*% coef(mod))
predprobit <- make.link("probit")$linkinv(X %*% coef(mprobit))
predcauchit <- make.link("cauchit")$linkinv(X %*% coef(mcauchit))
predcloglog <- make.link("cloglog")$linkinv(X %*% coef(mcloglog))

dev.new(9/2.54, 9/2.54)
par(mar = c(3,3,3,1), mgp = c(1.75, 0.6, 0), cex = 0.9)
plot(prop ~ dose, data=d, pch = c(0,1)[as.numeric(d$sex)], cex = 1.2,
     ylab = "Proportion d'individus morts")

lines(y = pred, x = X[,3], lty = 1, col = "black")
lines(y = predprobit, x = X[,3], lty = 2, col = "black")
lines(y = predcauchit, x = X[,3], lty = 3, col = "black")
lines(y = predcloglog, x = X[,3], lty = 4, col = "black")


legend(x = "top", inset = -0.25, xpd = NA, bty = "n", lty = 1:4,
      ncol = 2,
     legend = c("logit", "probit", "cauchit", "cloglog"))

X <- cbind(1, 1, seq(0, 20, 0.1))
pred <- invlogit(X %*% coef(mod))
predprobit <- make.link("probit")$linkinv(X %*% coef(mprobit))
predcauchit <- make.link("cauchit")$linkinv(X %*% coef(mcauchit))
predcloglog <- make.link("cloglog")$linkinv(X %*% coef(mcloglog))

lines(y = pred, x = X[,3], lty = 1, col = "black")
lines(y = predprobit, x = X[,3], lty = 2, col = "black")
lines(y = predcauchit, x = X[,3], lty = 3, col = "black")
lines(y = predcloglog, x = X[,3], lty = 4, col = "black")
```

4 fonctions de lien fonctionnent ici. Elles sont toutes très similaires (sigmoïde)



GLM binomial

Autres fonctions de lien

NB : il est souvent difficile de distinguer la meilleure fonction de lien sur base des données en particulier pour les modèles Binomiaux.

Souvent le choix de la fonction de lien est guidé par une connaissance préalable du phénomène étudié, des raisons pragmatiques ou des raisons historiques.

On utilise le plus souvent le logit qui reste interprétable en termes de odds ratio.
Mais par exemple en écotoxicologie on utilise très souvent le lien probit.

GLM binomial

DL50

Une question classique pour ce genre de jeux de données est de savoir quelle est la dose pour laquelle on observe 50 % de mortalité

Quand p = 0.5, logit(p) = 0
la dose correspondante est donc
6.6042/0.8682 = 7.6 l/ha pour les femelles et
(6.6042-3.5689)/0.8682 = 3.5 l/ha pour les mâles

```
> summary(mod)

             Estimate Std. Error z value Pr(>|z|)
(Intercept)  -6.6042     1.0208  -6.469 9.83e-11 ***
sexM          3.5689     0.7931   4.500 6.81e-06 ***
dose          0.8682     0.1236   7.025 2.15e-12 ***
```

GLM binomial

DL50

On peut aussi obtenir des erreurs standard et IC pour les valeurs de x correspondant à un y donné

```
> mod <- glm(prop ~ sex + dose, weights = n, data=d, family = binomial)
>
> library(MASS)
> LD <- dose.p(mod, cf = c(1,3), p=c(0.1, 0.5, 0.9))
> data.frame(
+     LD = LD[1:length(LD)],
+     CI_low = LD[1:length(LD)] - attr(LD,"SE")*1.96,
+     CI_high = LD[1:length(LD)] + attr(LD,"SE")*1.96
+ )
                LD   CI_low   CI_high
p = 0.1:  5.075988 3.803034  6.348942
p = 0.5:  7.606776 6.475813  8.737740
p = 0.9: 10.137564 8.746527 11.528602
>
> d$sex <- relevel(d$sex, ref = "M")
> mod <- glm(prop ~ sex + dose, weights = n, data=d, family = binomial)
> LD <- dose.p(mod, cf = c(1,3), p=c(0.1, 0.5, 0.9))
> data.frame(
+     LD = LD[1:length(LD)],
+     CI_low = LD[1:length(LD)] - attr(LD,"SE")*1.96,
+     CI_high = LD[1:length(LD)] + attr(LD,"SE")*1.96
+ )
                LD     CI_low  CI_high
p = 0.1: 0.9653343 0.09208808 1.838581
p = 0.5: 3.4961225 2.78305147 4.209194
p = 0.9: 6.0269107 4.90815140 7.145670
```

Tables de contingence - log-linear models

Lorsqu'on mesure sur une série de sujets uniquement des variables qualitatives (nominales ou ordinales), on peut organiser les données sous forme de tableau où on compte le nombre d'individus observé pour chaque combinaison des niveaux des variables qualitatives.

C'est ce qu'on appelle une table de contingence.

Traditionnellement on analyse de telles tables à R lignes x C colonnes avec un test chi carré ou un G test d'hétérogénéité, pour avoir si il y a une association entre les deux variables qualitatives.

On peut également analyser ce genre de cas mais aussi des tables beaucoup plus complexes avec des GLM à distribution de Poisson un peu particuliers ("modèles log-linéaires") où y est le nombre d'observations et toutes les variables qualitatives sont du côté "x" de l'équation

Tables de contingence - log-linear models

Exemple de table de contingence

On a observé des plantes sur lesquelles on a noté la présence de pucerons, la présence de parasitoïdes et la qualité de la plante (haute/basse selon la teneur en azote).

Il s'agit d'une table 2 x 2 x 2 : 3 variables qualitatives à 2 niveaux chacune

```
d <- data.frame(
    nb = c(48, 17, 27, 11, 3, 11, 9, 37),
    plantqual = rep(c("low", "high"), each = 4),
    parasit = rep(c("absent", "absent", "present", "present"), 2),
    aphids = rep(c("absent", "present"), 4))
```

```
> d
  nb plantqual parasit  aphids
1 48       low  absent  absent
2 17       low  absent present
3 27       low present  absent
4 11       low present present
5  3      high  absent  absent
6 11      high  absent present
7  9      high present  absent
8 37      high present present
```


Exemple tiré de Scheiner & Guerevitch : Design and analysis of ecological experiments

Tables de contingence - log-linear models

Table 2 x 2

On va commencer par analyser la table 2 x 2, pucerons x parasitoïdes. La question est de avoir si il y a une association entre la présence de pucerons et la présence de parasitoïdes.

On va comparer 4 approches qui donnent des résultats similaires à identiques : test chi carré, G test, modèle loglinéaire, modèle binomial

```
> d2 <- aggregate(d["nb"], d[, c("parasit", "aphids")], sum)
>
> matr <- xtabs(nb~., d2)
> matr
         aphids
parasit   absent present
  absent      51      28
  present     36      48
```

Pour les tests Chi carré et G on a besoin des données sous forme de matrice

Tables de contingence - log-linear models

Le G test n'est pas implémenté dans R base. La fonction suivante produit un test de G pour une table RxC avec des corrections de continuité pour les petits nombres (selon Sokal & Rholf 1995).

```
G.test.het <- function (obs, continuity.cor = TRUE) {
    obs <- as.matrix(obs)
    Gh <- 2* ( sum(obs* log(obs)) - sum(colSums(obs) * log(colSums(obs))) -
         sum(rowSums(obs) * log(rowSums(obs))) +
         sum(sum(rowSums(obs)) * log(sum(rowSums(obs))))  )
    df <- (nrow(obs)-1)*(ncol(obs)-1)

    # williams continuity correction for 2x2 tables
    if (nrow(obs)*ncol(obs) == 4 & continuity.cor == TRUE) {
         a <- obs[1,1] ; b <- obs[1,2] ;c <- obs[2,1] ;d <- obs[2,2] ; n <- sum(a,b,c,d)
         q <- 1+ ( (1/(6*n)) * ( (n/(a+b)) + (n/(c+d)) -1 )  * ( (n/(a+c)) + (n/(b+d)) -1))
         Gh <- Gh/q
         }
    # williams continuity correction for a x b tables
    if (nrow(obs)*ncol(obs) > 4 & continuity.cor == TRUE) {
         a <- nrow(obs); b <- ncol(obs) ; n <- sum(obs)
         q <- 1+ (((a+1)*(b+1))/(6*n) )
         Gh <- Gh/q
         }

    p <- pchisq(Gh, df, lower.tail = FALSE)
    heterogeneity <- as.data.frame(cbind(round(Gh,4),round(df,0),round(p,5)))
    colnames(heterogeneity) <- c("G","df","p")
    return(heterogeneity)
}
```

Tables de contingence - log-linear models

Table 2 x 2

Chi carré et G test d'indépendance (ou d'hétérogénéité)

NB : de manière générale, le G test est plus recommandé que le Chi carré pour diverses raisons (ea moins sensible quand certaines cases de la table contiennent des nombres proches de 0) Voir ea Sokal & Rholf 1995

Les deux tests montrent qu'il y a une association significative entre la présence de pucerons et de parasitoïdes.

```
> chisq.test(matr)

    Pearson's Chi-squared test with Yates' continuity correction

data:  matr
X-squared = 6.856, df = 1, p-value = 0.008835


> G.test.het(matr,continuity.cor=FALSE)
       G df       p
1 7.7714  1 0.00531
```

Tables de contingence - log-linear models

Table 2 x 2

Modèle log-linéaire : l'interaction aphids x parasitoïds teste l'association entre ces deux facteurs exactement de la même manière que le chi2 et le G test

```
> mod <- glm(nb ~ parasit * aphids, data=d2, family = poisson)
> summary(mod)
Coefficients:
                            Estimate Std. Error z value Pr(>|z|)
(Intercept)                   3.9318     0.1400  28.079  < 2e-16 ***
parasitpresent               -0.3483     0.2177  -1.600  0.10958
aphidspresent                -0.5996     0.2352  -2.549  0.01079 *
parasitpresent:aphidspresent  0.8873     0.3224   2.752  0.00592 **
```

```
> drop1(mod, test = "Chisq")
               Df Deviance    AIC    LRT Pr(>Chi)
<none>              0.0000 30.088
parasit:aphids  1   7.7714 35.859 7.7714 0.005308 **
```

```
> G.test.het(matr,continuity.cor=FALSE)
      G df       p
1 7.7714  1 0.00531
```

```
> d2
  parasit  aphids nb
1  absent  absent 51
2 present  absent 36
3  absent present 28
4 present present 48
```

Le test de rapport de vraisemblance de l'interaction est strictement identique au G test

Tables de contingence - log-linear models

Table 2 x 2

Le modèle log-linéaire est également étroitement lié au modèle binomial pour donnée binaires (et au modèle multinomial). Dans ce cas étant donné qu'au moins une des variables n'a que deux niveaux, on peut obtenir exactement le même résultat avec une régression logistique

```
> parasit <- rep(d2$parasit, times = d2$nb)
> aphids <- rep(d2$aphids, times = d2$nb)
> mod <- glm(parasit ~ aphids, family = binomial)
> summary(mod)
             Estimate Std. Error z value Pr(>|z|)
(Intercept)   -0.3483     0.2177  -1.600  0.10958
aphidspresent  0.8873     0.3224   2.752  0.00592 **

> drop1(mod, test = "Chisq")

parasit ~ aphids
       Df Deviance    AIC    LRT Pr(>Chi)
<none>      218.04 222.04
aphids  1   225.81 227.81 7.7714 0.005308 **

> G.test.het(matr,continuity.cor=FALSE)
       G df       p
1 7.7714  1 0.00531
```

Tables de contingence - log-linear models

Table 2 x 2 x 2 (3 facteurs)

L'avantage de l'approche GLM est qu'elle permet d'analyser des tables à plus de 2 facteurs et de mesurer l'association entre 2 (ou plus) facteurs tout en contrôlant pour l'effet d'autres facteurs (association marginale)

L'interaction pucerons x parasitoïdes x plante n'est pas significative. Il n'y a donc pas d'association entre ces 3 variables à la fois.

Par exemple il ne faut pas à la fois des plantes de qualité et des pucerons pour que les parasitoïdes soient plus présents.

```
> mod <- glm (nb ~ .^3, data=d, family = poisson)
> drop1(mod, test = "Chisq")

nb ~ (plantqual + parasit + aphids)^3
                         Df   Deviance    AIC        LRT Pr(>Chi)
<none>                      0.00000000 52.534
plantqual:parasit:aphids  1 0.00085458 50.535 0.00085458   0.9767
```

Tables de contingence - log-linear models

Table 2 x 2 x 2 (3 facteurs)

```
> mod <- glm (nb ~ .^2, data=d, family = poisson)
> summary(mod)

Coefficients:
                              Estimate Std. Error z value Pr(>|z|)
(Intercept)                     1.1097     0.4336   2.559  0.01049 *
plantquallow                    2.7608     0.4363   6.328 2.49e-10 ***
parasitpresent                  1.0838     0.4330   2.503  0.01230 *
aphidspresent                   1.2852     0.4358   2.949  0.00319 **
plantquallow:parasitpresent    -1.6573     0.4176  -3.969 7.23e-05 ***
plantquallow:aphidspresent     -2.3205     0.4171  -5.563 2.65e-08 ***
parasitpresent:aphidspresent    0.1331     0.3892   0.342  0.73233

> drop1(mod, test = "Chisq")

nb ~ (plantqual + parasit + aphids)^2
                  Df Deviance    AIC    LRT  Pr(>Chi)
<none>                 0.001 50.535
plantqual:parasit  1   17.338 65.872 17.337 3.130e-05 ***
plantqual:aphids   1   36.993 85.527 36.992 1.186e-09 ***
parasit:aphids     1    0.117 48.651  0.116    0.7332
```

Tables de contingence - log-linear models

Table 2 x 2 x 2 (3 facteurs)

L'interaction pucerons x parasitoides n'est pas significative contrairement à l'analyse sur la table 2 x 2 : il n'y a pas d'association entre les pucerons et les parasitoïdes une fois qu'on a contrôlé pour la qualité de la plante

Par contre, il y a une association positive entre la qualité de la plante et la présence de pucerons d'une part et la présence de parasitoïdes d'autre part.
L'association parasitoïdes - pucerons dans la analyse de la table 2 x 2 était donc due à l'attraction commune pour des plantes de haute qualité.

Ce genre d'effet est fréquent en particulier quand il y a une répartition non uniforme dans les différentes classes
(ex étude fumeurs, survie, classe d'age)

Tables de contingence - log-linear models

Table 2 x 2 x 2 (3 facteurs)

NB : on aurrait pu analyser également cette table 2x2x2 avec un modèle binomial parasitoïdes ~ pucerons + qualité de la plante et on aurait trouvé exactement les même résultats.

En général on utilise un modèle log linéaire quand on considère qu'il n'y a pas une variable qui est "expliquée" par les autres.
On mesure simplement l'association entre les différentes variables sur un pied d'égalité.

Dans les cas où toutes les variables ont plus de 2 niveaux le modèle binomial n'est plus une alternative mais il peut être remplacé par un modèle multinomial.

Surdispersion

Une caractéristique importante à vérifier dans les GLM est de savoir si la relation entre la moyenne et la variance respecte bien celle du modèle choisi (Poisson ou Binomial)

Pour la loi de Poisson :

$$Var(Y) = \varphi\lambda$$

Pour la loi Binomiale :

$$Var(Y) = \varphi\, Np(1-p)$$

Le paramètre φ est appelé "coefficient de surdispersion". Lorsqu'il est différent de 1, le modèle ne suit pas à proprement parler une loi de Poisson ou Binomiale.

Surdispersion

Comment estimer/détecter la surdispersion ?

Le rapport entre la déviance résiduelle et le nombre de degrés de liberté ("residual df") données par le summary donne une idée approximative de la surdispersion.

```
> mod <- glm(prop ~ sex + dose, weights = n, data=d, family = binomial)
> mod2 <- glm(cbind(dead, n-dead) ~ sex + dose, data=d, family = binomial)
> summary(mod)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -6.6042     1.0208  -6.469 9.83e-11 ***
sexM          3.5689     0.7931   4.500 6.81e-06 ***
dose          0.8682     0.1236   7.025 2.15e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 236.7523  on 11  degrees of freedom
Residual deviance:   6.7239  on  9  degrees of freedom
AIC: 28.658

Number of Fisher Scoring iterations: 6
```

Surdispersion :
6.7239/9 = 0.7471

Surdispersion

Comment estimer/détecter la surdispersion ?

On obtient une estimation plus précise sur base des "Pearson residuals"

NB : Dans les GLM, il existe plusieurs formes différentes de résidus :

Deviance residals, Pearson residuals, Response residuals, Working residuals

Les response residuals correspondent aux résidus tel qu'on les a vus jusqu'à présent : valeurs observées moins valeurs prédites.

La variance de ces résidus augmente donc en fonction de la moyenne pour les modèles de Poisson et Binomial.
On les utilise rarement pour les plots de résidus (tout comme les "working residuals")

Surdispersion

Comment estimer/détecter la surdispersion ?

Les Pearson residuals sont des résidus standardisés par l'écart type théorique des valeurs prédites.
On peut s'en servir pour estimer la surdispersion.

$$z_i = \frac{y_i - \hat{y}_i}{sd\left(\hat{y}_i\right)}$$

$$overdispersion = \frac{1}{n-k}\sum_{i=1}^{n} z_i^2$$

NB = la fonction residuals (ou resid) donne par défaut les "deviance residuals" qui sont une autre forme de standardisation des résidus.
Attention à ne pas confondre avec la "residual deviance".
Il se fait que la somme du carré des "deviance residuals" donne la "residual deviance"...

Pour les plots de résidus, on utilise les pearson ou deviance residuals.
Attention que `mod$residuals` donnes les "working residuals"
voir ?residuals pour plus de détails

Surdispersion

Comment estimer/détecter la surdispersion ?

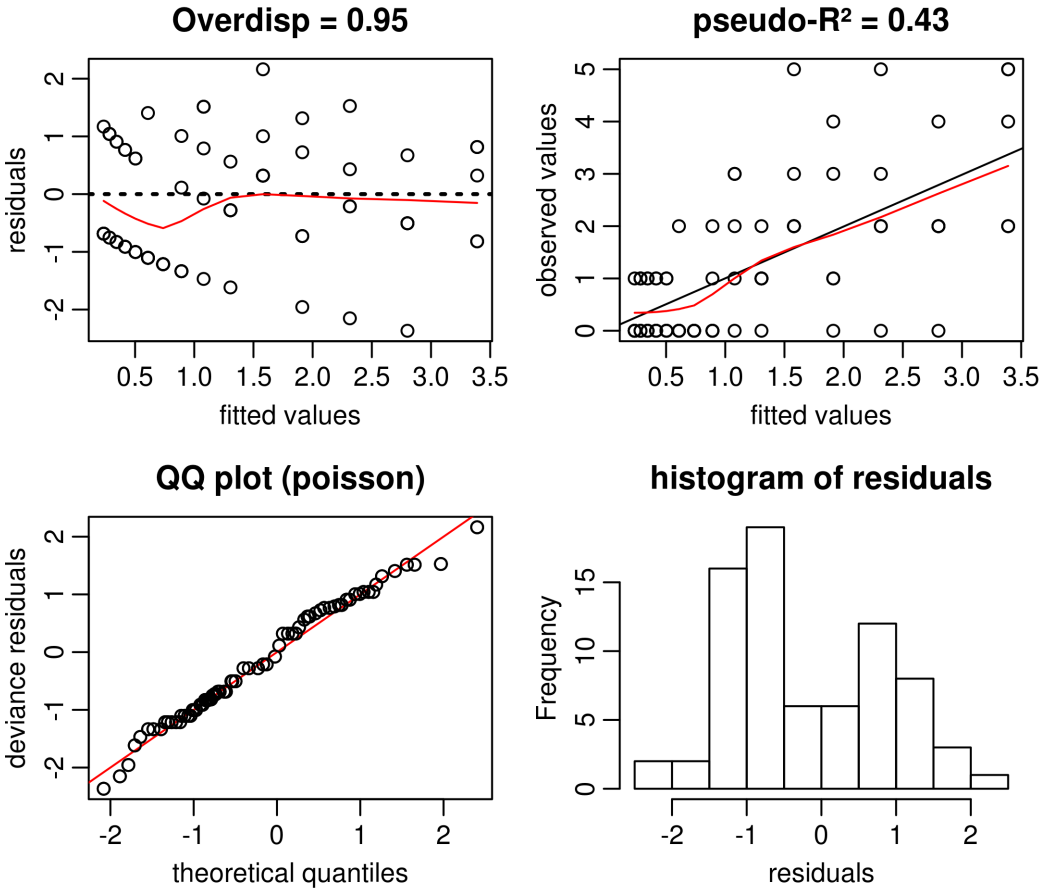
La fonction suivante permet donc d'estimer la surdispersion sur base de la déviance et sur base des "Pearson residuals" :

```
overdisp <- function(mod) {

    k <- attr(logLik(mod),"df")
    n <- length(fitted(mod))

    pearsonresid <- (1/(n-k)) * sum(resid(mod,"pearson")^2)
    dev <- deviance(mod)/(n-k)

    result <- c(pearsonresid,dev)
    names(result) <- c("pearsonresid","deviance")
    return(result)
    }
```

Surdispersion

Comment estimer/détecter la surdispersion ?

NB : la fonction diagplot (mytoolbox.R) vous donne le coefficient de surdispersion (sur base des résidus de Pearson), le pseudo-R^2 et un qq plot pour la distribution choisie (poisson dans cet exemple)



Pour les données binaires, les plots de résidus par rapport aux valeurs prédites sont en général inutilisables.
Si on a suffisamment de données, il peut être utile de regrouper les résidus par groupes et d'en faire la moyenne ("binned residuals).

Les plots résidus vs variables explicatives (diagplot2) sont par contre utiles en général

Surdispersion

Comment diminuer la surdispersion ?

La surdispersion peut avoir de nombreuses causes :

- valeurs extrêmes
- relations non linéaires
- interaction importante absente du modèle
- variable explicative importante absente du modèle
- non indépendance des observations
...

On peut souvent réduire la surdispersion en agissant sur ces points : transformations, modification du modèle, changement de fonction de lien, ajout de facteurs aléatoires ou d'interactions,...

Une surdispersion de l'ordre de 2 à 10 est cependant assez fréquente

Surdispersion

La surdispersion a des conséquences sur l'inférence

1) il faut utiliser les erreurs standard multipliées par la racine carrée du coefficient de surdispersion
(pour les intervalles de confiance, tests de Wald etc...)

2) il faut utiliser la log-vraissemblance divisée par le coefficient de surdispersion
pour les comparaisons de modèles (avec une distribution de F au lieu de Chi²), pour les AIC (=QAIC)

En pratique donc les inférences sont trop peu conservatives (les p valeurs, intervalles de confiance etc... sont trop petits).

Par contre les coefficients sont non biaisés.

Surdispersion

Comment traiter la surdispersion ?

Si après avoir essayé de la diminuer, il reste de la surdispersion il reste plusieurs possibilités :

- utiliser des corrections pour l'inférence
- utiliser une approche de quasi-vraisemblance qui estime la surdispersion et corrige les erreurs standard, Déviance etc...
- utiliser d'autres distributions comprenant un paramètre de variance

Negative Binomiale (pour une distribution de Poisson),

Beta Binomiale (pour une distribution binomiale)

- utiliser un modèle mixte et ajouter un effet aléatoire pour chaque observation

Si la surdispersion est vraiment très élevée, il est vraisemblable que le modèle n'est pas adapté et il vaut sans doute mieux renoncer à l'utiliser tel quel même avec des "corrections".

Surdispersion

Simulation de données

Simulation de données pour un modèle de Poisson surdispersé.
Deux méthodes donnant des résultats légèrement différents
(surdispersion additive vs multiplicative)
pour une surdispersion identique

1) simulation au moyen d'une distribution négative binomiale qui permet de fixer exactement le paramètre de surdispersion (multiplicative) à 4

```
overdisp <- 4
n <-5
x <- rep(0:14, each = n)
set.seed(12)
lambda <- 1 - 0.15 * x
set.seed(1)
y2 <- rnbinom(n*15, size = exp(lambda)/(overdisp-1), mu = exp(lambda))
```

Surdispersion

Simulation de données

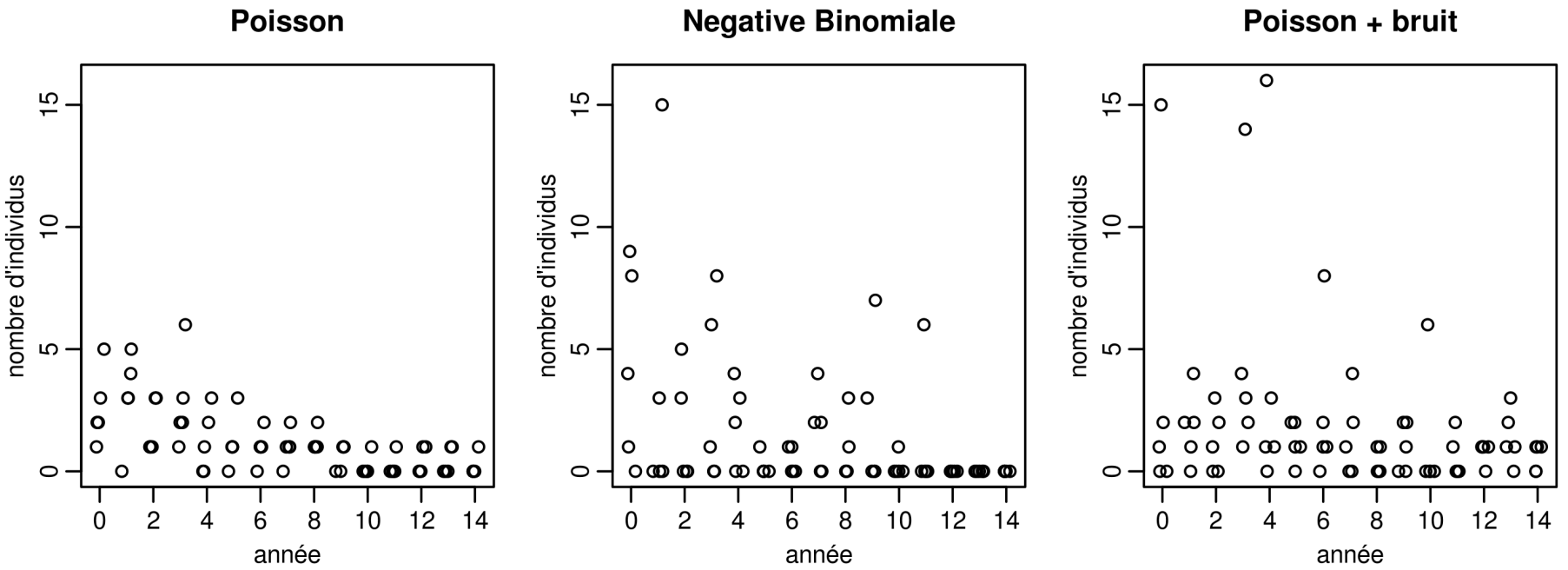
2) simulation en ajoutant du bruit (surdispersion additive) au "linear predictor". On a choisi un écart type de façon à obtenir une surdispersion de 4 également

```
n <-5
x <- rep(0:14, each = n)
set.seed(12)
lambda <- 1 - 0.15 * x + rnorm(n*15, 0, 1.15)
set.seed(1)
y3 <- rpois(n*15, exp(lambda))
```

Surdispersion

Comment traiter la surdispersion ?

Simulation de données pour modèle de poisson surdispersé



Surdispersion

Tests de Wald corrigés :

```
> mod <- glm(y2 ~ x, family = poisson)
> summary(mod)
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   1.3840     0.1423   9.723  < 2e-16 ***
x            -0.2015     0.0278  -7.250 4.18e-13 ***

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 280.22  on 74  degrees of freedom
Residual deviance: 215.94  on 73  degrees of freedom
AIC: 297.63

> overdisp(mod)
pearsonresid     deviance
    4.069687     2.958097
> summary(mod, dispersion = overdisp(mod)[1])

            Estimate Std. Error z value Pr(>|z|)
(Intercept)  1.38397    0.28715   4.820 1.44e-06 ***
x           -0.20154    0.05608  -3.594 0.000326 ***

(Dispersion parameter for poisson family taken to be 4.069687)

    Null deviance: 280.22  on 74  degrees of freedom
Residual deviance: 215.94  on 73  degrees of freedom
AIC: 297.63
```

Analyse classique non corrigée

Tests de Wald corrigés. NB : les coefficients sont les mêmes mais leur erreur standard a été multipliée par sqrt(4.07)

Attention l'AIC n'est pas corrigé et n'est pas correct !!

Surdispersion

Comparaison de modèles

On utilise une statistique de test adaptée du test de rapport de vraisemblance qui suit approximativement une distribution de F

$$F = \frac{\left(D_{small} - D_{large}\right) / \left(df_{small} - df_{large}\right)}{\varphi}$$

```
(> (F <- ((deviance(modnull)-deviance(mod))/
    (modnull$df.residual - mod$df.residual)) / overdisp(mod)[2])
deviance
21.73132
> pf(F, modnull$df.residual - mod$df.residual,
            mod$df.residual, lower.tail = FALSE)
   deviance
1.38063e-05

> drop1(mod, test = "F")

       Df Deviance    AIC F value    Pr(>F)
<none>      215.94 297.63
x       1   280.22 359.91  21.731 1.381e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
Message d'avis :
In drop1.glm(mod, test = "F") :
  le test F implique une famille 'quasipoisson'
```

NB : on a utilisé ici le coefficient de surdispersion basé sur la déviance qui est celui utilisé par la fonction drop1

Surdispersion

Quasilikelihood

L'argument 'family" permet de spécifier "quasipoisson" et "quasibinomial". Dans ce cas le coefficent de surdispersion est automatiquement estimé et les erreurs standard ajustées.

```
> mod <- glm(y2 ~ x, family = poisson)
> summary(mod)
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   1.3840     0.1423   9.723  < 2e-16 ***
x            -0.2015     0.0278  -7.250 4.18e-13 ***

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 280.22  on 74  degrees of freedom
Residual deviance: 215.94  on 73  degrees of freedom
AIC: 297.63

> mod <- glm(y2 ~ x, family = quasipoisson)
> summary(mod)
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.38397    0.28716   4.820 7.61e-06 ***
x           -0.20154    0.05608  -3.594 0.000589 ***

(Dispersion parameter for quasipoisson family taken to be 4.069724)

    Null deviance: 280.22  on 74  degrees of freedom
Residual deviance: 215.94  on 73  degrees of freedom
AIC: NA
```

L'AIC n'est plus disponible

Surdispersion

Quasilikelihood
Comparaison de modèles

```
> mod <- glm(y2 ~ x, family = quasipoisson)
> modnull <- glm(y2 ~ 1, family = quasipoisson)
> anova(mod, modnull, test="F")

  Resid. Df Resid. Dev Df Deviance      F    Pr(>F)
1        73     215.94
2        74     280.22 -1  -64.283 15.796 0.0001641 ***

> (F <- ((deviance(modnull)-deviance(mod))/ (
    modnull$df.residual - mod$df.residual)) / overdisp(mod)[1])
    15.79565
> pf(F, modnull$df.residual - mod$df.residual,
            mod$df.residual, lower.tail = FALSE)
0.0001640992

> drop1(mod, test = "F")
Single term deletions

Model:
y2 ~ x
       Df Deviance F value    Pr(>F)
<none>      215.94
x       1   280.22  21.731 1.381e-05 ***
```

curieusement, drop1 n'utilise pas l'estimation du coefficient de surdispersion estimé par le modèle `car::Anova(mod, test = "F")` par contre l'utilise

Surdispersion

Negative Binomial model

On peut utiliser la fonction `nb.glm` du package MASS qui est construite sur `glm`

```
> mod <- glm(y2 ~ x, family = quasipoisson)
> summary(mod)
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.38397    0.28716   4.820 7.61e-06 ***
x           -0.20154    0.05608  -3.594 0.000589 ***

(Dispersion parameter for quasipoisson family taken to be 4.069724)

    Null deviance: 280.22  on 74  degrees of freedom
Residual deviance: 215.94  on 73  degrees of freedom
AIC: NA
```

Pour les modèles binomiaux, on utilisera un modèle Beta Binomial disponible dans le package aod

L'interprétation est exactement identique. On peut choisir les mêmes fonctions de lien.

```
> library(MASS)
> mod <- glm.nb(y2 ~ x)
> summary(mod)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   1.4410     0.4198   3.433 0.000598 ***
x            -0.2106     0.0577  -3.650 0.000262 ***

(Dispersion parameter for Negative Binomial(0.3306) family taken to be 1)

    Null deviance: 70.272  on 74  degrees of freedom
Residual deviance: 56.584  on 73  degrees of freedom
AIC: 210.29
```

paramètre theta qui est une combinaison de paramètres

Attention, le modèle négatif binomial peut lui-même être surdispersé !

On dispose ici d'un AIC valide