Formation Statistiques - GLM : exercices

Gilles San Martin - gilles.sanmartin@gmail.com

Septembre 2013

Contents

Exercice 1 : simulation de jeux de données simples . . . 2
Exercice 2 : régression linéaire simple . . . 6
Exercice 2bis : micro analyse de puissance par simulation . . . 12
Exercice 3 : ANOVA1 et comparaisons multiples . . . 15
Exercice 4 : 2 variables explicatives qualitatives (ANOVA2 sans interaction) . . . 22
Exercice 5 : régression multiple . . . 31
Exercice 6 : interaction simple entre 2 variables qualitatives . . . 43
Exercice 7 : interaction entre une variable quantitative et une variable qualitative . . . 49
Exercice 8 : non linéarité et transformation de variables . . . 55
Exercice 9a : GLM logistique à une variable explicative . . . 70
Exercice 9b : GLM logistique à deux variables explicatives . . . 72
Exercice 10 : Vaches laitières - modèle mixte gaussien en carré latin . . . 79
Exercice 11 : Modèle mixte binomial et sélection de modèle . . . 86

Note préliminaire
Les premiers exercices qui suivent utilisent des faux jeu de données simulés qui du coup se comportent de manière idéale. Dans la réalité, l'analyse de tels jeux de données devrait souvent se faire différemment. Par exemple les données de comptage suivent rarement une distribution normale (sauf pour des moyennes élevées). Lorsqu'on fait des suivi de sites au cours du temps on garde en général les mêmes sites d'année en année mais il faut alors tenir compte de l'effet site dans l'analyse, etc...

Exercice 1 : simulation de jeux de données simples

Simulez 2 jeux de données :

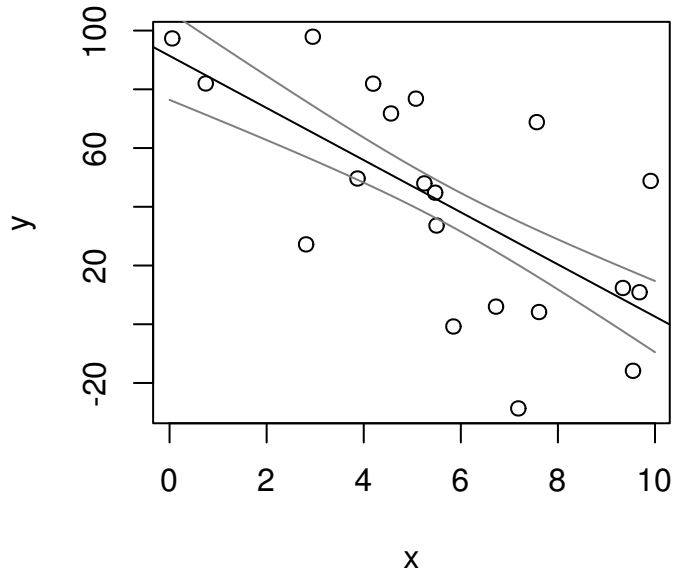
- l'un représente la relation linéaire négative entre deux variables quantitatives continues
- l'autre la relation entre une variable continue et une variable qualitative à 10 niveaux possibles

Analysez ces deux jeux de données et vérifiez que le modèle estime correctement les paramètres que vous avez choisis. Changez le nombre de répétitions, la variance résiduelle,... et visualisez l'effet sur les paramètres, leurs erreurs standard et le R^2.
Faites une représentation graphique des données, des valeurs prédites et de leurs erreurs standard.

```
# relation linéaire négative
n = 20
alpha <- 100
beta <- -10
sigma <- 30

x <- runif(n, 0, 10)
y <- alpha + beta*x + rnorm(n, 0, sigma)
mod <- lm(y ~ x)
summary(mod)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -56.266 -20.363   2.574  22.597  45.398
##
## Coefficients:
##             Estimate Std. Error t value   Pr(>|t|)
## (Intercept)   91.406     15.001   6.093 0.00000931 ***
## x             -8.880      2.371  -3.745    0.00148 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 29.22 on 18 degrees of freedom
## Multiple R-squared:  0.438,  Adjusted R-squared:  0.4067
## F-statistic: 14.03 on 1 and 18 DF,  p-value: 0.001482
```

```
X <- cbind(1, seq(0,10,0.1))
pred <- X %*% coef(mod)
se <- sqrt(diag(X %*% as.matrix(vcov(mod)) %*% t(X)))

plot(y~x)
abline(mod)
lines(x = X[,2], y = pred + se, col = "gray50")
lines(x = X[,2], y = pred - se, col = "gray50")
```

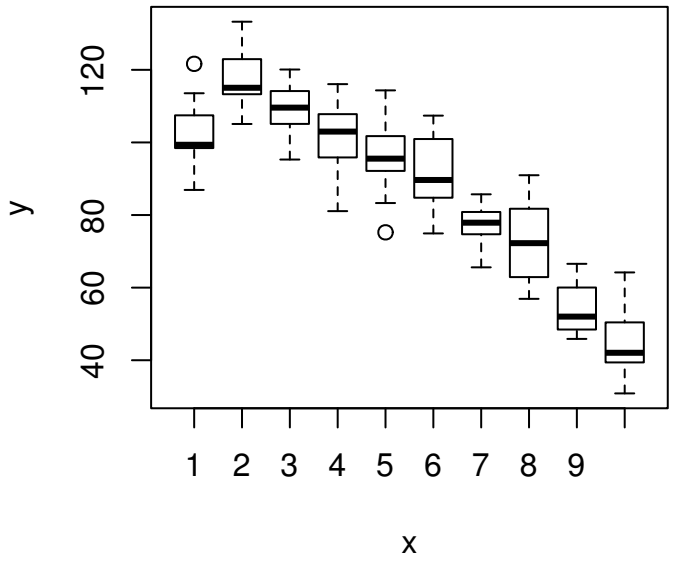



```
# variable qualitative à 10 niveaux

n = 100
B <- c(100, 20, 10, 0, -5, -10, -20, -25, -50, -55)
sigma <- 10

x <- as.factor(sample(1:10, n, replace = TRUE))
X <- model.matrix(~x)
X[1:15,]
```

```
##    (Intercept) x2 x3 x4 x5 x6 x7 x8 x9 x10
## 1            1  0  0  0  0  0  1  0  0   0
## 2            1  0  0  0  0  1  0  0  0   0
## 3            1  0  0  0  0  0  1  0  0   0
## 4            1  0  0  0  0  1  0  0  0   0
## 5            1  1  0  0  0  0  0  0  0   0
## 6            1  0  0  0  1  0  0  0  0   0
## 7            1  0  0  0  0  0  0  0  1   0
## 8            1  0  0  0  1  0  0  0  0   0
## 9            1  0  0  0  1  0  0  0  0   0
## 10           1  0  0  0  0  1  0  0  0   0
## 11           1  0  0  0  0  0  0  0  0   0
## 12           1  0  0  1  0  0  0  0  0   0
## 13           1  0  0  0  0  1  0  0  0   0
## 14           1  0  0  0  0  1  0  0  0   0
## 15           1  0  0  0  0  0  0  0  0   1
```

```
y <- X %*% B + rnorm(n, 0, sigma)
```

```
plot(y ~ x)
```

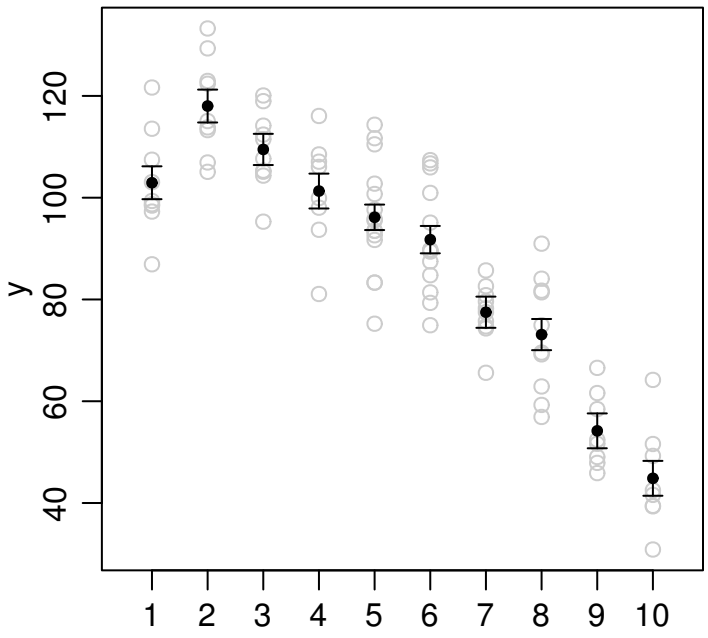



```r
mod <- lm(y ~ x)
summary(mod)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -20.908  -4.832  -1.801   5.984  19.340
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  102.937      3.230  31.866  < 2e-16 ***
## x2            15.074      4.568   3.300  0.00139 **
## x3             6.546      4.453   1.470  0.14498
## x4            -1.633      4.709  -0.347  0.72953
## x5            -6.794      4.086  -1.663  0.09982 .
## x6           -11.180      4.202  -2.661  0.00924 **
## x7           -25.446      4.453  -5.715 1.41e-07 ***
## x8           -29.838      4.453  -6.701 1.75e-09 ***
## x9           -48.755      4.709 -10.354  < 2e-16 ***
## x10          -58.086      4.709 -12.335  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.691 on 90 degrees of freedom
## Multiple R-squared:  0.8409, Adjusted R-squared:  0.825
## F-statistic: 52.87 on 9 and 90 DF,  p-value: < 2.2e-16
```

```r
# prédictions
X <- model.matrix(~ factor(1:10))
pred <- X %*% coef(mod)
se <- sqrt(diag(X %*% as.matrix(vcov(mod)) %*% t(X)))
```

```r
par(mar = c(3,3,2,1), mgp = c(1.75, 0.7, 0))
position <- as.numeric(x)-1
plot(y ~ position, xaxt="n", xlim = c(-0.5, 9.5),xlab = "", col = "grey80" )
axis(side = 1, at = sort(unique(position)), labels = levels(x))
points(x = sort(unique(position)), y = pred, pch=20 )
arrows(x0 = sort(unique(position)), y0 = pred-se, x1 = sort(unique(position)),
       y1 = pred + se, angle=90, length = 0.05, code = 3)
```



Exercice 2 : régression linéaire simple

Pour suivre l'évolution des populations d'une espèce de papillon, on sélectionne chaque année 10 sites aléatoirement répartis en Belgique. Sur chaque site on réalise plusieurs transects au cours de la saison et on en retire une abondance moyenne.

- Analysez ces données et faites une représentation graphique des données, du modèle et des erreurs standard des prédictions.
- Quel est le % de variance expliquée par le modèle ?
- Quel était le nombre de papillons moyen par transect en 2000 (estimation du modèle) ?
- Qu'elle est la tendance estimée dans cette population en nombre de papillons par an ? Calculez un intervalle de confiance à 90% sur cette valeur avec les fonctions de R.
- Qu'elle est la tendance estimée en % de diminution sur 10 ans ?
- A l'aide de la méthode bootstrap (non paramétrique) estimez un intervalle de confiance à 95% sur le % de diminution en 10 ans (NB : il n'existe pas de méthode paramétrique directe pour ce faire)
- A l'aide de la méthode bootstrap (non paramétrique) estimez un intervalle de confiance sur les valeurs prédites par le modèle. Faites en une représentation graphique (NB : ici on pourrait utiliser une méthode paramétrique à la place).

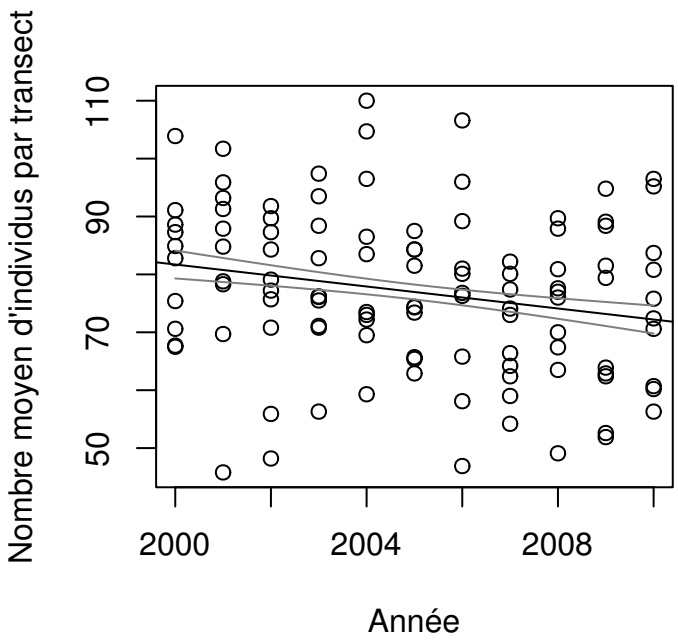
```r
# Génération du jeu de données
n <- 10
year <- rep(0:10, each = n)
set.seed(1)
y = round(80 - 1 * year + rnorm(length(year), 0, 15), 1)

# Stockage des données pour les présenter dans les énoncés
```

```r
d <- data.frame (
    nb = c(70.6, 82.8, 67.5, 103.9, 84.9, 67.7, 87.3, 91.1, 88.6, 75.4,
           101.7, 84.8, 69.7, 45.8, 95.9, 78.3, 78.8, 93.2, 91.3, 87.9,
           91.8, 89.7, 79.1, 48.2, 87.3, 77.2, 75.7, 55.9, 70.8, 84.3, 97.4,
           75.5, 82.8, 76.2, 56.3, 70.8, 71.1, 76.1, 93.5, 88.4, 73.5, 72.2,
           86.5, 84.3, 65.7, 65.4, 81.5, 87.5, 74.3, 89.2, 81, 65.8, 80.1,
           58.1, 96.5, 104.7, 69.5, 59.3, 83.5, 73, 110, 73.4, 84.3, 74.4,
           62.9, 76.8, 46.9, 96, 76.3, 106.6, 80.1, 62.4, 82.2, 59, 54.2,
           77.4, 66.4, 73, 74.1, 64.2, 63.5, 70, 89.7, 49.1, 80.9, 77, 87.9,
           67.4, 77.6, 76, 62.9, 89.1, 88.4, 81.5, 94.8, 79.4, 51.9, 62.4,
           52.6, 63.9, 60.7, 70.6, 56.3, 72.4, 60.2, 96.5, 80.8, 83.7, 75.8,
           95.2),
    year = c(2000, 2000, 2000, 2000, 2000, 2000, 2000, 2000, 2000, 2000,
             2001, 2001, 2001, 2001, 2001, 2001, 2001, 2001, 2001, 2001, 2002,
             2002, 2002, 2002, 2002, 2002, 2002, 2002, 2002, 2003, 2003,
             2003, 2003, 2003, 2003, 2003, 2003, 2004, 2004, 2004,
             2005, 2005, 2005, 2005, 2005, 2005, 2006, 2006, 2006, 2006, 2006,
             2004, 2004, 2004, 2004, 2004, 2004, 2004, 2005, 2005, 2005, 2005,
             2006, 2006, 2006, 2006, 2006, 2007, 2007, 2007, 2007, 2007, 2007,
             2007, 2007, 2007, 2007, 2008, 2008, 2008, 2008, 2008, 2008, 2008,
             2008, 2008, 2008, 2009, 2009, 2009, 2009, 2009, 2009, 2009, 2009,
             2009, 2009, 2010, 2010, 2010, 2010, 2010, 2010, 2010, 2010, 2010,
             2010))
```

```r
# on peut centrer les données sur l'année 2000 pour avoir un intercept interprétable
d$yearc <- d$year - 2000
mod <- lm(nb ~ yearc, data=d)
summary(mod)
```

```
##
## Call:
## lm(formula = nb ~ yearc, data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -34.960  -9.974   0.239   8.526  32.089
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  81.7100     2.4089  33.920   <2e-16 ***
## yearc        -0.9498     0.4072  -2.333   0.0215 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.5 on 108 degrees of freedom
## Multiple R-squared:  0.04797,    Adjusted R-squared:  0.03915
## F-statistic: 5.441 on 1 and 108 DF,  p-value: 0.02152
```

```r
# valeurs prédites et se
X <- cbind(1, seq(0,10,0.1))
pred <- X %*% coef(mod)
se <- sqrt(diag(X %*% as.matrix(vcov(mod)) %*% t(X)))

# graphique
plot(d$nb ~ d$yearc, xaxt = "n", ylab = "Nombre moyen d'individus par transect",
     xlab = "Année")
axis(side = 1, at = seq(0,10,2), labels= c(2000, 2002, 2004, 2006, 2008, 2010))
abline(mod)
lines(x = X[,2], y = pred + se, col = "gray50")
lines(x = X[,2], y = pred - se, col = "gray50")
```




```
# % de variance expliquée : c'est le R² (* 100 pour l'avoir en %):
summary(mod)$r.squared *100
```

```
## [1] 4.796644
```

```
# nombre moyen estimé en 2000. Comme on a centré les années sur cette valeur,
# c'est l'intercept :
coef(mod)[1]
```

```
## (Intercept)
##       81.71
```

```
# tendance en nombre d'individus en moins par an (c'est la pente) et intervalle de
# confiance:
coef(mod)[2]
```

```
##      yearc
## -0.9498182
```

```
confint(mod, level = 0.9)[2,]
```

```
##        5 %       95 %
## -1.6253638 -0.2742726
```

```
# % de diminution sur 10 ans on se base sur les valeurs estimées et on calcule
# (mu2010-mu2000) * 100/mu2000
((c(1,10) %*% coef(mod)) - (c(1,0) %*% coef(mod))) *100 / (c(1,0) %*% coef(mod))
```

```
##           [,1]
## [1,] -11.62426
```

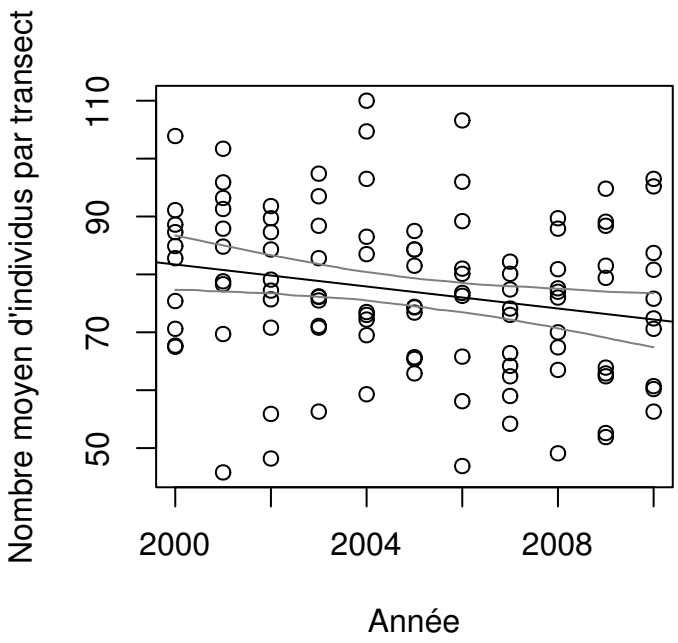
```r
# ou encore :
(c(0,10) %*% coef(mod) )*100 / c(1,0) %*% coef(mod)
```

```
##           [,1]
## [1,] -11.62426
```

```r
# Bootstrap
trendpct <- vector (length = 1000) # pour stocker les % de régression
predboot <- matrix(NA, nrow = length(seq(0,10,0.1)), ncol = 1000)
X <- cbind(1, seq(0,10,0.1)) # valeurs pour lesquelles on veut une prédiction

for (i in 1:1000) {
    boot <- d[sample(1:nrow(d), replace = TRUE),] # on rééchantillonne le dataset
    modboot <- lm(nb ~ yearc, data=boot)
    trendpct[i] <- (c(0,10) %*% coef(modboot) )*100 / c(1,0) %*% coef(modboot)
    predboot[,i] <- X %*% coef(modboot)
}
# Intervalle pour le % de régression :
quantile(trendpct, probs = c(0.025, 0.975))
```

```
##       2.5%      97.5%
## -20.666820  -2.471084
```

```r
# Intervalle pour les prédiction. On calcule les quantiles pour chaque ligne du tableau
ICboot <- apply(predboot, 1, quantile, probs = c(0.025, 0.975))

# graphique
plot(d$nb ~ d$yearc, xaxt = "n", ylab = "Nombre moyen d'individus par transect",
     xlab = "Année")
axis(side = 1, at = seq(0,10,2), labels= c(2000, 2002, 2004, 2006, 2008, 2010))
abline(mod)
lines(x = X[,2], y = ICboot[1,], col = "gray50")
lines(x = X[,2], y = ICboot[2,], col = "gray50")
```




```
# on peut aussi les calculer de manière paramétrique avec la fonction
# predict ou à la main.
predict(mod, data.frame(yearc = seq(0,10,0.1)), interval = "confidence") [1:10,]
```

```
##         fit      lwr      upr
## 1  81.71000 76.93513 86.48487
## 2  81.61502 76.90816 86.32187
## 3  81.52004 76.88079 86.15929
## 4  81.42505 76.85299 85.99712
## 5  81.33007 76.82475 85.83540
## 6  81.23509 76.79604 85.67414
## 7  81.14011 76.76685 85.51337
## 8  81.04513 76.73715 85.35311
## 9  80.95015 76.70691 85.19338
## 10 80.85516 76.67613 85.03420
```

```
pred <- X %*% coef(mod)
se <- sqrt(diag(X %*% as.matrix(vcov(mod)) %*% t(X)))
IC <- data.frame(
    fit = pred,
    lwr = pred + qt(p=c(0.025), df=mod$df.residual) *se,
    lwr = pred + qt(p=c(0.975), df=mod$df.residual) *se)
IC[1:10,]
```

```
##         fit      lwr    lwr.1
## 1  81.71000 76.93513 86.48487
## 2  81.61502 76.90816 86.32187
## 3  81.52004 76.88079 86.15929
## 4  81.42505 76.85299 85.99712
## 5  81.33007 76.82475 85.83540
## 6  81.23509 76.79604 85.67414
## 7  81.14011 76.76685 85.51337
## 8  81.04513 76.73715 85.35311
```

```
## 9  80.95015 76.70691 85.19338
## 10 80.85516 76.67613 85.03420
```

Exercice 2bis : micro analyse de puissance par simulation

Exercice subsidiaire pour ceux que la question intéresse. Pour des cas aussi simples, il existe des méthodes analytiques mais l'avantage des simulation est leur flexibilité et la possibilité de les appliquer dans pratiquement tous les cas.
On a commencé un programme de monitoring depuis 3 ans. Chaque année, on évalue l'abondance d'une espèce sur 5 sites échantillonnés au hasard.
On aimerait savoir combien de sites on devrait échantillonner chaque année pour pouvoir détecter une diminution de 3 individus par an (30 individus en 10 ans) avec une puissance de 0.8 (on détecte une pente significative dans 80 % des cas).

L'idée générale de l'analyse de puissance par simulation est la suivante : on génère des jeux de données selon un modèle qui a des caractéristiques posées par l'expérimentateur (dans notre cas la pente = -3) et d'autres estimées si possible sur base d'un jeu de données existant (dans notre cas : l'intercept et la variance résiduelle). On génère ces jeux de données un grand nombre de fois et on regarde combien de fois on détecte un effet significatif (c'est la puissance). Ensuite on recommence en faisant varier la taille d'échantillon et on examine son effet sur la puissance.

Voici comment procéder :

- Calculer la régression linéaire sur les données ci-dessous et récupérer l'intercept (alpha) et l'erreur standard des résidus (sigma). La pente beta est fixée à -3. Vous devez aussi générer les x, 10 années, avec pour commencer une seule observation par année.
- Générez un faux jeu de données avec ces paramètres, faites en l'analyse (lm) et récupérez la p valeur de la pente (dans le summary du modèle) puis stockez la.
- Recommencez l'opération un grand nombre de fois (200 fois pex)
- Recommencez à nouveau les deux points précédents mais en faisant varier le nombre de répétitions par année (de 1 à 15 devrait suffire). Idéalement les p-valeurs générées par ces simulations se stockent dans une matrice dont chaque colonne correspond à un nombre de répétitions différent.
- Finalement, comptez la proportion de p-valeurs en dessous du seuil alpha 0.05 pour chaque taille d'échantillon et regardez à partir de combien de données cette valeur devient >0.8.

```
# Génération du jeu de données
n <- 5
year <- rep(0:2, each = n)
set.seed(1)
y = round(100 - 1 * year + rnorm(length(year), 0, 30), 1)

# Stockage des données pour les présenter dans les énoncés
```

```
d <- data.frame(
    y = c(81.2, 105.5, 74.9, 147.9, 109.9, 74.4, 113.6, 121.1, 116.3,
          89.8, 143.4, 109.7, 79.4, 31.6, 131.7),
    year = rep(0:2, each = n))
```

```
mod <- lm(y ~ year, data = d)
summary(mod)
```

```
##
## Call:
## lm(formula = y ~ year, data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -68.067 -21.727   5.513  16.673  43.733
##
## Coefficients:
```

```
##             Estimate Std. Error t value   Pr(>|t|)
## (Intercept)   104.39      12.93   8.077 0.00000201 ***
## year           -2.36      10.01  -0.236      0.817
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 31.66 on 13 degrees of freedom
## Multiple R-squared:  0.004256,   Adjusted R-squared:  -0.07234
## F-statistic: 0.05557 on 1 and 13 DF,  p-value: 0.8173
```

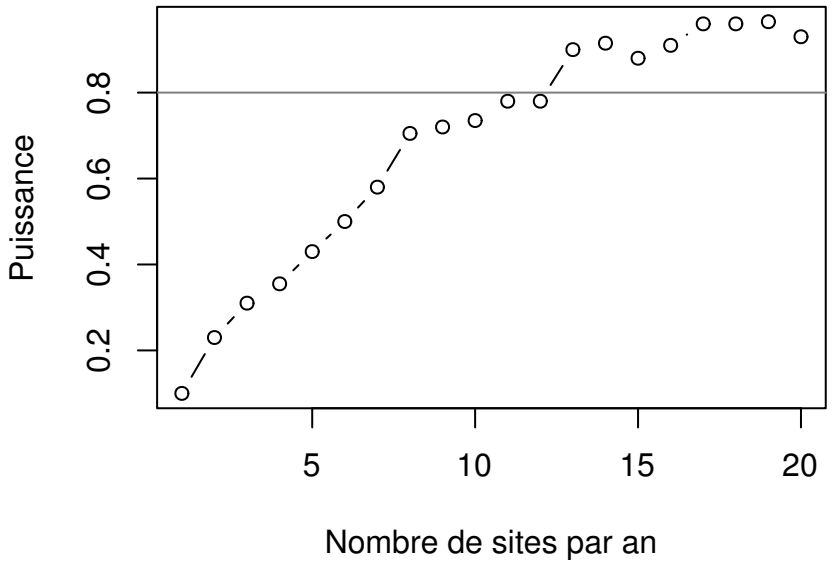
```
alpha <- coef(mod)[1]
sigma <- summary(mod)$sigma
beta <- -3

nsimul = 200 # nombre de simulations
nmax = 20 # nombre maximum de données par année

psim <- matrix(NA, nrow = nsimul, ncol = nmax) #stockage des p-valeurs des simulations

for (n in 1:nmax) {
    x <- rep(0:9,each = n)  # génère les années avec n répétitions par an

    for (i in 1:nsimul) {
        fake <- alpha + beta *x + rnorm(n*10, 0, sigma) # crée un faux dataset
        simmod <- lm(fake ~ x) # analyse du faux dataset
        psim[i,n] <- summary(simmod)$coefficients[2,4] # récupère ma p-valeur
    }
}

power <- apply(psim, 2, function(x) sum(x<0.05)/nsimul)
power
```

```
##  [1] 0.100 0.230 0.310 0.355 0.430 0.500 0.580 0.705 0.720 0.735 0.780 0.780 0.900 0.915 0.880 0.910 0.960
## [18] 0.960 0.965 0.930
```

```
# nombre de sites minimum pour avoir une puissance plus grande que 0.8
c(1:nmax)[power > 0.8][1]
```

```
## [1] 13
```

```
plot(power ~ c(1:nmax), type="b", ylab = "Puissance", xlab = "Nombre de sites par an",
     cex=0.9)
abline(h = 0.8, col = "grey50")
```


Puissance
0.2
0.4
0.6
0.8
5
10
15
20
Nombre de sites par an

Exercice 3 : ANOVA1 et comparaisons multiples

On a mesuré la diversité entomologique dans des vergers cultivés selon 5 modalités : vergers privés sans aucune intervention, vergers en permaculture, vergers bio, vergers cultivés en lutte intégrée et vergers en culture intensive. On considère que l'indice de diversité mesuré a une distribution à peu près normale.

- Réorganisez les niveaux du facteur verger de façon à les avoir dans le même ordre que dans l'énoncé
- Faites une représentation graphique des données brutes (boxplot)
- Construisez un modèle adapté à ces données, interprétez les résultats et faites une représentation graphique des données, des valeurs prédites et de leurs erreurs standard
- Faites les comparaisons multiples de tous les vergers par rapport aux vergers "privés" (similaire à un test de Dunnett).
- Faites toutes les comparaisons entre les paires de verger (similaire à un test de Tukey). Certaines comparaisons avec les vergers privés sont devenues non significatives alors qu'elles l'étaient avec le test de Dunnett. Pourquoi ? Déduisez de ces comparaisons la représentation compacte basée sur des lettres (cld). Vous pouvez les ajouter au graphe précédent en explorant le contenu de l'objet cld (fonction str) et en utilisant la fonction mtext qui permet d'ajouter du texte dans les marges.
- Faites les comparaisons multiples suivantes (en contrôlant pour le risque d'erreur global) au moyen d'une matrice de contrastes:
  - verger privé vs verger perma (-10.557)
  - verger bio vs verger intégré (3.841)
  - verger intégré vs verger intensif (9.062)
  - diversité moyenne (privé et perma) vs verger bio (14.612)
  - diversité moyenne (privé et perma) vs verger intégré (18.453)
  - diversité moyenne (privé, perma et bio) vs diversité moyenne (intégré, intensif) (18.113)

  Calculez en suite les moyennes observées pour chaque type de verger et recalculez à la main sur base de ces moyennes les comparaisons demandées ci-dessus. Vous devriez retrouver exactement les mêmes valeurs qu'avec la matrice de contrastes (par exemple en prenant la moyenne des vergers privés moins la moyenne des vergers "perma", vous devriez obtenir une différence de 10.557).

```r
# simulation du jeu de données
n = 35
B <- c(30, 12, -10, -12, -25)
sigma <- 6

set.seed(123)
x <- factor(sample(c("privé", "perma", "bio", "intégré", "intensif"), n, replace = TRUE),
            levels = c("privé", "perma", "bio", "intégré", "intensif"))
X <- model.matrix(~x)

set.seed(1234)
y <- X %*% B + rnorm(n, 0, sigma)
d <- data.frame(diversité = y, verger = x)

# Stockage des données pour les présenter dans les énoncés
```

```r
d <- data.frame(
    diversité = c(34.7576055036875, 19.664575452664, 26.5066470600983,
                  -9.07418621577609, 7.5747481328663, 33.0363353529454, 16.5515602391921,
                  1.72020886529488, 16.6132880054403, 14.6597730257354, 2.13684380147872,
                  14.0096813308418, 13.3424766321721, 20.3867529036576, 35.7569643538246,
```

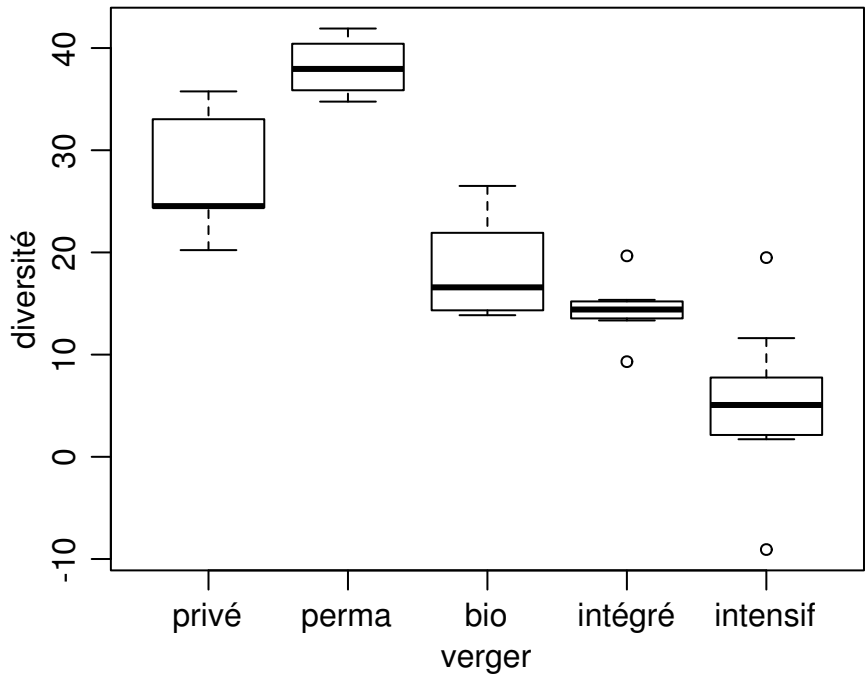
```
                  4.33828703365535, 38.9339429651601, 24.5328275002211, 36.9769699183864,
                  19.495011070936, 5.80452932091219, 15.0558846198543, 15.3567127658806,
                  7.75753664603512, 13.8376785183751, 9.31077053768117, 23.4485343254044,
                  13.8580656622167, 41.9091701978149, 24.3843083929896, 11.6137852772016,
                  2.14644152678566, 13.743359774925, 14.9924516364314, 20.2254391855278
    ),
    verger = c("perma", "intégré", "bio", "intensif", "intensif", "privé",
               "bio", "intensif", "bio", "bio", "intensif", "bio", "intégré",
               "bio", "privé", "intensif", "perma", "privé", "perma", "intensif",
               "intensif", "intégré", "intégré", "intensif", "intégré",
               "intégré", "bio", "bio", "perma", "privé", "intensif", "intensif",
               "intégré", "intégré", "privé"))
```

```
# résumé du jeu de données
summary(d)
```

```
##    diversité           verger
##  Min.   :-9.074   bio     : 8
##  1st Qu.:10.462   intégré : 8
##  Median :15.056   intensif:10
##  Mean   :17.296   perma   : 4
##  3rd Qu.:23.916   privé   : 5
##  Max.   :41.909
```

```
# On réorganise la variable "verger" de façon à avoir les vergers "privés" comme référence
# et les autres catégories classées depuis le moins intensif vers le plus intensif
d$verger <- factor(d$verger, levels = c("privé", "perma", "bio", "intégré", "intensif"))
```

```
# plot des données brutes et modèle
par(mar = c(3,3,2,1), mgp = c(1.75, 0.7, 0))
plot(diversité ~ verger, data=d, cex = 0.75)
```



```r
mod <- lm(diversité ~ verger, data=d)
summary(mod)
```

```
##
## Call:
## lm(formula = diversité ~ verger, data = d)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.4255  -3.2097  -0.6696   2.3148  14.1437
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      27.587      2.454  11.240 2.81e-12 ***
## vergerperma      10.557      3.682   2.868 0.007499 **
## vergerbio        -9.333      3.129  -2.983 0.005626 **
## vergerintégré   -13.174      3.129  -4.211 0.000213 ***
## vergerintensif  -22.236      3.006  -7.397 3.05e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.488 on 30 degrees of freedom
## Multiple R-squared:  0.8066, Adjusted R-squared:  0.7808
## F-statistic: 31.28 on 4 and 30 DF,  p-value: 2.594e-10
```

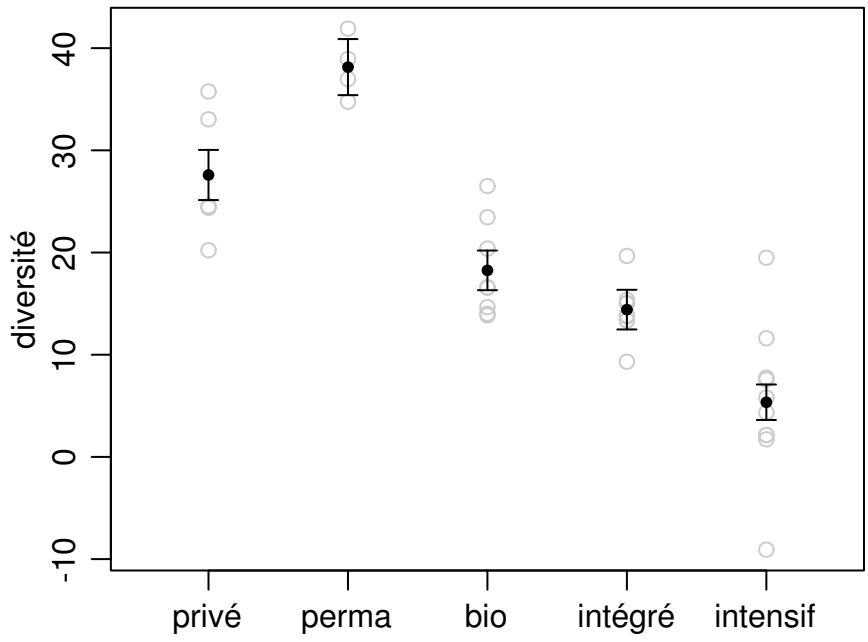
```r
# prédictions et erreurs standard des prédictions
# On pourrait écrire "à la main" la matrice X des valeurs de x pour lesquelles on veut
# une prédiction.
# On utilise ici à la place la fonction model.matrix
v <- factor(c("privé", "perma", "bio", "intégré", "intensif"),
            levels = c("privé", "perma", "bio", "intégré", "intensif"))
X <- model.matrix(~ v)
X
```

```
##   (Intercept) vperma vbio vintégré vintensif
## 1           1      0    0        0         0
## 2           1      1    0        0         0
## 3           1      0    1        0         0
## 4           1      0    0        1         0
## 5           1      0    0        0         1
## attr(,"assign")
## [1] 0 1 1 1 1
## attr(,"contrasts")
## attr(,"contrasts")$v
## [1] "contr.treatment"
```

```r
pred <- X %*% coef(mod)
V <- as.matrix(vcov(mod))
se <- sqrt(diag(X %*% V %*% t(X)))
```

```r
# graphique
par(mar = c(3,3,2,1), mgp = c(1.75, 0.7, 0))
position <- as.numeric(d$verger)-1
plot(diversité ~ position, data = d, xaxt = "n", xlim = c(-0.5, 4.5), xlab = "",
     col = "grey80")
axis(side = 1, at = sort(unique(position)), labels =
         levels(d$verger))
points(x = sort(unique(position)), y = pred, pch=20 )
```

```
arrows(x0 = sort(unique(position)), y0 = pred-se, x1 = sort(unique(position)),
       y1 = pred + se, angle=90, length = 0.05, code = 3)
```




```
# comparaisons multiples
library(multcomp)

# Dunnett
# NB : il faut avoir réorganisé les niveaux du facteur "verger" pour que la comparaison
# se fasse avec la catégorie "privé"
modmc <- glht(mod, linfct = mcp(verger = "Dunnett"))
summary(modmc)
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Dunnett Contrasts
##
##
## Fit: lm(formula = diversité ~ verger, data = d)
##
## Linear Hypotheses:
##                       Estimate Std. Error t value Pr(>|t|)
## perma - privé == 0      10.557      3.682   2.868   0.0245 *
## bio - privé == 0        -9.333      3.129  -2.983   0.0185 *
## intégré - privé == 0   -13.174      3.129  -4.211   <0.001 ***
## intensif - privé == 0  -22.236      3.006  -7.397   <0.001 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

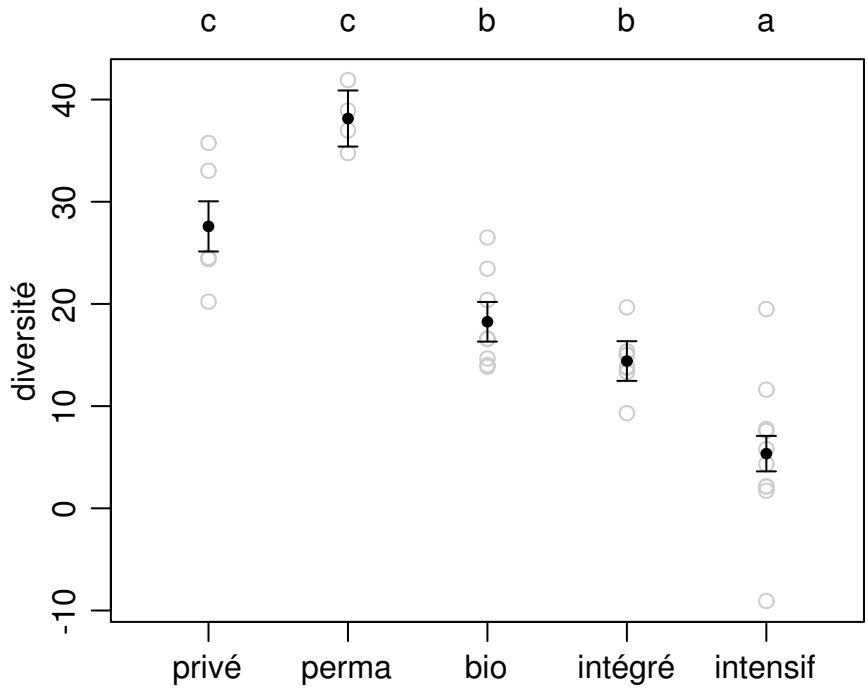
```r
# Tukey
modmc <- glht(mod, linfct = mcp(verger = "Tukey"))
summary(modmc)
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: lm(formula = diversité ~ verger, data = d)
##
## Linear Hypotheses:
##                          Estimate Std. Error t value Pr(>|t|)
## perma - privé == 0         10.557      3.682   2.868   0.0529 .
## bio - privé == 0           -9.333      3.129  -2.983   0.0408 *
## intégré - privé == 0      -13.174      3.129  -4.211   0.0018 **
## intensif - privé == 0     -22.236      3.006  -7.397   <0.001 ***
## bio - perma == 0          -19.890      3.361  -5.918   <0.001 ***
## intégré - perma == 0      -23.731      3.361  -7.061   <0.001 ***
## intensif - perma == 0     -32.793      3.247 -10.100   <0.001 ***
## intégré - bio == 0         -3.841      2.744  -1.400   0.6287
## intensif - bio == 0       -12.903      2.603  -4.956   <0.001 ***
## intensif - intégré == 0    -9.062      2.603  -3.481   0.0122 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

```r
cld(modmc)
```

```
##    privé    perma      bio  intégré intensif
##      "c"      "c"      "b"      "b"      "a"
```

```r
# cld sur le graphique (seule la dernière ligne change par rapport au code précédent)
position <- as.numeric(d$verger)-1
plot(diversité ~ position, data = d, xaxt = "n", xlim = c(-0.5, 4.5), xlab = "",
     col = "grey80")
axis(side = 1, at = sort(unique(position)), labels = levels(d$verger))
points(x = sort(unique(position)), y = pred, pch=20 )
arrows(x0 = sort(unique(position)), y0 = pred-se, x1 = sort(unique(position)),
       y1 = pred + se, angle=90, length = 0.05, code = 3)
mtext(cld(modmc)$mcletters$Letters, side = 3, line = 0.5, at = sort(unique(position)))
```




```
# comparaisons multiples choisies :
C <- rbind("privé - perma" = c(0, -1, 0, 0, 0),
           "bio - intégré" = c(0, 0, 1, -1, 0),
           "intégré - intensif" = c(0, 0, 0, 1, -1),
           "1/2*(privé + perma) - bio" = c(0, 0.5, -1, 0, 0),
           "1/2*(privé + perma) - intégré" = c(0, 0.5, 0, -1, 0),
           "1/3(privé+perma+bio)-1/2(intégré+intensif)" = c(0, 1/3, 1/3, -1/2, -1/2))

modmc <- glht(mod, linfct = C)
summary(modmc)
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Fit: lm(formula = diversité ~ verger, data = d)
##
## Linear Hypotheses:
##                                                  Estimate Std. Error t value Pr(>|t|)
## privé - perma == 0                                -10.557      3.682  -2.868   0.0365 *
## bio - intégré == 0                                  3.841      2.744   1.400   0.5523
## intégré - intensif == 0                             9.062      2.603   3.481   0.0079 **
## 1/2*(privé + perma) - bio == 0                     14.612      2.675   5.463   <0.001 ***
## 1/2*(privé + perma) - intégré == 0                 18.453      2.675   6.899   <0.001 ***
## 1/3(privé+perma+bio)-1/2(intégré+intensif) == 0    18.113      1.902   9.522   <0.001 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

```
# recalcul à partir des valeurs observées :
aggregate(d$diversité, d["verger"], mean)
```

```
##     verger        x
## 1    privé 27.587175
```

```
## 2    perma 38.144422
## 3      bio 18.254288
## 4  intégré 14.412989
## 5 intensif  5.351321
```

```
27.587175 - 38.144422
```

```
## [1] -10.55725
```

```
18.254288 -  14.412989
```

```
## [1] 3.841299
```

```
14.412989 -  5.351321
```

```
## [1] 9.061668
```

```
1/2 * (27.587175 + 38.144422) - 18.254288
```

```
## [1] 14.61151
```

```
1/2 * (27.587175 + 38.144422) - 14.412989
```

```
## [1] 18.45281
```

```
1/3 * (27.587175 + 38.144422 + 18.254288) - (1/2 * (14.412989 +  5.351321))
```

```
## [1] 18.11314
```

Exercice 4 : 2 variables explicatives qualitatives (ANOVA2 sans interaction)

On veut étudier comment réagi un ravageur à 3 techniques culturales. Cinq champs ont été sélectionnés, divisés en 3 parties et chaque type de technique culturale (A, B, C) a été attribuée à une des parcelles au hasard. On a mesuré ensuite le niveau de dégâts en surface attaquée en m^2 sur une zone de $100m^2$ située au centre de chaque parcelle. Cependant, l'essai a raté sur certaines parcelles et on a donc quelques valeurs manquantes (NA).

Sans tenir compte de l'effet site

- Faites une représentation graphique des données brutes en fonction des pratiques culturales sans tenir compte des sites (pex un boxplot)
- Comparez l'effet des techniques culturales sur le niveau de dégats (sans tenir compte des sites) au moyen d'un modèle linéaire (ANOVA).
- Faites une représentation graphique des données brutes, des valeurs prédites et de leur erreur standard.
- Interprétez les résultats grâce au graphique et aux sorties du modèle.
- On voudrait effectuer des comparaisons multiples pour évaluer quelle technique culturale a un effet différent des autres. Est-ce que çà a un sens ici ?
- Calculez les dégâts moyens pour chaque type de technique culturale et comparez les aux valeurs prédites par le modèle.

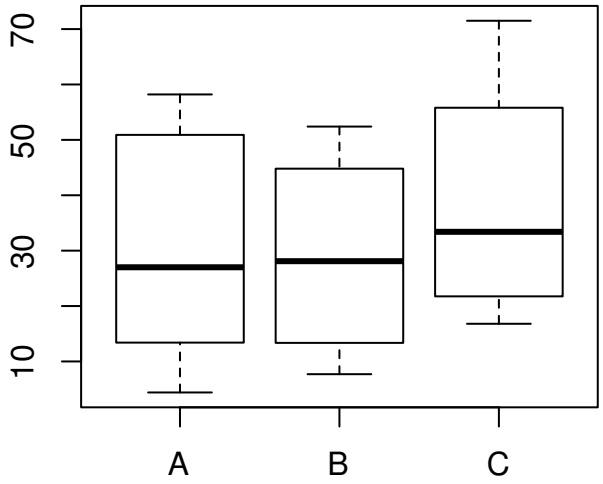
En tenant compte de l'effet site

- Essayez de faire une représentation graphique des données brutes montrant les valeurs pour chaque site et chaque technique culturale. Vous pouvez utiliser par exemple la fonction boxplot (syntaxe utilisant les formules). Vous pouvez aussi faire un graphique plus complexe en collant les facteurs "site" et "cult".
- Comparez l'effet des techniques culturales sur le niveau de dégâts *en tenant compte des sites* au moyen d'un modèle linéaire (ANOVA2).
- Faites une représentation graphique des données brutes, des valeurs prédites et de leur erreur standard, pour chaque type de pratique culturale et indépendamment du site (pour un site moyen).
- Interprétez les résultats grâce au graphique et aux sorties du modèle.
- Effectuez des comparaisons multiples pour évaluer quelle technique culturale a un effet différent des autres, indépendamment du site?
- Calculez les dégâts moyens pour chaque type de technique culturale et comparez les aux valeurs prédites par le modèle. Elles sont différentes, pourquoi ?

```r
# génération du jeu de données
nsites <- 5
ntrait <- 3

site <- as.factor(rep(1:nsites, each = ntrait))
cult <- rep(c("A", "B", "C"), nsites)
X <- model.matrix(~ cult + site)

B <- c(30, 4, 12, 30, -25, 20, -15)
set.seed(12)
y <- round(X %*% B + rnorm(nsites*ntrait , 0, 2),1)
is.na(y[c(5,12)]) <- TRUE

# Stockage des données pour les présenter dans les énoncés
```

```r
d <- data.frame(
    degats = c(27, 37.2, 40.1, 58.2, NA, 71.5, 4.4, 7.7, 16.8, 50.9, 52.4, NA, 13.4,
               19, 26.7),
    site = c("1", "1", "1", "2", "2", "2", "3", "3", "3", "4", "4", "4", "5", "5", "5"),
    cult = c("A", "B", "C", "A", "B", "C", "A", "B", "C", "A", "B", "C",  "A", "B", "C"))
```

```r
# représentation des données brutes
boxplot(degats ~ cult, data=d)
```



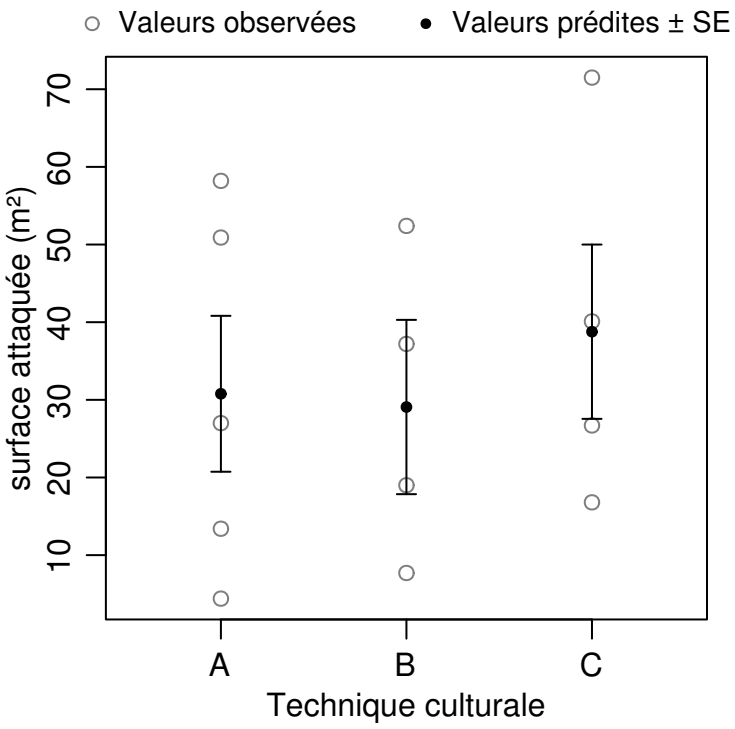

```r
# Sans tenir compte de l'effet site
# ********************************

# modèle linéaire simple et test global de l'effet "cult" (fonction Anova)
mod <- lm(degats ~ cult, data=d)
summary(mod)
```

```
##
## Call:
## lm(formula = degats ~ cult, data = d)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -26.38 -17.38  -3.78  20.12  32.73
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   30.780     10.039   3.066   0.0119 *
## cultB         -1.705     15.058  -0.113   0.9121
## cultC          7.995     15.058   0.531   0.6070
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 22.45 on 10 degrees of freedom
##   (2 observations deleted due to missingness)
## Multiple R-squared:  0.04158,    Adjusted R-squared:  -0.1501
## F-statistic: 0.2169 on 2 and 10 DF,  p-value: 0.8087
```

```r
library(car)
Anova(mod)
```

```
## Anova Table (Type II tests)
##
## Response: degats
##           Sum Sq Df F value Pr(>F)
## cult       218.6  2  0.2169 0.8087
## Residuals 5038.8 10
```

```r
# valeurs prédites et représentation graphique
X <- rbind(A = c(1,0,0),
           B = c(1,1,0),
           C = c(1,0,1))
pred <- X %*% coef(mod)
se <- sqrt(diag(X %*% vcov(mod) %*% t(X)))

par(mar = c(3,3,2,1), mgp = c(1.75, 0.7, 0))
plot(y = d$degats, x = as.numeric(d$cult), xaxt= "n", xlim = c(0.5, 3.5) , col = "grey50",
     ylab = "surface attaquée (m²)", xlab = "Technique culturale")
axis(side=1, labels = levels(d$cult), at = c(1,2,3))
points(x = 1:3, y = pred, pch = 20)
arrows(x0 = 1:3, y0 = pred-se, x1 = 1:3, y1 = pred + se,
       angle=90, length = 0.05, code = 3)
legend("top", xpd = TRUE, inset = -0.12, bty = "n", cex = 0.9,, horiz = TRUE,
       pch = c(1,20), col = c("grey50", "black"),
       legend =  c("Valeurs observées", "Valeurs prédites ± SE"))
```




```r
# Conclusion : Les différentes techniques culturales semblent avoir des niveaux moyens
# d'infestation différents mais la variabilité dans les données brutes est très grande
# et on aurait très bien pu obtenir un tel résultat par hasard (p = 0.8087).
```

```
# Le test de l'effet global "cult" est non significatif (p = 0.8087), ce qui signifie qu'
# aucune des pratique culturale n'a un effet significativement différent des autres.
# Il est donc inutile ici d'effectuer des comparaisons multiples.
# Le modèle estime que le niveau d'infestation est plus faibles de 1.7 m² avec
# la pratique culturale B par rapport à la B mais l'erreur standard est trop élevée
# (15.06 m²) et l'effet pourrait très bien être positif (c'est le cas en réalité car
# dans ce jeu de données simulé, la différence B-A est de +4 m²)
```

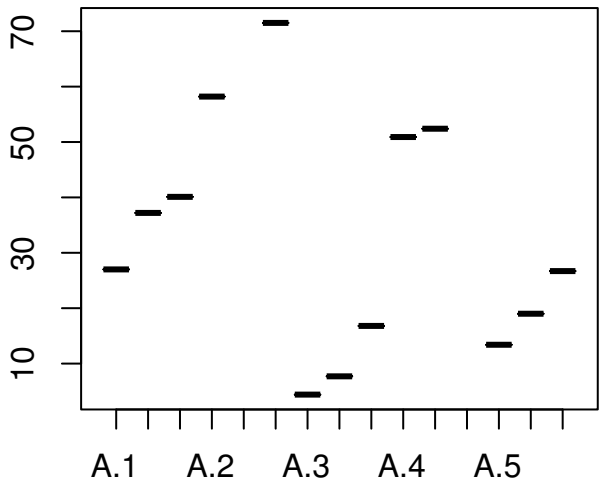
```
# On obtient bien les mêmes valeurs avec les moyennes par technique culturale et les
# valeurs prédites
aggregate(d$degats, d["cult"], mean, na.rm = TRUE)
```

```
##   cult      x
## 1    A 30.780
## 2    B 29.075
## 3    C 38.775
```

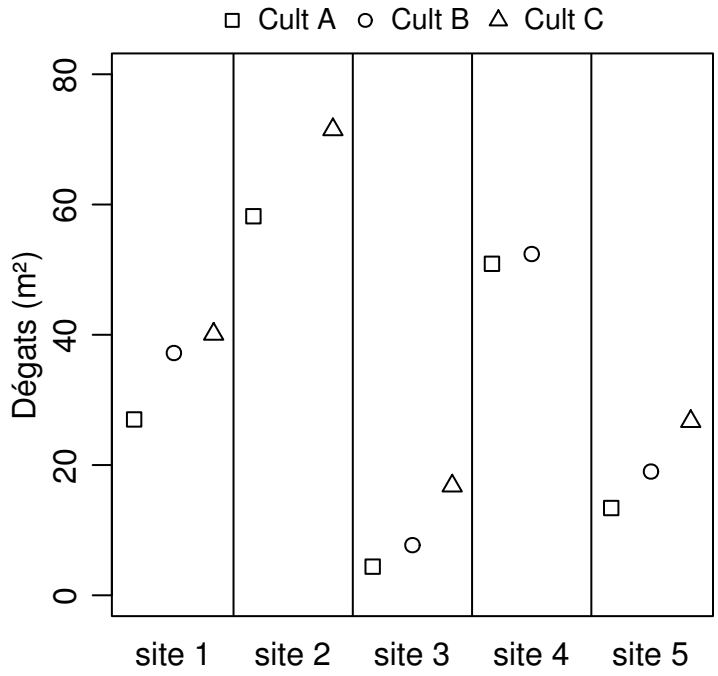
```
pred
```

```
##     [,1]
## A 30.780
## B 29.075
## C 38.775
```

```
# En tenant compte de l'effet site
# ********************************
# Graphique, version simple et rapide :
boxplot(y ~ cult + site)
```

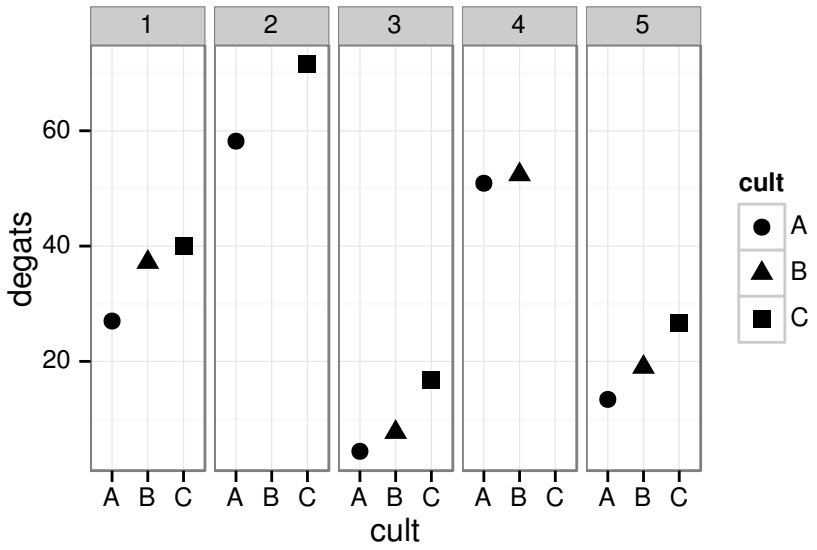


```r
# version plus construite mais moins rapide ! ...
par( mar = c(3,3,2,1), mgp = c(1.75, 0.7, 0))
id <- as.factor(paste(d$site, d$cult, sep = ""))
plot(d$degats ~ as.numeric(id), xlab = "", ylab = "Dégats (m²)", xaxt="n",
     pch = 0:2, ylim = c(0,80))
mtext(side=1, text = paste("site", 1:5), at = c(2,5,8,11,14), line = 0.5)
abline(v = c(3.5, 6.5, 9.5, 12.5))
legend("top", xpd = TRUE, inset = -0.12, bty = "n", cex = 0.9,, horiz = TRUE,
       pch = 0:2, legend =  c("Cult A", "Cult B", "Cult C"))
```




```r
# Le package ggplot (ou lattice) est particulièrement bien adapté pour ce genre de cas
```

```r
library(ggplot2)
ggplot(d) +
    geom_point(aes(y = degats, x = cult, shape = cult), size = 3) +
    facet_grid(.~ site) + theme_bw()
```




```
# Modèle
# Après avoir contrôlé l'effet site, l'effet "pratique culturale" devient significatif
# (p = 0.000333), ce qui veut dire qu'au moins une des pratique a un effet
# significativement différent des autres sur le niveau de dégât.
# Le modèle estime qu'avec la pratique culturale B le niveau d'infestation est plus élevé
# de 5.24 m² que la pratique A, et de 12.5m² pour la pratique C.
mod <- lm(degats ~ cult + site, data = d)
summary(mod)
```

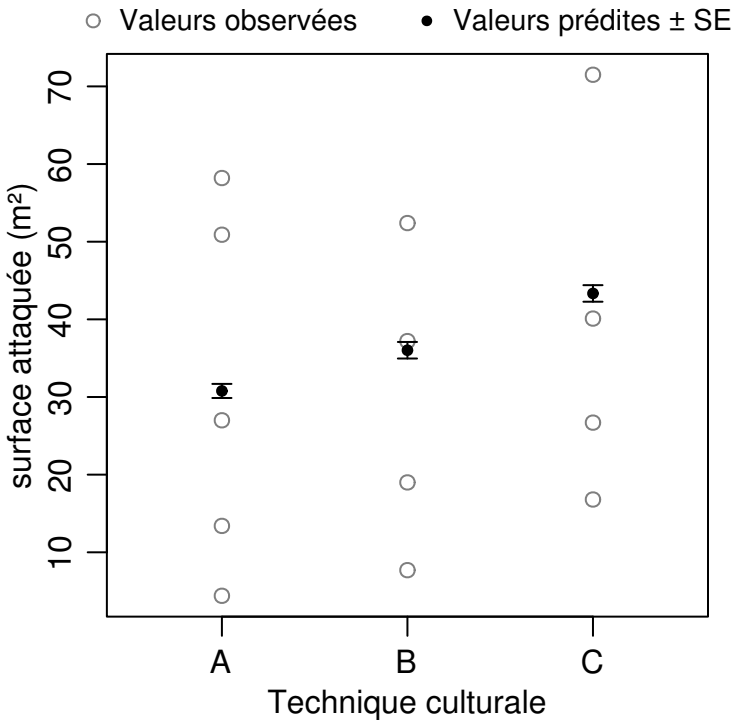
```
##
## Call:
## lm(formula = degats ~ cult + site, data = d)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.87143 -1.24286 -0.00952  0.54286  3.12381
##
## Coefficients:
##             Estimate Std. Error t value    Pr(>|t|)
## (Intercept)   28.833      1.407  20.492 0.000000878 ***
## cultB          5.243      1.399   3.747    0.009545 **
## cultC         12.557      1.399   8.974    0.000107 ***
## site2         29.738      1.900  15.649 0.000004314 ***
## site3        -25.133      1.656 -15.180 0.000005156 ***
## site4         20.195      1.900  10.627 0.000040905 ***
## site5        -15.067      1.656  -9.100 0.000098899 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.028 on 6 degrees of freedom
##   (2 observations deleted due to missingness)
## Multiple R-squared:  0.9953, Adjusted R-squared:  0.9906
## F-statistic: 212.1 on 6 and 6 DF,  p-value: 0.000001026
```

```
Anova(mod)
```

```
## Anova Table (Type II tests)
##
```

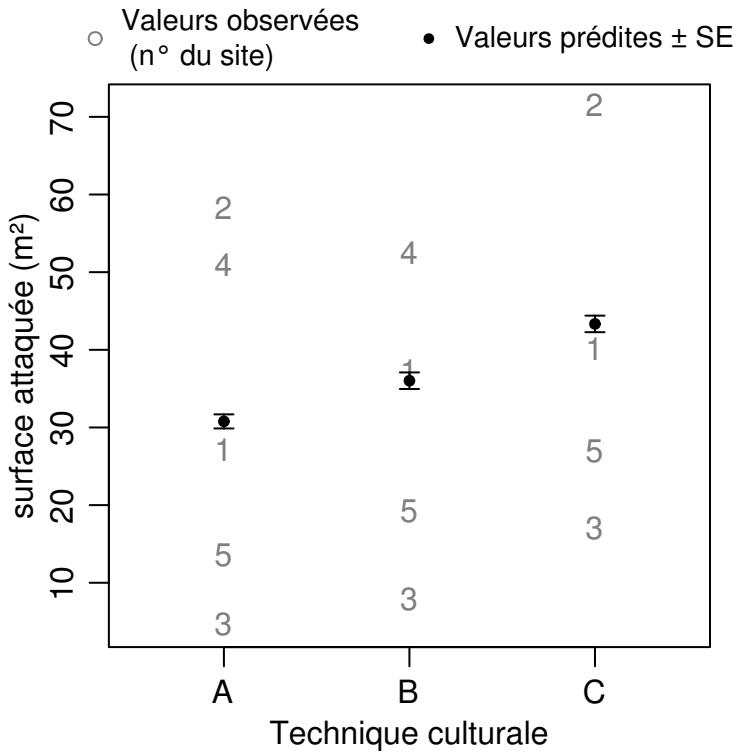
```
## Response: degats
##           Sum Sq Df F value       Pr(>F)
## cult       331.3  2  40.281     0.000333 ***
## site      5014.1  4 304.865 0.0000004678 ***
## Residuals   24.7  6
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Valeurs prédites et représentation graphique
# On donne un poids égal de 1/5 à chaque site. Si on donnait la valeur 0 pour chaque site
# on estimerait les valeurs prédites pour le premier site.
# On voit bien sur le graphique que les erreur standard des prédictions sont beaucoup
# plus petites que lorsqu'on ne contrôle pas pour l'effet"site".
X <- rbind(A = c(1,0,0, 1/5, 1/5, 1/5, 1/5),
           B = c(1,1,0, 1/5, 1/5, 1/5, 1/5),
           C = c(1,0,1, 1/5, 1/5, 1/5, 1/5))
pred <- X %*% coef(mod)
se <- sqrt(diag(X %*% vcov(mod) %*% t(X)))

par(mar = c(3,3,2,1), mgp = c(1.75, 0.7, 0))
plot(y = d$degats, x = as.numeric(d$cult), xaxt= "n", xlim = c(0.5, 3.5) , col = "grey50",
     ylab = "surface attaquée (m²)", xlab = "Technique culturale")
axis(side=1, labels = levels(d$cult), at = c(1,2,3))
points(x = 1:3, y = pred, pch = 20)
arrows(x0 = 1:3, y0 = pred-se, x1 = 1:3, y1 = pred + se,
       angle=90, length = 0.05, code = 3)
legend("top", xpd = TRUE, inset = -0.12, bty = "n", cex = 0.9,, horiz = TRUE,
       pch = c(1,20), col = c("grey50", "black"),
       legend =  c("Valeurs observées", "Valeurs prédites ± SE"))
```




```r
# Sur le graphique précédent, les erreurs standard peuvent paraître démesurément petites
# par rapport à la variabilité des valeurs observées. Il ne faut pas oublier que les
```

```r
# erreurs standard sont estimées sur base de données dont on a éliminé une bonne partie de
# la variation dûe à l'effet site. A la place des points observés, on peur représenter les
# numéros des sites ce qui permet de visualiser que les différences entre pratiques
# culturales au sein d'un site sont similaires quelque soit le site.
par(mar = c(3,3,2,1), mgp = c(1.75, 0.7, 0))
plot(y = d$degats, x = as.numeric(d$cult), xaxt= "n", xlim = c(0.5, 3.5) , col = "grey50",
     ylab = "surface attaquée (m²)", xlab = "Technique culturale", pch = as.character(d$site))
axis(side=1, labels = levels(d$cult), at = c(1,2,3))
points(x = 1:3, y = pred, pch = 20)
arrows(x0 = 1:3, y0 = pred-se, x1 = 1:3, y1 = pred + se,
       angle=90, length = 0.05, code = 3)
legend("top", xpd = TRUE, inset = -0.18, bty = "n", cex = 0.9,, horiz = TRUE,
       pch = c(1,20), col = c("grey50", "black"),
       legend =  c("Valeurs observées\n (n° du site)", "Valeurs prédites ± SE"))
```




```r
# comparaisons multiples
# A est significativement différent de B et de C et B est significativement différent de C
C <- rbind ("B - A" = c(1, 1, 0, 1/5, 1/5, 1/5, 1/5) - c(1, 0, 0, 1/5, 1/5, 1/5, 1/5),
            "C - A" = c(1, 0, 1, 1/5, 1/5, 1/5, 1/5) - c(1, 0, 0, 1/5, 1/5, 1/5, 1/5),
            "C - B" = c(1, 0, 1, 1/5, 1/5, 1/5, 1/5) - c(1, 1, 0, 1/5, 1/5, 1/5, 1/5))
library(multcomp)
modmc <- glht(mod, linfct = C)
summary(modmc)
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Fit: lm(formula = degats ~ cult + site, data = d)
##
## Linear Hypotheses:
##            Estimate Std. Error t value Pr(>|t|)
## B - A == 0    5.243      1.399   3.747  0.02212 *
```

```
## C - A == 0   12.557      1.399   8.974  < 0.001 ***
## C - B == 0    7.314      1.533   4.772  0.00733 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

```
# On obtient des valeurs différentes entre les moyennes calculées et les valeurs prédites.
# Les valeurs prédites sont des meilleurs estimateurs car il corrigent pour l'effet site
# avant d'estimer la moyenne. Les vraies valeurs de ce jeu de données simulées sont
# en effet : 30, 34 et 42
aggregate(d$degats, d["cult"], mean, na.rm = TRUE)
```

```
##   cult      x
## 1    A 30.780
## 2    B 29.075
## 3    C 38.775
```

```
pred
```

```
##       [,1]
## A 30.78000
## B 36.02286
## C 43.33714
```

```
# NB : on pourrait aussi donner à chaque site un poids proportionnel aux nombres de
# données observées dans chaque site.
Beta <- apply(model.matrix(mod), 2, mean) [4:7]
X <- rbind(A = c(1,0,0, Beta[1], Beta[2], Beta[3], Beta[4]),
           B = c(1,1,0, Beta[1], Beta[2], Beta[3], Beta[4]),
           C = c(1,0,1, Beta[1], Beta[2], Beta[3], Beta[4]))
X %*% coef(mod)
```

```
##       [,1]
## A 27.23846
## B 32.48132
## C 39.79560
```

Exercice 5 : régression multiple

Suite à l'expérience précédente (ex 4), on aimerait bien comprendre quels sont les facteurs qui expliquent les différences entre les sites. On suspecte que le paysage autour du champ peut influencer le niveau de dégât. On a donc sélectionné aléatoirement 30 champs, présentant des pratiques culturales similaires, on a mesuré au centre de chacun la surface attaquée sur un carré de 10 x 10 m. On a ensuite mesuré la surface des différent types d'occupations du sol que l'on pense pouvoir influencer le niveau de dégâts dans un rayon de 2km autour du centroïde du champ : pommes de terre (pdt), céréales, betterave, colza, prairies, maïs, forêts, autres sites semi-naturels.

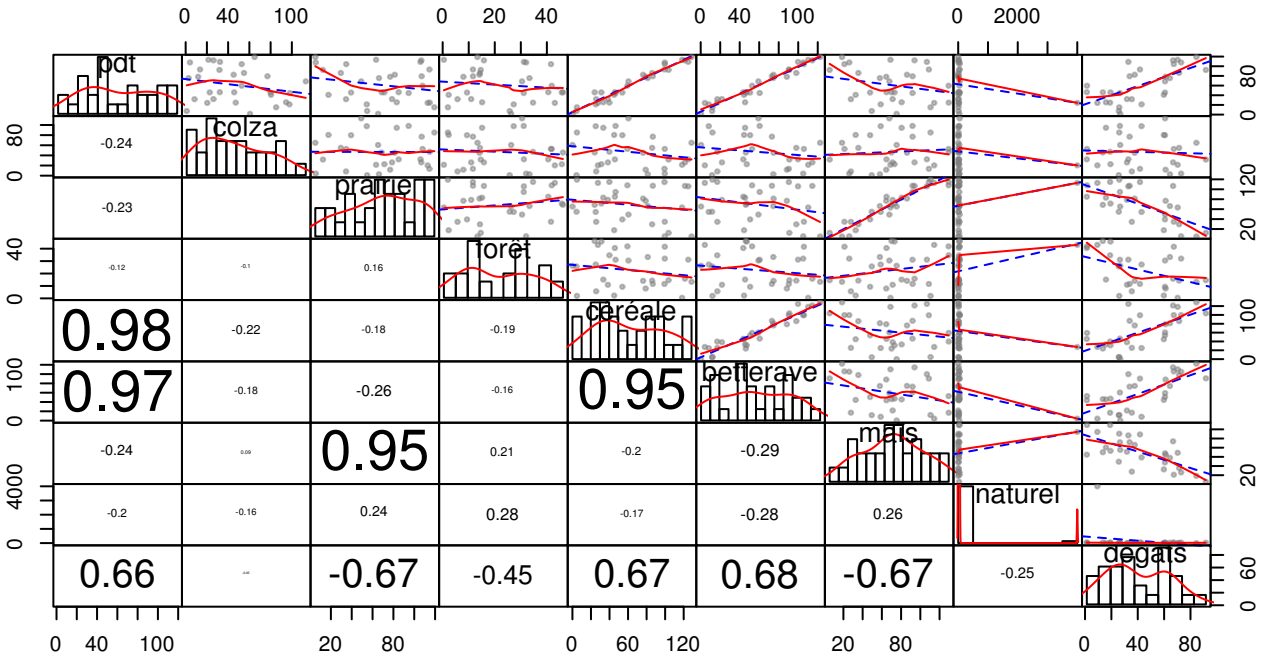
- Explorez le jeu de données. Vous pouvez utiliser la fonction pairs2 fournie dans le script "mytoolbox.R" (utilisez la fonction source pour charger le contenu du script en mémoire). Est-ce que toutes les données sont OK ? Si certaines valeurs vous semblent impossibles remplacez les par des valeurs manquantes.
- Examinez les corrélations entre les variables explicatives. Calculez également les VIFs d'un modèle contenant toutes les variables explicatives. Que constatez-vous ? Quelle solutions pourriez-vous adopter ?
- Une fois les dispositions prises pour éviter des problèmes potentiels, construisez un modèle prédictif du niveau de dégâts en fonction des variables paysagères et interprétez les sorties du modèle.
- Quelle est la variable explicative qui a l'effet le plus important en termes biologique/agronomique sur le niveau de dégât ? Standardisez les variables explicatives (enlever la moyenne, diviser par l'écart type, ou utiliser la fonction scale) et estimez à nouveau le modèle. Interprétez les résultats.
- Pour chaque variable explicative faites un graphique représentant les données et les valeurs prédites en contrôlant pour les autres variables explicatives.
- Comparez le R^2 et le R^2 ajusté. Sont-ils très différents ? Pourquoi ?
- Quel est le % de variance additionnel expliqué par chaque variable explicative ? (pour chaque variable estimer un modèle sans cette variable et faire la différence avec le R^2 du modèle complet)

```
# génartion du jeu de données
n = 30
set.seed(1)
pdt <- runif(n, 0,120)
set.seed(12)
colza <- runif(n, 0,120)
set.seed(123)
prairie <- runif(n, 0,120)
set.seed(1234)
forêt <- runif(n, 0,50)

set.seed(1)
céréale <- pdt + rnorm(n, sd = 10)
set.seed(12)
betterave <- pdt + rnorm(n, sd = 10)

set.seed(123)
maïs <- prairie + rnorm(n, sd = 12)

set.seed(1234)
naturel <- forêt + rnorm(n, sd = 15)

set.seed(2)
dégats <- 50 + 0.3 * céréale - 0.35 * prairie - 0.20 * forêt + rnorm(n, sd = 10)

# erreur d'encodage ...
naturel[5] <- 4000

d <- abs(round(data.frame(pdt, colza, prairie, forêt, céréale, betterave, maïs, naturel,
                          dégats),2))

# Stockage du jeu de données pour les énoncés :
```

```
    d <- structure(list(pdt = c(31.86, 44.65, 68.74, 108.98, 24.2, 107.81,
    113.36, 79.3, 75.49, 7.41, 24.72, 21.19, 82.44, 46.09, 92.38,
    59.72, 86.11, 119.03, 45.6, 93.29, 112.16, 25.46, 78.2, 15.07,
    32.07, 46.33, 1.61, 45.89, 104.36, 40.84), colza = c(8.32, 98.13,
    113.11, 32.33, 20.32, 4.07, 21.45, 77, 2.75, 1, 47.12, 97.67,
    45.15, 45.7, 31.79, 52.72, 54.91, 64.88, 79.88, 13.52, 26.2,
    94.54, 11.74, 85.18, 26.14, 32.15, 60.57, 22.63, 52.73, 80.38
    ), prairie = c(34.51, 94.6, 49.08, 105.96, 112.86, 5.47, 63.37,
    107.09, 66.17, 54.79, 114.82, 54.4, 81.31, 68.72, 12.35, 107.98,
    29.53, 5.05, 39.35, 114.54, 106.74, 83.14, 76.86, 119.31, 78.68,
    85.02, 65.29, 71.3, 34.7, 17.65), forêt = c(5.69, 31.11, 30.46,
    31.17, 43.05, 32.02, 0.47, 11.63, 33.3, 25.71, 34.68, 27.25,
    14.14, 46.17, 14.62, 41.86, 14.31, 13.34, 9.34, 11.61, 15.83,
    15.13, 7.95, 2, 10.94, 40.53, 26.28, 45.73, 41.57, 2.29), céréale = c(25.6,
    46.49, 60.39, 124.94, 27.5, 99.6, 118.24, 86.68, 81.25, 4.36,
    39.83, 25.09, 76.23, 23.95, 103.63, 59.27, 85.95, 128.47, 53.82,
    99.23, 121.35, 33.28, 78.95, 4.83, 38.26, 45.77, 0.05, 31.18,
    99.58, 45.02), betterave = c(17.06, 60.43, 59.17, 99.78, 4.23,
    105.08, 110.21, 73.01, 74.43, 11.69, 16.94, 8.25, 74.65, 46.21,
    90.86, 52.69, 98, 122.43, 50.67, 90.36, 114.4, 45.53, 88.32,
    12.04, 21.81, 43.66, 0.38, 47.2, 105.82, 44.46), maïs = c(27.78,
    91.83, 67.78, 106.81, 114.41, 26.05, 68.9, 91.91, 57.93, 49.45,
    129.51, 58.72, 86.12, 70.04, 5.68, 129.42, 35.5, 18.55, 47.77,
    108.87, 93.93, 80.52, 64.55, 110.57, 71.18, 64.78, 75.34, 73.14,
    21.04, 32.7), naturel = c(12.42, 35.28, 46.73, 4.02, 4000, 39.61,
    8.15, 3.43, 24.84, 12.36, 27.52, 12.27, 2.49, 47.14, 29.01, 40.21,
    6.65, 0.33, 3.22, 47.85, 17.84, 7.77, 1.34, 8.89, 0.53, 18.81,
    34.91, 30.38, 41.34, 11.75), dégats = c(35.49, 26.46, 60.72,
    32.86, 9.34, 72.89, 70.27, 33.8, 64.4, 25.6, 19, 42.85, 37.66,
    13.5, 91.67, 1.49, 71.37, 84.46, 60.63, 41.68, 66.79, 15.86,
    61.09, 25.94, 31.8, 1.35, 26.68, 19.29, 67.34, 59.77)), .Names = c("pdt",
    "colza", "prairie", "forêt", "céréale", "betterave", "maïs",
    "naturel", "dégats"), row.names = c(NA, -30L), class = "data.frame")
```

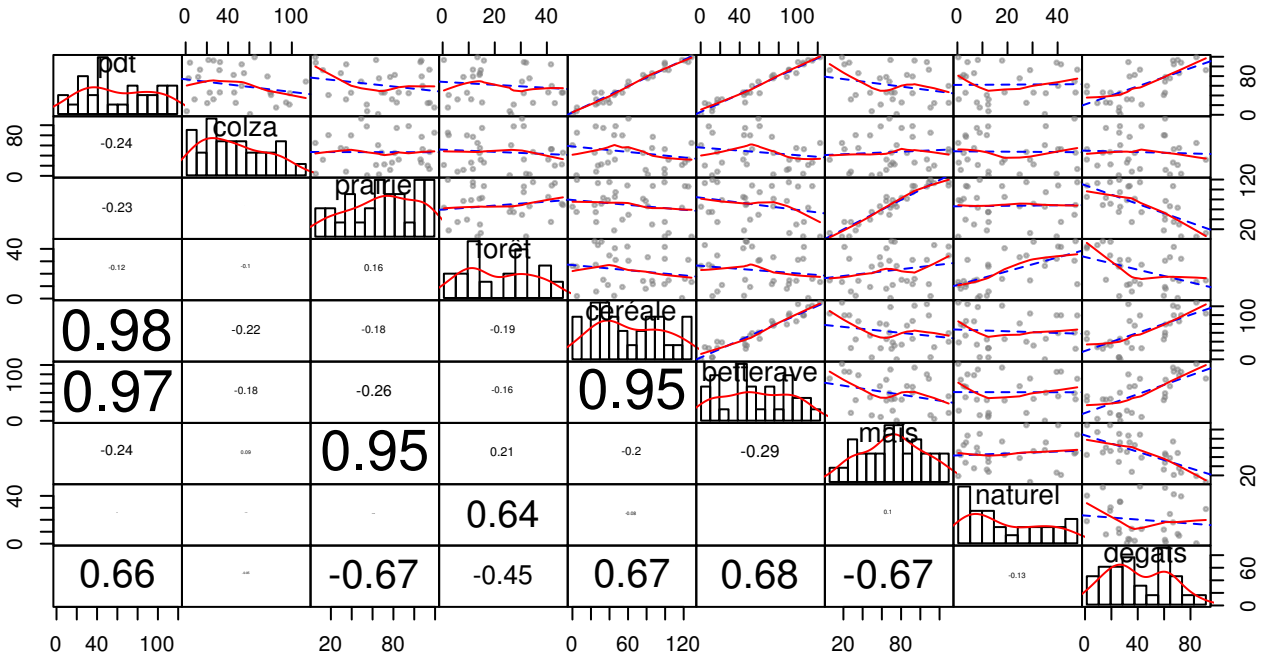
```
# On visualise les données dans un scatterplot matrix.
# On charge d'abbord en mémoire les fonctions qui se trouvent dans le fichier mytoolbox.R
# et qui contient notamment la fonction pairs2.
source("/home/gilles/stats/mytoolbox.R")
```

```
pairs2(d, gap = 0, pt.cex = 0.7)
```




```
# On remarque immédiatement qu'il y a un problème avec la variable "naturel" dont une des
# valeurs est 4000 ha ce qui est impossible dans un rayon de 2km. On remplace donc sa
# valeur par une valeur manquante (NA)
d[d$naturel > 100, "naturel"] <- NA
```

```
pairs2(d, gap = 0, pt.cex = 0.7)
```

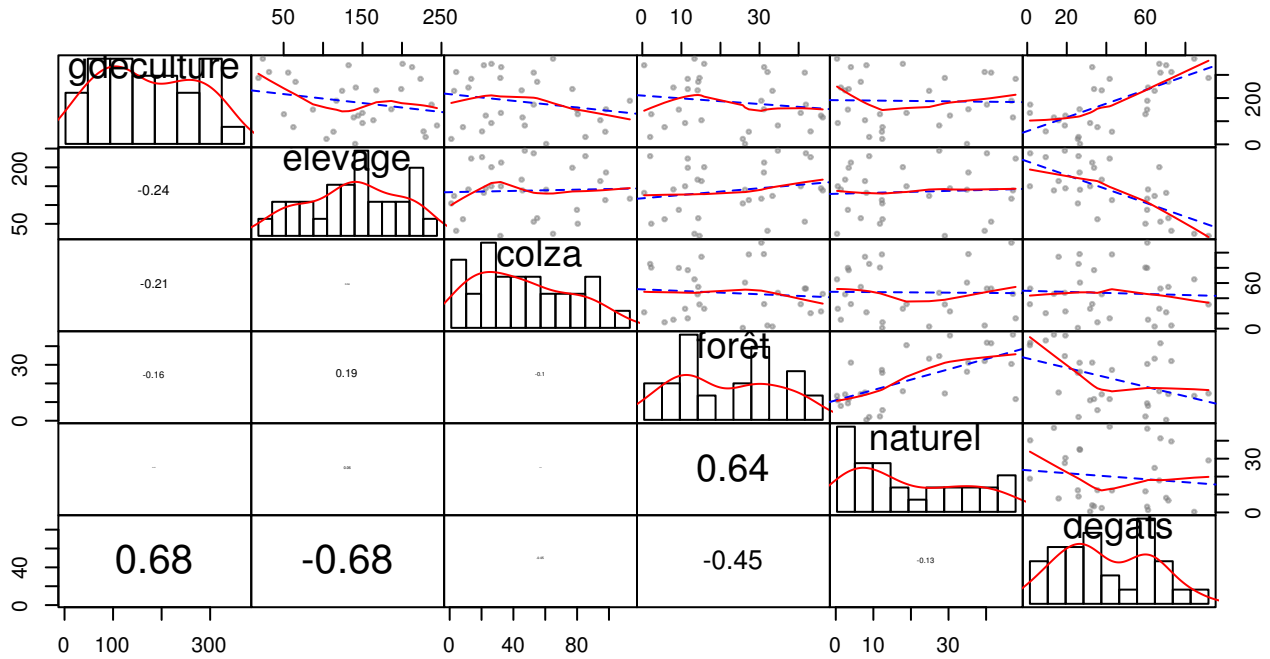


```
# On constate ensuite qu'il y a de fortes corrélations entre certaines variables
# explicatives. Typiquement 3 grandes cultures sont fortement corrélées : pomme de terre,
# céréales, et betterave. Deux types d'occupations du sol liés à l'élevage sont également
# corrélés : prairies et maïs.  Les forêts et les site naturels sont corrélés dans une
# moindre mesure.
# Ces corrélations transparaissent également dans les VIFs du modèle (Variance Inflation
# factors). On voit par exemple que l'erreur standard de pdt sera multipliée par la racine
# carrée de 41.45 à cause du fait que cette variable est corrélée à une ou plusieurs
# autres variables.

mod <- lm(dégats ~ pdt + céréale + betterave + colza + prairie + maïs + forêt +
             naturel, data = d)
library(car)
vif(mod) # dans le package "car"
```

```
##       pdt   céréale betterave     colza   prairie      maïs     forêt   naturel
## 46.464280 26.696092 21.788113  1.287041 10.961256 11.538824  1.758540  1.952520
```

```
# On décide donc de regrouper une partie des variables. Le choix de regrouer ou d'éliminer
# certaines variables ou pas doit se faire en fonction du problème étudié et des
# questions qu'on se pose. Ici par exemple on décide de garder "forêt" et "naturel"
# séparés car leur corrélation n'est pas trop trop forte et on veut estimer l'effet des
# forêts indépendamment des sites naturels et vice-versa. Ce faisant on accepte de perdre
# un peu de précision dans les estimations des paramètres de ces variables au risque
# qu'ils ne soient plus significatifs. Par contre on va regrouper les autres variables
# en deux nouvelles variables.
d$gdeculture <- d$pdt + d$céréale + d$betterave
d$élevage <- d$maïs + d$prairie
```

```
pairs2(d[, c("gdeculture", "élevage", "colza", "forêt", "naturel", "dégats")],
       gap = 0, pt.cex = 0.7)
```



```r
# Voici donc le modèle final.
# L'intercept estime qu'on a en moyenne 54.22 m² de dégâts pour un champs qui serait
# situé dans une zone avec 0 ha de grande culture, d'élevage, de colza, de forêt et de
# sites naturels... Pas très réaliste donc. Pour avoir une valeur plus informative, il
# faudrait centrer les variables explicatives. L'intercept représenterait alors le niveau
# moyen d'infestation sur l'ensemble des champs.
# Trois coefficients sont très significativement différents de 0 : lorsque la surface de
# grande culture augmente d'1 ha, les dégâts augmentent de 0.12 m², ils diminuent de
# 0.19 m² pour 1ha d'élevage et ils diminuent de 0.67 m² pour 1ha de forêt.
mod <- lm(dégats ~ gdeculture + élevage + colza + forêt + naturel, data = d)
vif(mod)
```

```
## gdeculture    élevage      colza      forêt    naturel
##   1.121489   1.055114   1.081941   1.743416   1.695313
```

```r
summary(mod)
```

```
##
## Call:
## lm(formula = dégats ~ gdeculture + élevage + colza + forêt +
##     naturel, data = d)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -19.5135  -6.0291  -0.1155   6.7859  21.9874
##
## Coefficients:
##             Estimate Std. Error t value   Pr(>|t|)
## (Intercept) 54.22019    8.52335   6.361 0.00000172 ***
## gdeculture   0.12263    0.02062   5.947 0.00000461 ***
## élevage     -0.19495    0.03279  -5.945 0.00000462 ***
## colza        0.05027    0.06787   0.741    0.46637
## forêt       -0.66771    0.19961  -3.345    0.00281 **
## naturel      0.24691    0.17124   1.442    0.16280
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.22 on 23 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.8309, Adjusted R-squared:  0.7941
## F-statistic:  22.6 on 5 and 23 DF,  p-value: 0.0000000354
```

```r
drop1(mod, test= "F")
```

```
## Single term deletions
##
## Model:
## dégats ~ gdeculture + élevage + colza + forêt + naturel
##            Df Sum of Sq    RSS    AIC F value      Pr(>F)
## <none>                  2893.4 145.48
## gdeculture  1    4448.6 7342.0 170.49 35.3623 0.000004611 ***
## élevage     1    4446.8 7340.2 170.48 35.3480 0.000004624 ***
## colza       1      69.0 2962.4 144.17  0.5486    0.466366
## forêt       1    1407.6 4301.0 154.98 11.1893    0.002808 **
## naturel     1     261.6 3155.0 145.99  2.0791    0.162803
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# On pourait donc être tenter de dire que c'est la surface de forêts dans les environs du
# champs qui a le plus d'influence sur le niveau de dégâts. Mais c'est faux.
# Si l'on regarde la gamme de variation entre variables explicatives, on constate que
# la surface de grande culture varie de 0 à 400, celle d'élevage
# de 0 à 250 et celle de fors de 0 à 50. Une augmentation de 1 ha est donc beaucoup plus
# importante pour la variable forêt que pour la variable grande culture.
# Si on standardise les variables explicatives, on peut comparer directement les
# coefficients bien que l'interprétation biologique directe soit plus délicate.
# On constate que les p-valeurs n'ont absolument pas changé sauf pour l'intercept.
# L'intercept estime maintenant que le niveau moyen de dégâts est de 41.89m². Lorsque
# la surface de grandes cultures augmente de 1 écart-type (ici = ~183 ha), la surface de
# dégâts augmente de 13.44 m², elle diminue de 12.14 m² quand les surfaces d'élevage
# augmentent de 1 écart-type (~67 ha) et elle diminue de 9.55 m² quand la surface de
# forêt augmente de  1 écart-type (14 ha). On peut ici dirrectement comparer les
# coefficients. Ce sont donc bien les grandes cultures et les zones d'élevage qui ont le
# plus grand effet sur les dégats et pas les zones forestières.

d2 <- d # copie du jeu de données
d2[, colnames(d2)!="dégats"] <- scale(d2[, colnames(d2)!="dégats"])
mod2 <- lm(dégats ~ gdeculture + élevage + colza + forêt + naturel, data = d2)
summary(mod2)
```

```
##
## Call:
## lm(formula = dégats ~ gdeculture + élevage + colza + forêt +
##     naturel, data = d2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -19.5135  -6.0291  -0.1155   6.7859  21.9874
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   41.889      2.091  20.032 4.67e-16 ***
## gdeculture    13.443      2.261   5.947 4.61e-06 ***
## élevage      -13.137      2.210  -5.945 4.62e-06 ***
## colza          1.624      2.193   0.741  0.46637
## forêt         -9.551      2.855  -3.345  0.00281 **
## naturel        3.980      2.760   1.442  0.16280
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.22 on 23 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.8309, Adjusted R-squared:  0.7941
## F-statistic:  22.6 on 5 and 23 DF,  p-value: 0.0000000354
```

```r
drop1(mod, test= "F")
```

```
## Single term deletions
##
## Model:
## dégats ~ gdeculture + élevage + colza + forêt + naturel
##            Df Sum of Sq    RSS    AIC F value      Pr(>F)
## <none>                  2893.4 145.48
## gdeculture  1    4448.6 7342.0 170.49 35.3623 0.000004611 ***
## élevage     1    4446.8 7340.2 170.48 35.3480 0.000004624 ***
## colza       1      69.0 2962.4 144.17  0.5486    0.466366
## forêt       1    1407.6 4301.0 154.98 11.1893    0.002808 **
```

```
## naturel     1     261.6 3155.0 145.99  2.0791    0.162803
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# La standardisation par la moyenne et l'écart-type est vraiment la plus classiquement
# utilisée. Cependant rien n'empêche d'utiliser d'autres transformations. Par exemple
# si on veut estimer l'effet maximal d'une variable explicative (comment varie y quand x
# passe de sa valeur minimale à sa valeur maximale), il suffit d'enlever le minimum et
# de diviser par le maximum de x-min(x). On obtient alors des valeurs comprises entre 0
# (= valeur minimale) et 1 (valeur maximale).
d3 <- d
d3[, colnames(d2)!="dégats"] <- apply(d3[, colnames(d2)!="dégats"], 2,
                                      function (x) (x-min(x, na.rm = TRUE))/max(x- min(x, na.rm = TRUE), na
mod3 <- lm(dégats ~ gdeculture + élevage + colza + forêt + naturel, data = d3)
summary(mod3)
```

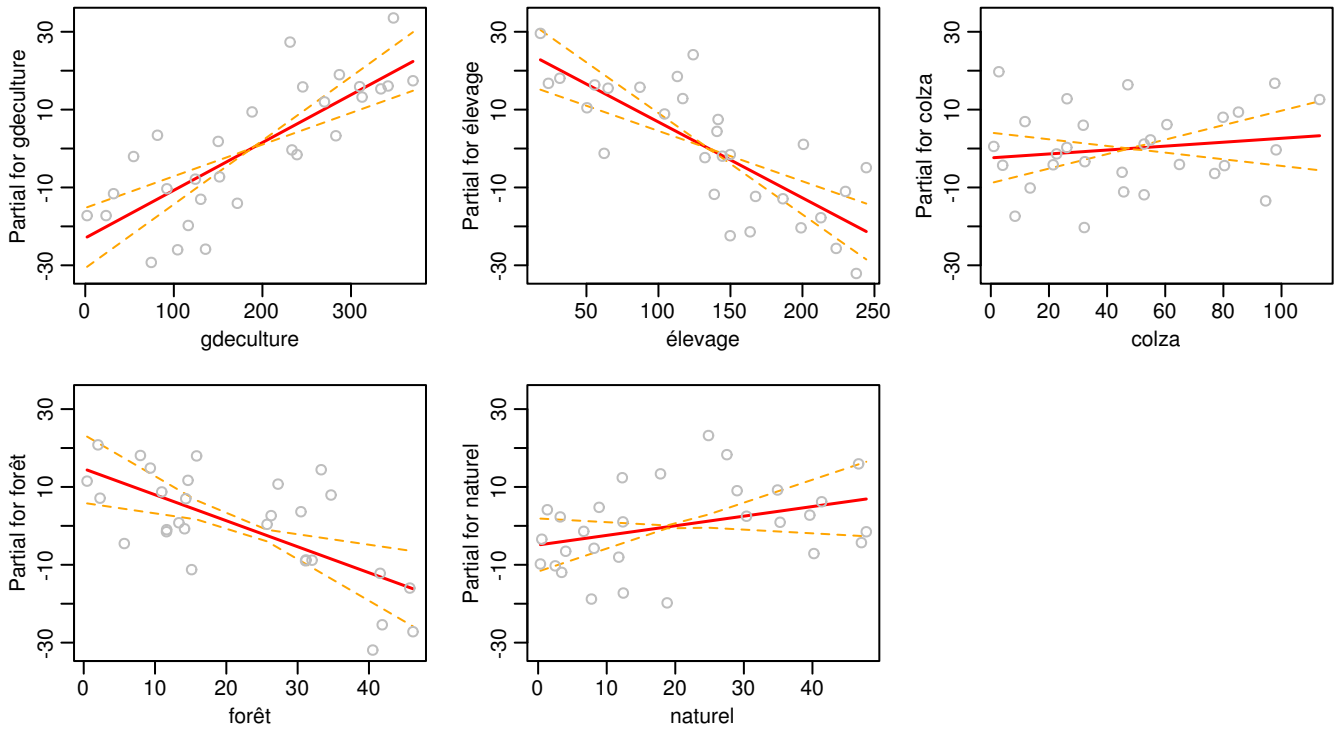
```
##
## Call:
## lm(formula = dégats ~ gdeculture + élevage + colza + forêt +
##     naturel, data = d3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -19.5135  -6.0291  -0.1155   6.7859  21.9874
##
## Coefficients:
##             Estimate Std. Error t value   Pr(>|t|)
## (Intercept)   50.773      8.129   6.246 0.00000226 ***
## gdeculture    45.115      7.587   5.947 0.00000461 ***
## élevage      -44.117      7.420  -5.945 0.00000462 ***
## colza          5.636      7.609   0.741    0.46637
## forêt        -30.514      9.122  -3.345    0.00281 **
## naturel       11.733      8.137   1.442    0.16280
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.22 on 23 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.8309, Adjusted R-squared:  0.7941
## F-statistic:  22.6 on 5 and 23 DF,  p-value: 0.0000000354
```

```r
# Donc par exemple pour les grandes cultures, on augmente la surface de dégats de 45.115m²
# quand on passe de la valeur minimale (~2ha) à la valeur maximale (~370 ha) ce qui
# multiplié pas le coefficient du premier modèle (0.12263) donne bien la même valeur :
(max(d$gdeculture) - min(d$gdeculture)) * coef(mod)["gdeculture"]
```

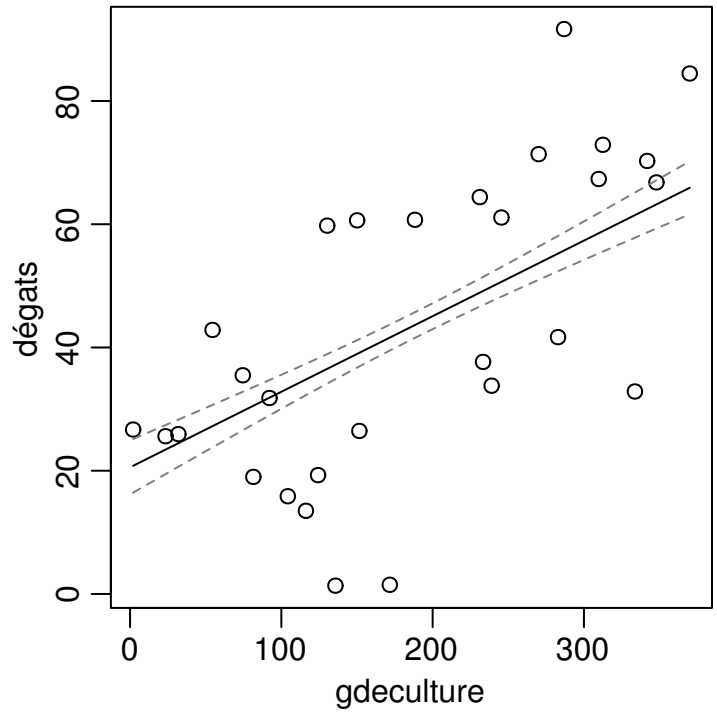
```
## gdeculture
##   45.11539
```

```r
# représentation graphique des modèles
```

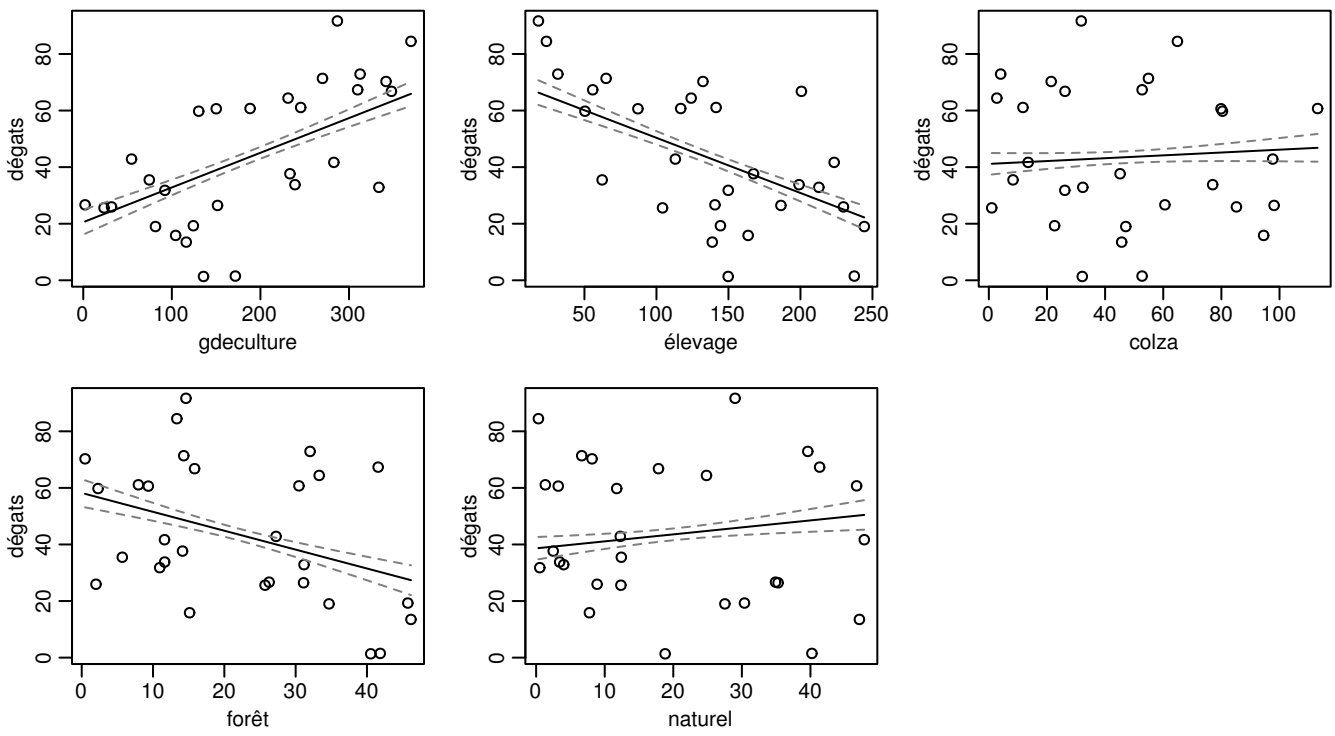
```r
# On peut utiliser la fonction termplot qui est rapide. Mais pas toujours facile
# de savoir ce que donne exactement une telle fonction. De plus elle sera d'un intérêt
# limité avec d'autres types de modèles. Les points représentés ne correspondent pas aux
# valeurs observés mais à des "résidus partiels". Les barres d'erreurs ne sont pas les
# erreurs standard des prédiction.
par(mfrow= c(2,3),mar = c(3,3,1,1), mgp = c(1.7, 0.6,0))
termplot(mod, partial.resid=T, se=TRUE)
```



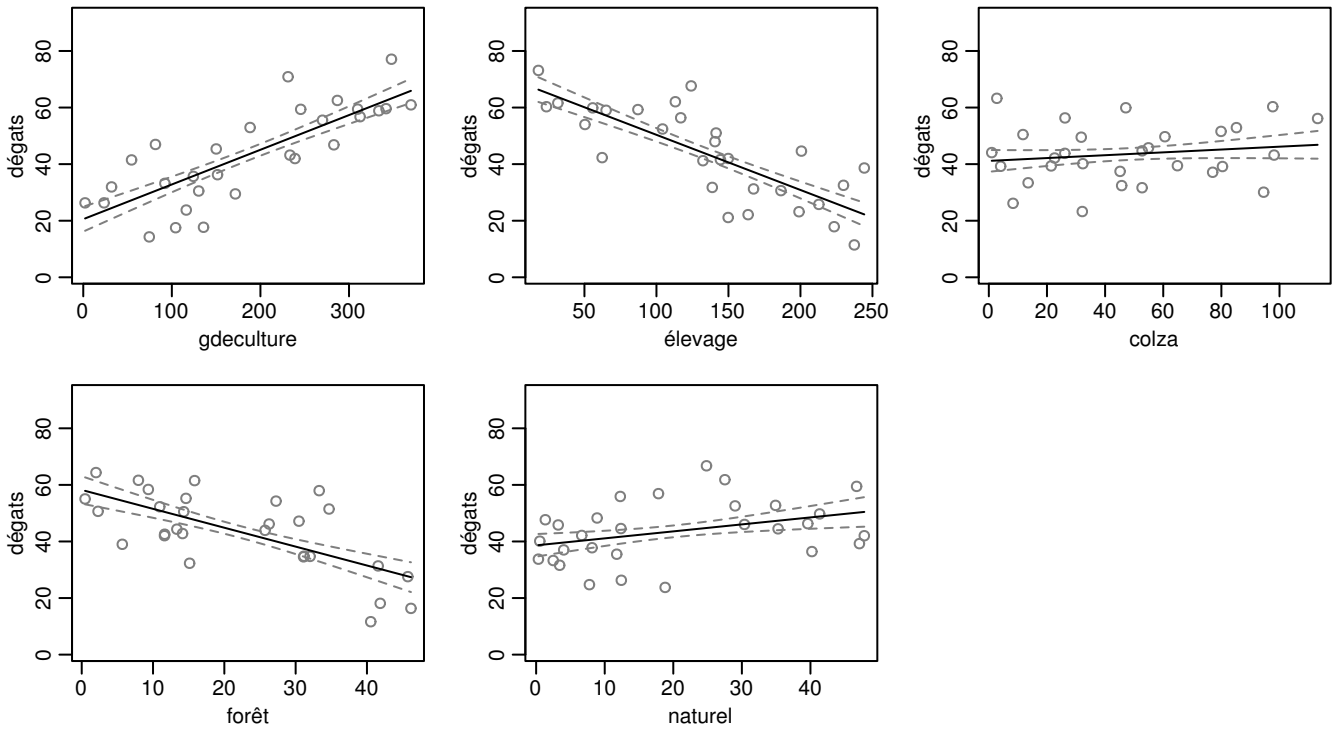

```
# On peut faire la prédiction pour chaque variable une par une
# Attention, dans le jeu de données, les lignes contenant des NA sont supprimées avant
# analyse. Il vaut donc mieux aller rechercher les données réellement utilisées par
# le modèle dans mod$model. La matrice X contient les moyennes de chaque variable sauf
# pour la variable d'intérêt pour laquelle on génère 1000 valeurs comprises entre la
# valeur minimale et la valeur maximale
obs <- mod$model
X <- cbind(1, seq(min(obs[,2]), max(obs[,2]), length.out=1000), mean(obs[,3]),
           mean(obs[,4]), mean(obs[,5]), mean(obs[,6]))
pred <- X %*% coef(mod)
se <- sqrt(diag(X %*% vcov(mod) %*% t(X)))

par(mfrow= c(1,1),mar = c(3,3,1,1), mgp = c(1.7, 0.6,0))
plot(obs[,1] ~ obs[,2], ylab = colnames(obs)[1], xlab = colnames(obs[2]))
lines(pred ~ X[, 2])
lines(y = pred + se, x= X[, 2], col = "grey50", lty = 2)
lines(y = pred - se, x= X[, 2], col = "grey50", lty = 2)
```




```
# On peut aussi traiter toutes les variables au moyen d'une boucle.
# Ici on a par exemple commencé par créer une matrice contenant pour chaque variable
# 1000 répétitions de la moyenne (objet "means").  Et un autre matrice contenant des
# séquences de  1000 valeurs comprises entre le minimum et le maximum de chaque variable.
# A chaque tour de la boucle, on va remplacer dans la matrice means la colonne
# correspondant à la variable d'intérêt par la séquence de 1000 nombres entre le minimum
# et le maximum.
# Le graphique résultant est similaire en apparence au graphique donné par termplot mais
# il est plus facilement interprétable (ea l'échelle des y). Par contre les points
# représentent les valeurs brutes.

means <- sapply(apply(mod$model[,-1], 2, mean), rep, times = 1000)
minmax <- rbind(apply(mod$model[,-1], 2, min), apply(mod$model[,-1], 2, max))
minmax <- apply(minmax, 2, function(x) seq(x[1], x[2], length.out = 1000))

par(mfrow = c(2,3), mar = c(3,3,1,1), mgp = c(1.7, 0.6,0))

for( i in 2:length(coef(mod))) {
    X <- cbind(1,means)
    X[,i] <- minmax[,i-1]
    pred <- X %*% coef(mod)
    se <- sqrt(diag(X %*% vcov(mod) %*% t(X)))
    plot(obs[,1] ~ obs[,i], ylab = colnames(obs)[1], xlab = colnames(obs[i]))
    lines(pred ~ X[, i])
    lines(y = pred + se, x= X[, i], col = "grey50", lty = 2)
    lines(y = pred - se, x= X[, i], col = "grey50", lty = 2)
}
```




```
# Pour info : on peut représenter des valeurs corrigées en ajoutant les résidus aux
# valeurs prédites.
# La représentation est plus proche du modèle (les points montrent la variation après
# avoir contrôlé pour les autres variables) mais il faut indiquer très clairement qu'on ne
# représente pas les valeurs réelles mais des valeurs corrigées.

means2 <- sapply(apply(mod$model[,-1], 2, mean), rep, times = nrow(obs))

par(mfrow = c(2,3), mar = c(3,3,1,1), mgp = c(1.7, 0.6,0))

for( i in 2:length(coef(mod))) {
    X <- cbind(1,means)
    X[,i] <- minmax[,i-1]
    pred <- X %*% coef(mod)
    se <- sqrt(diag(X %*% vcov(mod) %*% t(X)))

    X2 <- cbind(1,means2)
    X2[,i] <- obs[, i]
    partialresid <- X2 %*% coef(mod) + resid(mod)

    plot(obs[,1] ~ obs[,i], ylab = colnames(obs)[1], xlab = colnames(obs[i]), type = "n")
    points(y = partialresid, x = X2[,i], col = "grey50")
    lines(pred ~ X[, i])
    lines(y = pred + se, x= X[, i], col = "grey50", lty = 2)
    lines(y = pred - se, x= X[, i], col = "grey50", lty = 2)
}
```




```
# Le R² et R² ajusté sont donnés à la fin du summary du modèle. On peut aussi extraire
# l'info du summary Les deux valeurs sont assez proches dans ce cas car le nombre de
# données est assez élevé par rapport au nombre de paramètres
summary(mod)$r.squared
```

```
## [1] 0.8308624
```

```
summary(mod)$adj.r.squared
```

```
## [1] 0.7940934
```

```
# % de variance additionnelle expliqué par chaque variable explicative
modfull <- lm(dégats ~ gdeculture + élevage + colza + forêt + naturel, data = d)
mod_gdeculture <- lm(dégats ~ élevage + colza + forêt + naturel, data = d)
mod_élevage <- lm(dégats ~ gdeculture + colza + forêt + naturel, data = d)
mod_colza <- lm(dégats ~ gdeculture + élevage + forêt + naturel, data = d)
mod_forêt <- lm(dégats ~ gdeculture + élevage + colza + forêt , data = d)
mod_naturel <- lm(dégats ~ gdeculture + élevage + colza + forêt + naturel, data = d)

summary(mod)$r.squared *100
```

```
## [1] 83.08624
```

```
(summary(mod)$r.squared - summary(mod_gdeculture)$r.squared) *100
```

```
## [1] 26.00473
```

```
(summary(mod)$r.squared - summary(mod_élevage)$r.squared) *100
```

```
## [1] 25.99426
```

```r
(summary(mod)$r.squared - summary(mod_colza)$r.squared) *100
```

```
## [1] 0.4034657
```

```r
(summary(mod)$r.squared - summary(mod_forêt)$r.squared) *100
```

```
## [1] 1.016974
```

```r
(summary(mod)$r.squared - summary(mod_naturel)$r.squared) *100
```

```
## [1] 0
```

Exercice 6 : interaction simple entre 2 variables qualitatives

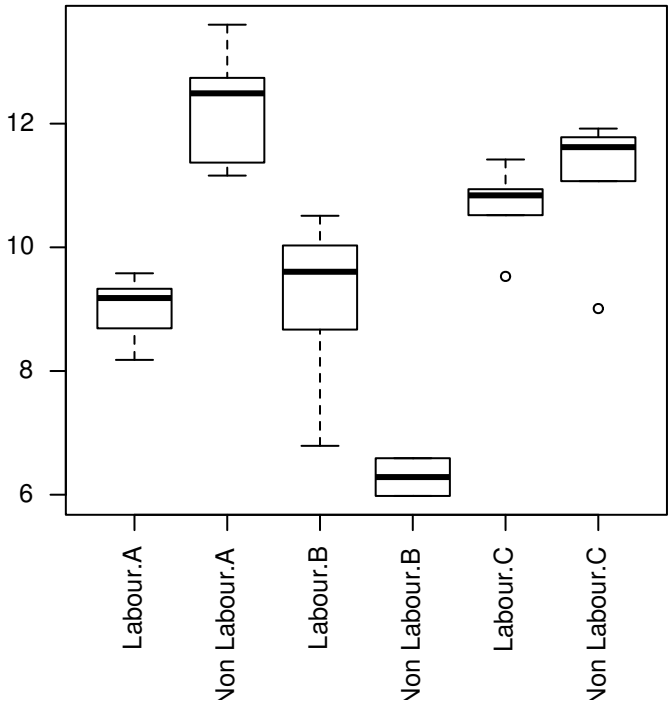
On a mesuré le rendement de froment (variable "y", en tonnes par hectares) sur une trentaine de champs semés avec 3 variétés différentes (A, B, C) et cultivés avec ou sans labour.

- Faites une représentation rapide des données (pex boxplot)
- Estimez un modèle prédictif du rendement
- Calculez les valeurs prédites, leurs erreurs standard et faites une représentation graphique du modèle et des données
- Interprétez les coefficients du modèle.
- Est-ce que le rendement est différent entre les champs labourés et non labourés ? Est-ce que le rendement est différent entre les variétés ? Quels tests globaux (comparaisons de modèles emboîtés) pourriez-vous faire ici et quelle serait leur signification ?
- Construisez la matrice de contrastes non indépendants permettant de faire les comparaisons multiples suivantes :
  - comparer chaque variété entre labour et non labour
  - comparer les variétés entre elles dans les champs labourés
  - comparer les variétés entre elles dans les champs non labourés

```
# génération du jeu de données
set.seed(3)
d <- data.frame(
    var = rep(c("A", "B", "C"), each = 10),
    culture = sample(c("Non Labour", "Labour"), 30, replace = TRUE)
)

X <- model.matrix(~ var * culture, data=d)
B <- c(9, 0 , 2, 3 , -6, -3)

set.seed(1)
d$y <- round(X %*% B + rnorm(30, 0, 1),2)

# Données pour l'énoncé
```

```
d <- structure(list(var = structure(c(1L, 1L, 1L, 1L, 1L, 1L, 1L,
1L, 1L, 1L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 3L, 3L, 3L,
3L, 3L, 3L, 3L, 3L, 3L, 3L), .Label = c("A", "B", "C"), class = "factor"),
culture = structure(c(2L, 1L, 2L, 2L, 1L, 1L, 2L, 2L, 1L,
1L, 1L, 1L, 1L, 1L, 1L, 1L, 2L, 1L, 1L, 2L, 2L, 2L, 2L, 2L,
2L, 1L, 1L, 1L, 1L, 1L), .Label = c("Labour", "Non Labour"
), class = "factor"), y = structure(c(11.37, 9.18, 11.16,
13.6, 9.33, 8.18, 12.49, 12.74, 9.58, 8.69, 10.51, 9.39,
8.38, 6.79, 10.12, 8.96, 5.98, 9.94, 9.82, 6.59, 11.92, 11.78,
11.07, 9.01, 11.62, 10.94, 10.84, 9.53, 10.52, 11.42), .Dim = c(30L,
1L), .Dimnames = list(c("1", "2", "3", "4", "5", "6", "7",
"8", "9", "10", "11", "12", "13", "14", "15", "16", "17",
"18", "19", "20", "21", "22", "23", "24", "25", "26", "27",
"28", "29", "30"), NULL))), .Names = c("var", "culture",
"y"), row.names = c(NA, -30L), class = "data.frame")
```

```
# Plot rapide des données
par(las=2, mar = c(6, 3, 2, 2), cex = 0.8)
boxplot(y ~ culture + var, data=d)
```

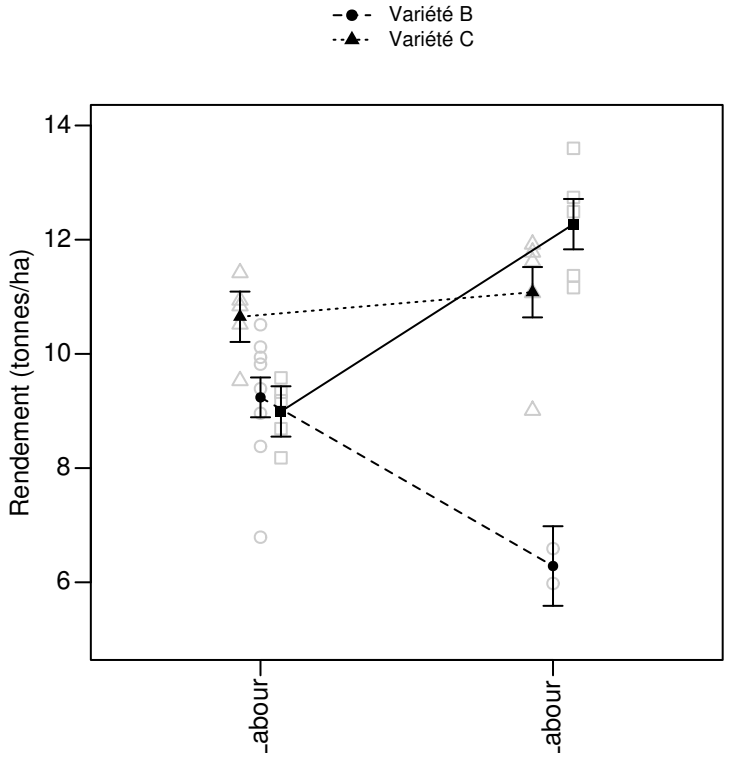



```
# Modèle
options(scipen = 4)
mod <- lm (y ~ var * culture, data = d)
summary(mod)
```

```
##
## Call:
## lm(formula = y ~ var * culture, data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.4487 -0.3043  0.2040  0.5863  1.3280
##
## Coefficients:
##                         Estimate Std. Error t value   Pr(>|t|)
## (Intercept)               8.9920     0.4409  20.395    < 2e-16 ***
## varB                      0.2467     0.5620   0.439    0.66456
## varC                      1.6580     0.6235   2.659    0.01373 *
## cultureNon Labour         3.2800     0.6235   5.261 0.00002151 ***
## varB:cultureNon Labour   -6.2337     0.9981  -6.246 0.00000187 ***
## varC:cultureNon Labour   -2.8500     0.8818  -3.232    0.00355 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9858 on 24 degrees of freedom
## Multiple R-squared:  0.7527, Adjusted R-squared:  0.7012
## F-statistic: 14.61 on 5 and 24 DF,  p-value: 0.000001285
```

```
# Valeurs prédites et se
X <- rbind("Var A - Labour" = c(1,0,0,0,0,0),
           "Var B - Labour" = c(1,1,0,0,0,0),
           "Var C - Labour" = c(1,0,1,0,0,0),
           "Var A - Non Labour" = c(1,0,0,1,0,0),
           "Var B - Non Labour" = c(1,1,0,1,1,0),
```

```
           "Var C - Non Labour" = c(1,0,1,1,0,1)
           )

pred <- X %*% coef(mod)
se <- sqrt(diag(X %*% vcov(mod) %*% t(X)))
lwr <- pred - se
upr <- pred + se


# représentation graphique
# dev.new(10/2.54, 10/2.54)
shift <- 0.07 # valeur pour décaler les points

par(mar = c(3,3,3.5,1), mgp = c(1.75, 0.6, 0))
plot(d$y ~ as.numeric(d$culture), xlim = c(0.5, 2.5), ylim = c(5,14),
     type = "n", xaxt = "n", xlab = "", ylab = "Rendement (tonnes/ha)")
axis(side = 1, at = c(1,2), labels = c("Labour", "Non Labour"))

points(d[d$var == "A","y"] ~ I(as.numeric(d[d$var == "A","culture"]) + shift),
       col = "gray80", pch = 0)
points(d[d$var == "B","y"] ~ I(as.numeric(d[d$var == "B","culture"])),
       col = "gray80", pch = 1)
points(d[d$var == "C","y"] ~ I(as.numeric(d[d$var == "C","culture"]) - shift),
       col = "gray80", pch = 2)

points(y = pred, x = c(1,1,1,2,2,2) + c(shift, 0, -shift, shift, 0, -shift),
       pch = c(15:17, 15:17), cex = 0.9)
arrows(y0 = lwr, x0 = c(1,1,1,2,2,2) + c(shift, 0, -shift, shift, 0, -shift),
       y1 = upr, x1 = c(1,1,1,2,2,2) + c(shift, 0, -shift, shift, 0, -shift),
       code = 3, angle = 90, length = 0.05)

segments(y0 = pred[1:3], y1 = pred[4:6], x0 = c(1,1,1) + c(shift,0, -shift),
         x1 = c(2,2,2) + c(shift,0, -shift), lty = c(1:3))

legend("top", xpd = TRUE, inset = -0.25, bty = "n", pt.cex = 1, cex = 0.8, lty = 1:3,
       pch = c(15:17), legend =  c("Variété A", "Variété B", "Variété C"))
```




```
# Interprétation du modèle
# Est-ce que le rendement est différent entre les champs labourés et non labourés ?
# Réponse : oui mai ça dépend de la variété ! (Interaction significative)
# Est-ce que le rendement est différent entre les variétés ? Réponse : Oui mais çà
# dépend du mode de culture.
drop1(mod, test = "F")
```

```
## Single term deletions
##
## Model:
## y ~ var * culture
##             Df Sum of Sq    RSS     AIC F value      Pr(>F)
## <none>                   23.325  4.4505
## var:culture  2    38.182 61.508 29.5390  19.643 0.000008847 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# L'interaction étant fortement significative, on ne devrait pas tester les effets
# principaux seuls.
# Si on veut vraiment voir si il y a une différence globale entre les deux modes de
# culture, sans connaître la variété semée, on peut faire le test suivant.
# Conclusion, sans savoir quel type de variété on a semé, on ne peut pas à l'avance dire
# si un mode de culture sera plus rentable qu'un autre
mod1 <- lm (y ~ var + culture, data = d)
mod2 <- lm (y ~ var , data = d)
anova(mod1, mod2)
```

```
## Analysis of Variance Table
##
## Model 1: y ~ var + culture
## Model 2: y ~ var
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
```

```
## 1     26 61.508
## 2     27 64.643 -1   -3.1354 1.3254 0.2601
```

```
# En ce qui concerne les variétés il semble que certaines variétés soient plus rentables
# que d'autres même sans savoir dans quel mode de travail du sol elles ont été cultivées
# (p = 0.018).
#
mod1 <- lm (y ~ var + culture, data = d)
mod2 <- lm (y ~ culture, data = d)
anova(mod1, mod2)
```

```
## Analysis of Variance Table
##
## Model 1: y ~ var + culture
## Model 2: y ~ culture
##   Res.Df    RSS Df Sum of Sq      F  Pr(>F)
## 1     26 61.508
## 2     28 83.695 -2   -22.187 4.6894 0.01824 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# NB on aurait pu obtenir les mêmes valeurs plus simplement avec drop1 :
drop1(lm(y ~ var + culture, data = d), test = "F")
```

```
## Single term deletions
##
## Model:
## y ~ var + culture
##         Df Sum of Sq    RSS    AIC F value  Pr(>F)
## <none>               61.508 29.539
## var      2   22.1871 83.695 34.779  4.6894 0.01824 *
## culture  1    3.1354 64.643 29.031  1.3254 0.26011
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# comparaisons multiples
# La variété A réagi positivement au non labour (3.2 tonnes en plus en moyenne).
# La variété B réagi négativement au labour (2.95 tonnes en moins en moyenne).
# La variété C semble se comporter de manière similaire dans les deux modes de culture.
# Dans les champs labourés, les 3 variétés ont un rendement similaire. La variété C
# tend à avoir un rendement légèrement supérieur (+ 1.4/+1.6) mais on aurait pu obtenir
# de telles différences par hasard. Les données sont donc insuffisantes pour affirmer
# qu'il y a réellement une différence.
# Dans les champs non labourés  la variété B a un rendement nettement inférieur aux autres
# de 4.8 à 6 tonnes.
coef(mod)
```

```
##           (Intercept)                  varB                  varC      cultureNon Labour
##               8.99200               0.24675               1.65800               3.28000
## varB:cultureNon Labour varC:cultureNon Labour
##              -6.23375              -2.85000
```

```
C <- rbind(
    "A : lab - NonLab" = c(0,0,0,-1,0,0),
    "B : lab - NonLab" = c(0,0,0,-1,-1,0),
    "C : lab - NonLab" = c(0,0,0,-1,0,-1),
    "lab : A - B" = c(0,-1,0,0,0,0),
```

```
    "lab : A - C" = c(0,0,-1,0,0,0),
    "lab : B - C" = c(0,1,-1,0,0,0),
    "Nonlab : A - B" = c(0,-1,0,0,-1,0),
    "Nonlab : A - C" = c(0,0,-1,0,0,-1),
    "Nonlab : B - C" = c(0,1,-1,0,1,-1)
    )

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

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Fit: lm(formula = y ~ var * culture, data = d)
##
## Linear Hypotheses:
##                       Estimate Std. Error t value Pr(>|t|)
## A : lab - NonLab == 0  -3.2800     0.6235  -5.261  < 0.001 ***
## B : lab - NonLab == 0   2.9538     0.7794   3.790  0.00666 **
## C : lab - NonLab == 0  -0.4300     0.6235  -0.690  0.97118
## lab : A - B == 0       -0.2467     0.5620  -0.439  0.99617
## lab : A - C == 0       -1.6580     0.6235  -2.659  0.08842 .
## lab : B - C == 0       -1.4112     0.5620  -2.511  0.11876
## Nonlab : A - B == 0     5.9870     0.8248   7.259  < 0.001 ***
## Nonlab : A - C == 0     1.1920     0.6235   1.912  0.34256
## Nonlab : B - C == 0    -4.7950     0.8248  -5.813  < 0.001 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

Exercice 7 : interaction entre une variable quantitative et une variable qualitative

On a suivi chaque année depuis 1990 l'abondance d'une espèce de papillon sur un site fauché chaque année. En 2000, on décide de changer de mode de gestion en passant à du pâturage extensif que l'on pense être plus adapté et on continue à suivre la population jusqu'en 2010. Chaque année on compte les papillons le long de 5 transects indépendants et positionnés aléatoirement sur le site. Pour simplifier le modèle on utilisera la moyenne par transect (on pourrait utiliser les données brutes mais alors il faudrait inclure une variable explicative "date"). La question est de savoir si la population a réagi positivement au changement de mode de gestion sur ce site.

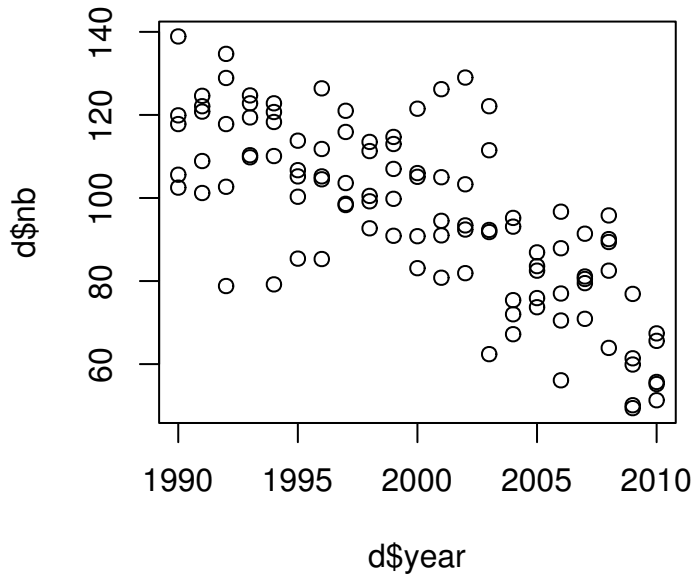
- Comme toujours faites un graphique de vos données brutes pour les visualiser...
- Créez une variable (facteur) permettant de distinguer les années avant ou après le changement de mode de gestion (on l'appellera "change"). On va construire un modèle du nombre de papillons en fonction de l'année, de la variable "change" et de leur interaction
- Ne serait-il pas judicieux de centrer la variable "year" et si oui, sur quelle valeur ?
- Estimez le modèle, faites les inférences et interprétez les paramètres estimés. Quelle était la tendance en nombre d'individus par an avant et après le changement de gestion (après = -4.1067 individus par an).
- Faites une représentation graphique des données et du modèle
- Quelle était la tendance en % d'individus perdus sur 10 ans avant et après le changement (avant : -13.34%, après : - 39.45%)
- Est-ce que dans cette étude, enlever le facteur "change" tout en gardant l'interaction change x years aurait du sens ? Estimez ce modèle et faites en une représentation graphique (avec les erreurs standard des prédictions)
- sur base de ce modèle évaluez les 3 hypothèses/questions suivantes en corrigeant les p-valeurs pour prendre en compte le risque global :1 - Est-ce que la pente était différente de 0 avant 2000 ?, 2 - Est-ce qu'elle l'était après 2000 ?, 3 - est-ce que la pente était différente avant et après 2000 ?
- Est-ce qu'on peut extrapoler ces résultats à d'autres sites ? Est-ce qu'on peut conclure définitivement que c'est le mode de gestion qui a aggravé la situation ?

```r
# Génération du jeu de données
n <- 5
year <- rep(-10:10, each = n)
change <- factor(ifelse(year < 0, "before", "after" ), levels = c("before", "after"))
X <- model.matrix(~ year + year:change)
B <- c(100, -1.5, -2 )
set.seed(1)
y <- round( X%*%B + rnorm (n*21, 0, 15), 1)
d <- data.frame(nb = y, year = year + 2000)

# Stockage des données pour les présenter dans les énoncés
```

```r
d <- data.frame(
    nb = c(105.6, 117.8, 102.5, 138.9, 119.9, 101.2, 120.8,
           124.6, 122.1, 108.9, 134.7, 117.8, 102.7, 78.8, 128.9, 109.8,
           110.3, 124.7, 122.8, 119.4, 122.8, 120.7, 110.1, 79.2, 118.3,
           106.7, 105.2, 85.4, 100.3, 113.8, 126.4, 104.5, 111.8, 105.2,
           85.3, 98.3, 98.6, 103.6, 121, 115.9, 100.5, 99.2, 113.5, 111.3,
           92.7, 90.9, 107, 113, 99.8, 114.7, 106, 90.8, 105.1, 83.1, 121.5,
           126.2, 91, 80.8, 105, 94.5, 129, 92.4, 103.3, 93.4, 81.9, 92.3,
           62.4, 111.5, 91.8, 122.1, 93.1, 75.4, 95.2, 72, 67.2, 86.9, 75.9,
           82.5, 83.6, 73.7, 70.5, 77, 96.7, 56.1, 87.9, 80.5, 91.4, 70.9,
           81.1, 79.5, 63.9, 90.1, 89.4, 82.5, 95.8, 76.9, 49.4, 59.9, 50.1,
           61.4, 55.7, 65.6, 51.3, 67.4, 55.2),
    year = c(1990, 1990, 1990, 1990, 1990, 1991, 1991, 1991, 1991, 1991,
             1992, 1992, 1992, 1992, 1992, 1993, 1993, 1993, 1993, 1993, 1994,
             1994, 1994, 1994, 1994, 1995, 1995, 1995, 1995, 1995, 1996, 1996,
             1996, 1996, 1996, 1997, 1997, 1997, 1997, 1997, 1998, 1998, 1998,
             1998, 1998, 1999, 1999, 1999, 1999, 1999, 2000, 2000, 2000, 2000,
```

```
            2000, 2001, 2001, 2001, 2001, 2001, 2002, 2002, 2002, 2002, 2002,
            2003, 2003, 2003, 2003, 2003, 2004, 2004, 2004, 2004, 2004, 2005,
            2005, 2005, 2005, 2005, 2006, 2006, 2006, 2006, 2006, 2007, 2007,
            2007, 2007, 2007, 2008, 2008, 2008, 2008, 2008, 2009, 2009, 2009,
            2009, 2009, 2010, 2010, 2010, 2010, 2010)
    )
```

```
# graphique
plot(d$nb ~ d$year)
```



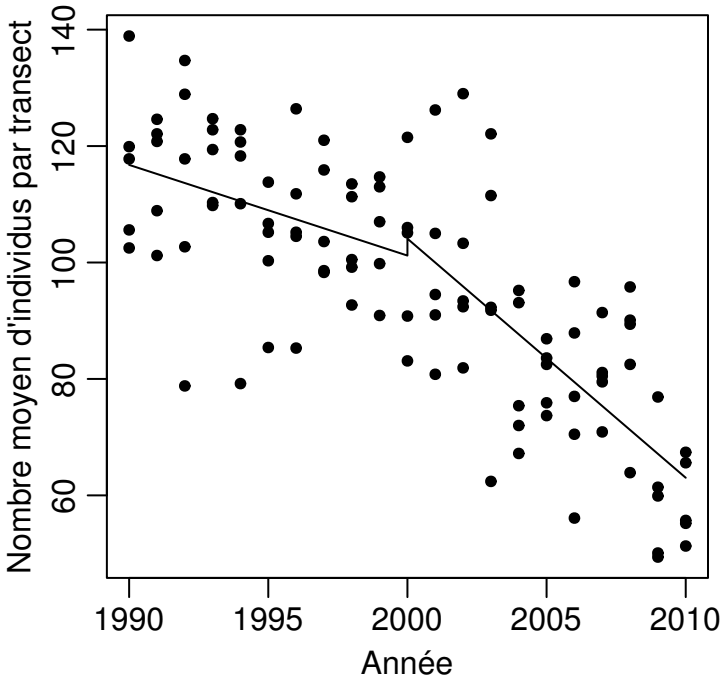

```
# Création de la variable "change"
d$change <- factor(ifelse(d$year < 2000, "before", "after" ),
                   levels = c("before", "after"))

# On centre les données sur la date pivot 2000. On aurait put aussi centrer sur l'année
# 1990 mais centrer sur 2000 s'avérera utile par la suite.
d$yearc <- d$year - 2000

# estimation du modèle et résultats
# En 2000 on estime qu'il y avait en moyenne 101.2 papillons par transect (intercept).
# L'effet "change" n'est pas significativement différent de 0, ce qui est logique :
# l'abondance moyenne estimée en 2000 est similaire pour les deux droites.
# Avant le changement de gestion on perdait en moyenne 1.56 individus par an et après
# le changement de gestion, on perdait -1.5577 - 2.5490 = -4.1067 individus par an
# L'interaction est fortement significative ce qui montre que le déclin est
# significativement plus fort après le changement de gestion.
# La population était donc déjà en déclin avant le changement mais elle l'est encore
# plus après.
mod <- lm(nb ~ yearc + change + yearc:change, data=d)
summary(mod)
```

```
##
## Call:
## lm(formula = nb ~ yearc + change + yearc:change, data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -34.852  -7.826   0.027   8.447  33.120
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       101.1907     4.1040  24.657   <2e-16 ***
## yearc              -1.5577     0.6614  -2.355   0.0205 *
## changeafter         2.9030     5.3223   0.545   0.5867
## yearc:changeafter  -2.5490     0.8750  -2.913   0.0044 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.43 on 101 degrees of freedom
## Multiple R-squared:  0.6079, Adjusted R-squared:  0.5962
## F-statistic: 52.19 on 3 and 101 DF,  p-value: < 2.2e-16
```

```
Anova(mod)
```

```
## Anova Table (Type II tests)
##
## Response: nb
##               Sum Sq  Df F value    Pr(>F)
## yearc         8745.2   1 48.4607 3.463e-10 ***
## change         194.6   1  1.0785  0.301516
## yearc:change  1531.6   1  8.4870  0.004404 **
## Residuals    18226.4 101
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Graphique
# NB : pour un résultat plus "clean" il faudrait tracer deux droites séparées
# cependant le résultat étrange obtenu (le hiatus en 2000) est le reflet réel de notre
# modèle et montre qu'il pourrait être amélioré en enlevant  le facteur "change"
# tout en gardant l'interaction
par(mar = c(3,3,2,1), mgp = c(1.75, 0.6, 0))
X <- cbind(1, c(-10, 0, 0, 10), c(0,0,1,1), c(0,0,0,10))
pred <- X %*% coef(mod)
plot(y = d$nb, x = d$year, pch = 20, xlab = "Année",
     ylab = "Nombre moyen d'individus par transect")
lines(y = pred, x = X[,2]+2000)
```




```
# pour estimer le % de perte en 10 ans, on utilise le nombre d'individus en début
# et en fin de période :
(pred[2,] - pred[1,]) * 100/ pred[1,] # avant
```

```
## [1] -13.34014
```

```
(pred[4,] - pred[3,]) * 100/ pred[3,] # après
```

```
## [1] -39.45224
```

```
# modèle sans l'effet "change"
# Considérer que l'effet "change" est nul a du sens dans ce cas particulier. En effet,
# comme on a centré les données sur l'année 2000, cela reviens à forcer les deux droites
# à estimer un même nombre de papillons (égal à l'intercept) en 2000. Les deux positions
# peuvent en fait se défendre. Ceux qui considèrent que le modèle doit être le plus
# cohérent possible avec le système étudié. On retire alors l'effet "change". D'autres
# considèrent que dans ce cas le plus important est d'estimer au mieux les pentes des
# droites et que dans ce cas il vaut mieux ne pas les forcer à passer toutes les deux par
# le même point en 2000 quitte à avoir deux estimations différentes pour la même année.
mod <- lm(nb ~ yearc  + yearc:change, data=d)
summary(mod)
```

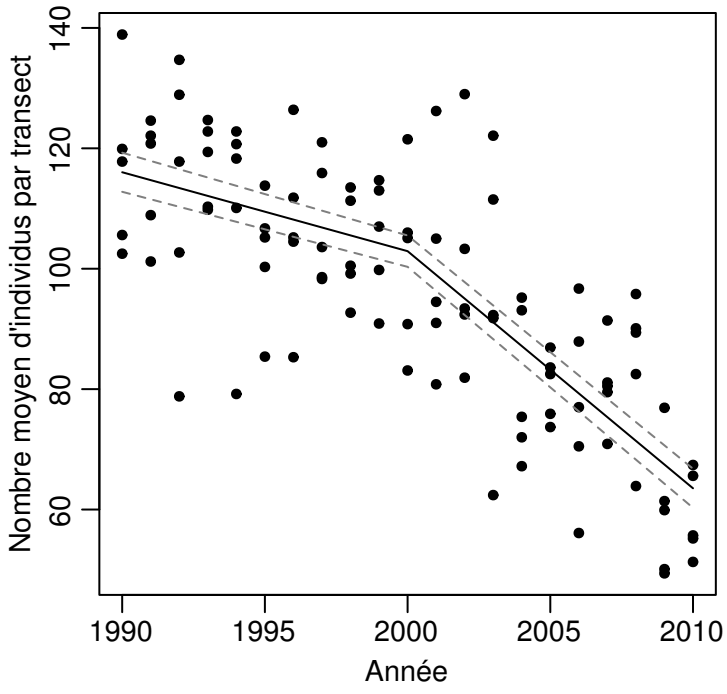
```
##
## Call:
## lm(formula = nb ~ yearc + yearc:change, data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -34.606  -8.331   0.376   8.260  33.960
##
## Coefficients:
```

```
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       102.9168     2.6041  39.522  < 2e-16 ***
## yearc              -1.3111     0.4811  -2.725  0.00757 **
## yearc:changeafter  -2.6275     0.8601  -3.055  0.00287 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.39 on 102 degrees of freedom
## Multiple R-squared:  0.6067, Adjusted R-squared:  0.599
## F-statistic: 78.67 on 2 and 102 DF,  p-value: < 2.2e-16
```

```
Anova(mod)
```

```
## Anova Table (Type II tests)
##
## Response: nb
##               Sum Sq  Df  F value    Pr(>F)
## yearc        26526.0   1 148.0109 < 2.2e-16 ***
## yearc:change  1672.5   1   9.3323  0.002873 **
## Residuals    18280.1 102
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Graphique
par(mar = c(3,3,2,1), mgp = c(1.75, 0.6, 0), cex = 0.9)
X <- cbind(1, c(-10, 0, 0, 10), c(0,0,0,10))
pred <- X %*% coef(mod)
se <- sqrt(diag(X %*% vcov(mod) %*% t(X)))
plot(y = d$nb, x = d$year, pch = 20, xlab = "Année",
     ylab = "Nombre moyen d'individus par transect")
lines(y = pred, x = X[,2]+2000)
lines(y = pred+se, x = X[,2]+2000, col = "grey50", lty = 2)
lines(y = pred-se, x = X[,2]+2000, col = "grey50", lty = 2)
```



```r
# Comparaisons multiples:
# La pente avant 2000 est de -1.3 individu/an et de -3.9 individus/an après 2000.
# Elles sont toute deux statistiquement différentes de 0.
#

C <- rbind("Pente avant 2000" = c(0,1,0),
           "Pente après 2000" = c(0,1,1),
           "Différence de pente" = c(0,0,1))
library(multcomp)
modmc <- glht(mod, linfct = C)
summary(modmc)
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Fit: lm(formula = nb ~ yearc + yearc:change, data = d)
##
## Linear Hypotheses:
##                           Estimate Std. Error t value Pr(>|t|)
## Pente avant 2000 == 0      -1.3111     0.4811  -2.725  0.01535 *
## Pente après 2000 == 0      -3.9386     0.4811  -8.186  < 0.001 ***
## Différence de pente == 0   -2.6275     0.8601  -3.055  0.00604 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

```r
# On ne peut pas extrapoler à d'autres sites. La population d'intérêt ici est l'ensemble
# des transects réalisables sur ce site. Pour pouvoir extrapoler, il aurait fallu
# échantillonner plusieurs sites (en tenant compte de l'effet site dans l'analyse).
# Le fait qu'il y ait un changement brusque dans la population juste au moment où on
# change de mode de gestion semble indiquer un lien de cause à effet, cependant on ne peut
# pas exclure qu'il y ait eu au même moment un autre effet qui soit la cause réelle.
# En multipliant le nombre de sites, on diminue la probabilité qu'un autre facteur
# puisse provoquer le même résultat, en particulier si l'année pivot où on change le mode
# de gestion n'est pas la même d'un site à l'autre.
# On pourrait aussi ajouter un (idéalement plusieurs) site contrôle dans lequel on continue
# à gérer de la même façon. Il faudra alors comparer les pentes des droites avant et après
# par rapport au contrôle (interaction triple année x change x contrôle/pas contrôle).
#
```

Exercice 8 : non linéarité et transformation de variables

Voici un jeu de données avec une variable y que l'on veut prédire en fonction de 5 variables explicatives x1 à x5. Aucune de ces relations ne sont linéaires. Le but de l'exercice est de trouver les bonnes transformations pour linéariser ces relations.

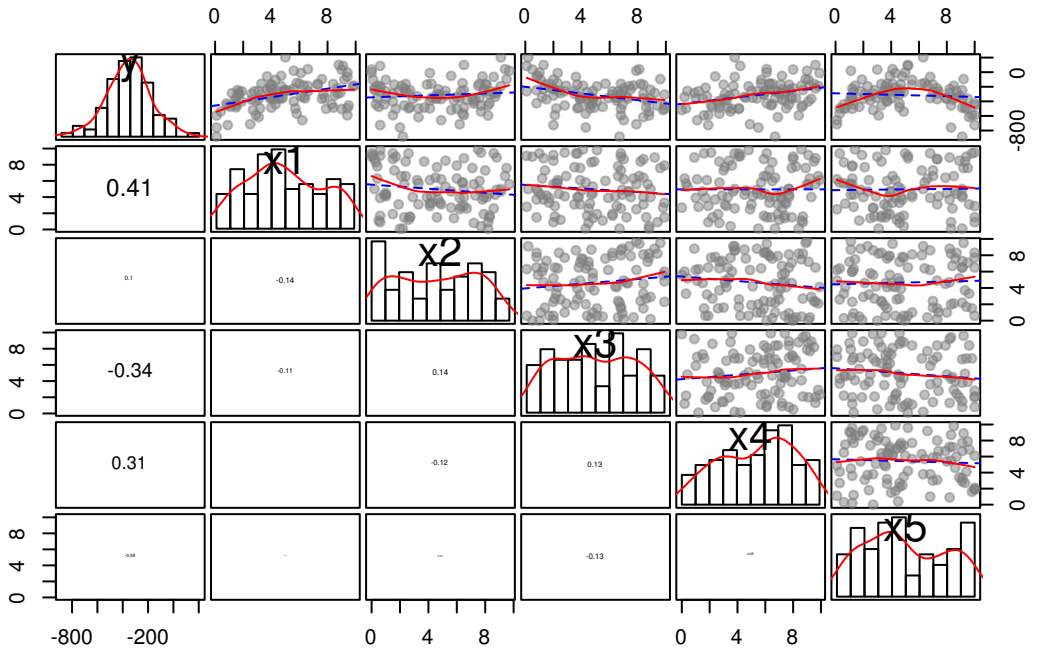
- Explorez graphiquement les relations 2 à 2 entre y et les 5 variables explicatives. Vous pouvez faire un graphique pour chaque combinaison (dans une boucle par exemple) mais vous pouvez aussi utiliser la fonction pairs2 (après avoir sourcé le script mytoolbox.R) qui vous permettra de vérifier en plus qu'il n'y a pas trop de corrélation entre les variables explicatives
- Faites un modèle simple de y en fonction des 5 variables explicatives sans interactions. Notez le R^2 et l'erreur standard résiduelle. Faites un graphique des résidus en fonction des valeurs prédites et des graphiques permettant d'examiner la distribution des résidus. Vous pouvez utiliser plot(votremodèle) ou la fonction diagplot(votremodèle) après avoir sourcé le script(mytoolbox.R). Interprétez-les.
- Faites un graphique des résidus en fonction de chaque variable explicative. Vous pouvez utiliser la fonction diagplot2 (dans mytoolbox.R). Essayez de trouver une transformation/méthode permettant de linéariser la variable dont la relation est la plus clairement non linéaire (x5 pour commencer). Refaites des plots résidus vs variables explicatives et recommencez jusqu'à avoir linéarisé les 5 relations. Aidez-vous de la règle de Mosteller & Tukey vue dans la partie théorique.
- Lorsque vous obtenez votre modèle final, vérifiez qu'il est bon et comparez les R^2 et l'erreur standard résiduelle avec les valeurs obtenues dans le modèle de départ. Faites une représentation graphique des données et du modèle pour chaque variable explicatives. Faites ces graphiques sur l'échelle d'origine (pex y ~ x1 et pas y~x1^3).

```
x1 <- runif(100, 0, 10)
x2 <- runif(100, 0, 10)
x3 <- runif(100, 0, 10)
x4 <- runif(100, 0, 10)
x5 <- runif(100, 0, 10)

set.seed(1)
y <- 100 * log(x1) + 0.025* exp(x2) - 300*x3^0.25 + 2 * x4^2 + (x5-5) - 15*(x5-5)^2 +
    rnorm(100, 0, 10)
d <- round(data.frame(y, x1, x2, x3, x4, x5),1)
```

```r
# Jeu de données pour énnoncé

d <- structure(list(y = c(-149.6, -95.1, -307.6, -319.7, -590.5, -247.2,
-713.4, -295.8, -502.2, -401.9, -425.3, -262.1, -400.7, -364.7,
-299.1, -272.3, 99, -187, -293.6, -355.8, -63.6, -377.9, 203.2,
-261.4, -456.4, -234.6, -170.7, -11.3, -212.9, 36.2, -658.1,
-340.7, -191.7, -241.3, -481.6, -758.9, -104.9, -350.5, -101.2,
-169.6, -3.5, -306.7, -360.4, -593.8, -233.6, -364.9, -387.1,
41.5, -884.3, -505.6, -283.4, -358.2, -371.4, -519.5, -263.7,
-257.3, -496.4, -497.1, 20, -158.9, -256.2, -50, -589.1, -540,
-245.7, -261.3, -288.3, 86.2, -161.6, -57.8, -257.5, -338.3,
-353.2, -261.6, -361.5, -441.2, -494.7, -486.4, -567.5, -441.3,
-223.3, -455.1, -185.6, -475.4, -725.7, -486.5, -351.1, -183.5,
-171.9, -220.5, -159.6, -193.8, -342.3, -329.7, -157.4, -615.9,
-391.5, -369.2, -430.4, -455.8), x1 = c(5.8, 7.6, 1.7, 8.5, 9.4,
5, 2.5, 2.6, 7.9, 1.9, 1.5, 1.9, 5.6, 4, 7.7, 1.7, 8.8, 6.6,
8.7, 3.9, 9.4, 2.6, 5, 4.1, 9.5, 4, 7.3, 6.1, 3.9, 9, 6.2, 6.3,
4.7, 5, 1.5, 0.1, 4.3, 4.1, 9.2, 4, 9.4, 3.3, 4.5, 0.1, 3.7,
7.6, 1.4, 6.6, 0.1, 9, 8.9, 6.4, 4.2, 2.7, 6.6, 5.3, 3.2, 5.7,
5.6, 3.3, 3.1, 4.7, 2.9, 0.4, 1.6, 4.9, 1.2, 9.1, 8.4, 3.7, 4.4,
4.3, 7.8, 2.8, 4.7, 8.3, 9.4, 8.4, 5.8, 4.3, 5.6, 1.5, 9.9, 1.1,
0.9, 4, 4.8, 6.3, 4, 8.7, 7.3, 9.8, 1.5, 6.1, 6, 3.1, 2.9, 3.1,
0.8, 0.9), x2 = c(6.4, 9.5, 8.3, 5.2, 1.5, 6.5, 6.2, 8.6, 6,
1.5, 8.3, 7.9, 7.2, 2.9, 4.6, 0.6, 6.5, 2.1, 1.5, 6.2, 8.6, 6.7,
9.1, 1, 7.7, 7.9, 1.6, 4.1, 0.6, 0, 4.9, 6, 9.1, 0.2, 2.7, 2,
1, 4.2, 0.1, 3.9, 0, 4.8, 1, 8.2, 4.7, 7.1, 5.8, 9.4, 2.2, 1.7,
8.8, 4.5, 7.3, 5.6, 2.4, 5.2, 4.1, 3.3, 9.7, 4.2, 1.9, 5.6, 6.5,
7.7, 2.7, 8.8, 2.6, 0.6, 2.4, 7.8, 8.3, 8.7, 0.4, 6.3, 4.6, 0.3,
2.6, 3.7, 6.4, 0.9, 2.3, 2.3, 7.5, 0.5, 0.6, 6.5, 7.6, 0.4, 8.1,
0.2, 8, 0.6, 4.6, 3.2, 7.6, 3.3, 4.1, 4.9, 7.2, 8.2), x3 = c(2.3,
9.6, 6.2, 2, 6.8, 0.8, 7.9, 6.8, 9.9, 5.1, 0.8, 6.2, 8.7, 8.9,
3.5, 4.4, 1.3, 2.9, 8.4, 5.9, 3.6, 4.5, 0.2, 2.8, 2.8, 7.5, 1.4,
2.3, 1.7, 0.5, 7, 1.2, 7.4, 8.9, 8.6, 4.3, 1.1, 6.4, 2, 3.7,
1, 9.9, 4.8, 8.8, 1.3, 3.9, 4.8, 8.1, 6.7, 6.8, 9.2, 8.4, 1.5,
9.8, 6.6, 4.8, 4.6, 2.2, 7.9, 1.7, 3.5, 0.9, 2.1, 8.8, 2.1, 6.9,
2.5, 0.1, 8.2, 0.8, 2.4, 4.7, 6.6, 0.1, 7.3, 4.7, 3.7, 4.9, 4.3,
6.9, 2, 9.9, 4.1, 5.5, 2, 3.8, 4, 7.7, 8.4, 7, 6.8, 5.1, 7.2,
3.9, 5.7, 8.4, 6.5, 3.9, 4.5, 9.4), x4 = c(5.7, 8.9, 5.9, 3.4,
1.6, 7.7, 4.9, 3.8, 3.6, 2.6, 6.1, 6.9, 9.1, 9.3, 3.1, 7.4, 9.6,
9.8, 5.9, 1, 3.8, 7.2, 3.8, 6.7, 2.1, 6.3, 3, 9.9, 1.7, 8, 1.1,
0.2, 3.7, 7.7, 2.2, 5.5, 5.7, 5.1, 9.1, 7.1, 6.1, 7.4, 5.4, 8.4,
7.2, 7.9, 7.4, 6.1, 6.7, 4.1, 0.9, 6.7, 2.9, 0, 5.9, 0.4, 7.5,
0.3, 9, 6.3, 5, 7.2, 1.9, 3.5, 8.7, 5.4, 7, 4.2, 9.5, 2.4, 6,
4.5, 8.2, 1.2, 1.5, 6.5, 4.1, 2.7, 2.7, 7.7, 9.6, 6.4, 8.6, 8.6,
1.9, 3.3, 4.6, 9.2, 7.7, 8, 6.3, 4.8, 7.9, 1.8, 6.5, 6.9, 3.2,
3.1, 2.7, 8.7), x5 = c(6.4, 9.2, 3.3, 8.9, 9.8, 9.1, 0.1, 2.7,
1.2, 3.7, 10, 3.9, 1, 2, 2.1, 4.9, 5.2, 8.5, 7.3, 6.8, 6.8, 8.4,
6.3, 7.4, 10, 3.8, 7, 4.3, 4.3, 3, 0.1, 1.2, 2.3, 4.7, 4.6, 8.3,
5.5, 2.9, 1.9, 4.7, 5, 4.6, 8, 3.1, 8.2, 9.5, 2.5, 4.1, 9.1,
9.4, 8.2, 7.9, 9.1, 2.7, 7, 4.5, 0.7, 0.6, 1.1, 4.1, 3.5, 3.9,
10, 4.9, 2.8, 1.8, 4.4, 5.2, 3.3, 4.8, 8.7, 1.1, 1.2, 1.3, 4,
0.6, 0.3, 9.3, 0.3, 1.3, 1.2, 3.3, 1.4, 1.7, 10, 9.1, 8.6, 6.6,
5.5, 7.3, 6.2, 3.6, 3.6, 2.4, 3.7, 9.6, 3.7, 7.3, 4.6, 8.3)), .Names = c("y",
"x1", "x2", "x3", "x4", "x5"), row.names = c(NA, -100L), class = "data.frame")
```

```
# Avec un scatterplot matrix on voit que la relation entre y et x5 est fortement non
# linéaire en U inversé. Les autres relations semblent à peu près linéaires sauf parfois
# aux extrémités.
```

```
source("/home/gilles/stats/mytoolbox.R")
pairs2(d)
```



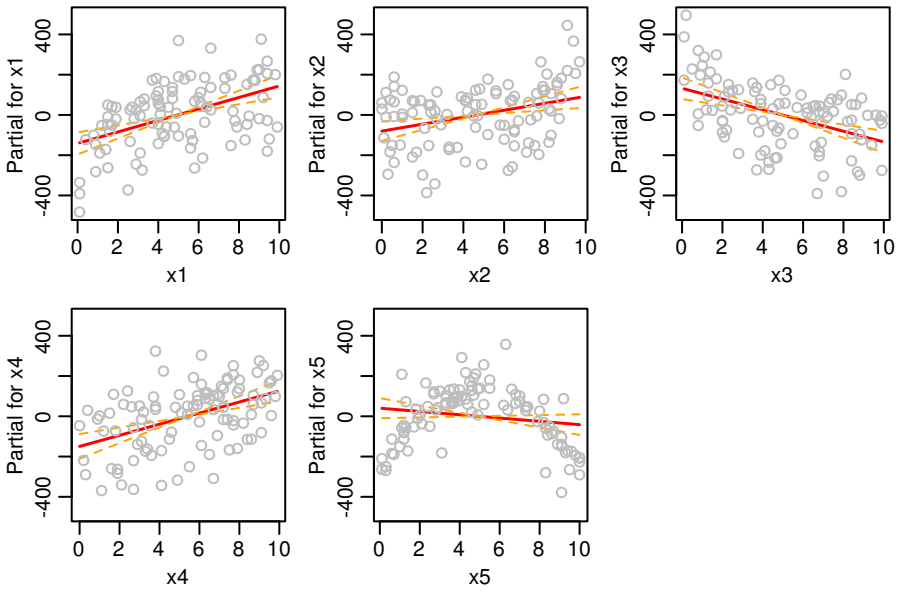

```
# Dans ce modèle basique toutes les variables sont déjà significatives sauf la variable
# x5. Il semble pourtant qu'il y ait bien une relation entre les deux. Les variables
# x1, x2 et x3 on une relation positive avec y et x3 une relation négative. Le R² est de
# 0.45 et l'erreur standard résiduelle de 148. Le R² ajusté est assez proche du R² ce qui
# indique que le modèle n'est pas fortement sursaturé.
mod <- lm(y ~ x1 + x2 + x3 + x4 + x5, data=d)
summary(mod)
```

```
##
## Call:
## lm(formula = y ~ x1 + x2 + x3 + x4 + x5, data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -344.07 -107.80   42.96  102.94  368.37
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -512.844     63.719  -8.049 2.53e-12 ***
## x1            28.551      5.511   5.181 1.26e-06 ***
## x2            17.248      5.208   3.312  0.00132 **
## x3           -26.651      5.421  -4.916 3.73e-06 ***
## x4            27.571      5.651   4.879 4.34e-06 ***
## x5            -8.171      5.063  -1.614  0.10993
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 148.1 on 94 degrees of freedom
## Multiple R-squared:  0.4481, Adjusted R-squared:  0.4187
## F-statistic: 15.26 on 5 and 94 DF,  p-value: 5.802e-11
```

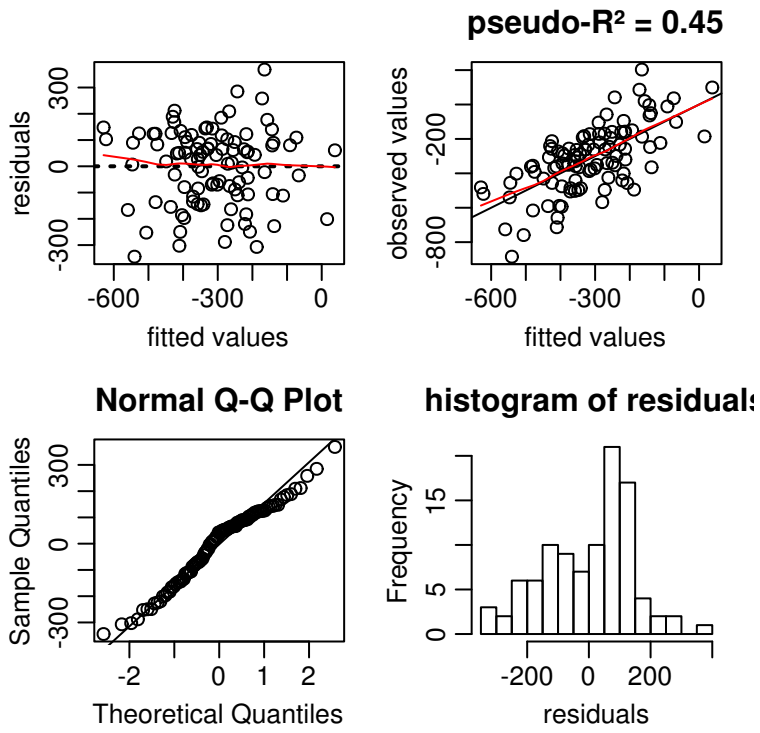
```
# Le termplot permet de visualiser le sens des relations et le fait que l'absence de
# relation avec x5 est dû à un mauvais ajustement du modèle
```

```
par(mfrow = c(2,3), mar = c(3,3,0.5,0.5), mgp = c(1.7, 0.6,0))
termplot(mod, partial.resid=T, se=TRUE)
```



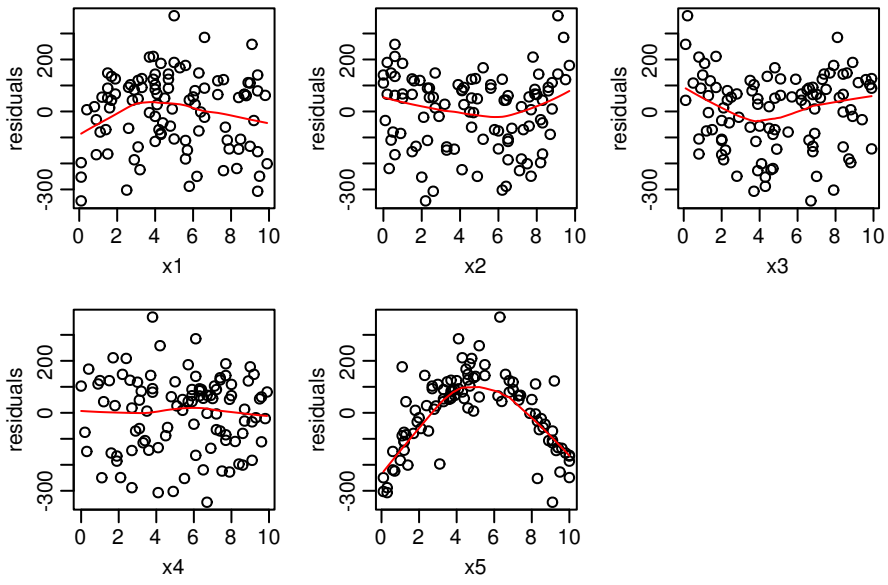

```
# La première série de plots diagnostiques ne montre aucun problème majeur.
# Le plot résidus vs fitted values ne montre aucun pattern laissant apparaître des
# relations non linéaires, la variance des résidus est à peu près homogène, il n'y a pas
# de valeurs extrêmes. Le modèle prédit correctement les données avec un R² de 0.45.
# Les deux plots de la ligne inférieure montrent que la distribution des résidus ne
# s'écarte pas de manière majeure de la normalité. On remarque cependant un léger excès
# de résidus proches de zéro et un léger déficit dans les résidus à grandes valeurs
# positives.
```

```
diagplot(mod)
```



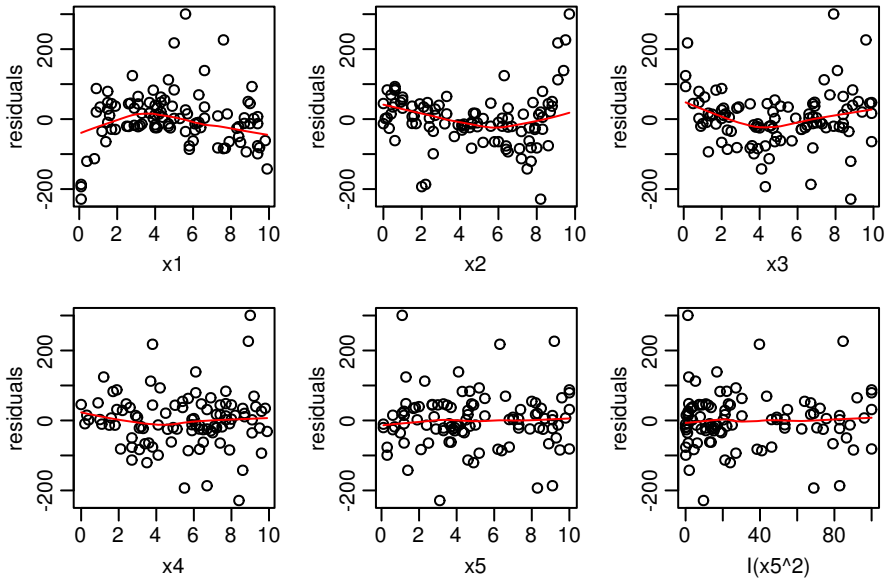

```
# Dans les plots des résidus en fonction des variables explicatives, on peut voir qu'il
# n'y a aucune valeur extrême (qui peuvent parfois être invisibles sur les graphiques
# résidus vs valeurs prédites). Par contre on voit clairement les problèmes de non
# linéarité en particulier pour x5.
```

```
diagplot2(mod)
```



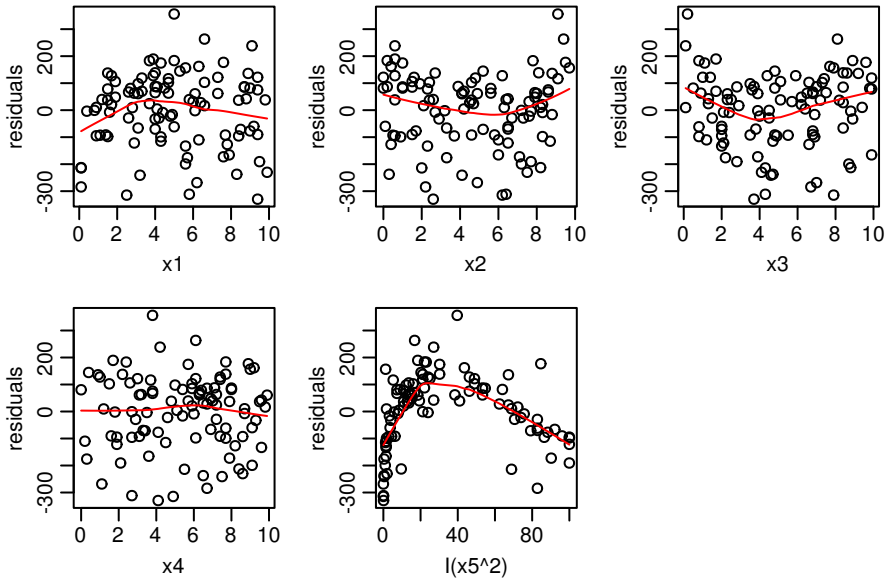

```
# La variable x5 a une relation nulle avec y (cfr summary du modèle, termplot,
# ou scatterplot matrix) et une forme de cloche typique des relations polynomiales
# d'ordre 2. On va donc rajouter une terme quadratique x5^2 en plus de x5. Notez
# l'utilisation de I() pour éviter que R interprète le ^2 comme une interaction double.
```

```
mod <- lm(y ~ x1 + x2 + x3 + x4 + x5 + I(x5^2), data=d)
diagplot2(mod)
```



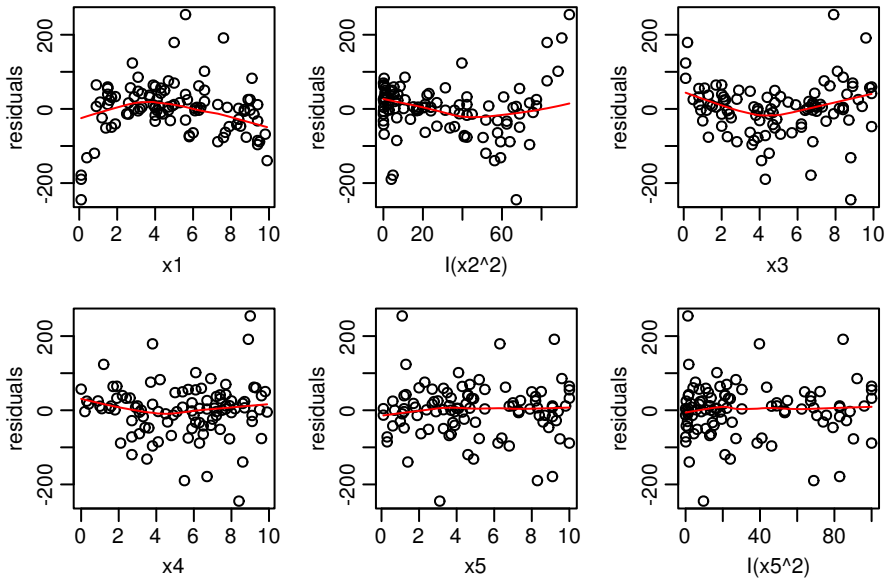

```
# Notez que si vous transformez juste la variable x5 sans utiliser de polynôme,
# çà ne fonctionne pas du tout !
```

```
mod <- lm(y ~ x1 + x2 + x3 + x4 + I(x5^2), data=d)
diagplot2(mod)
```

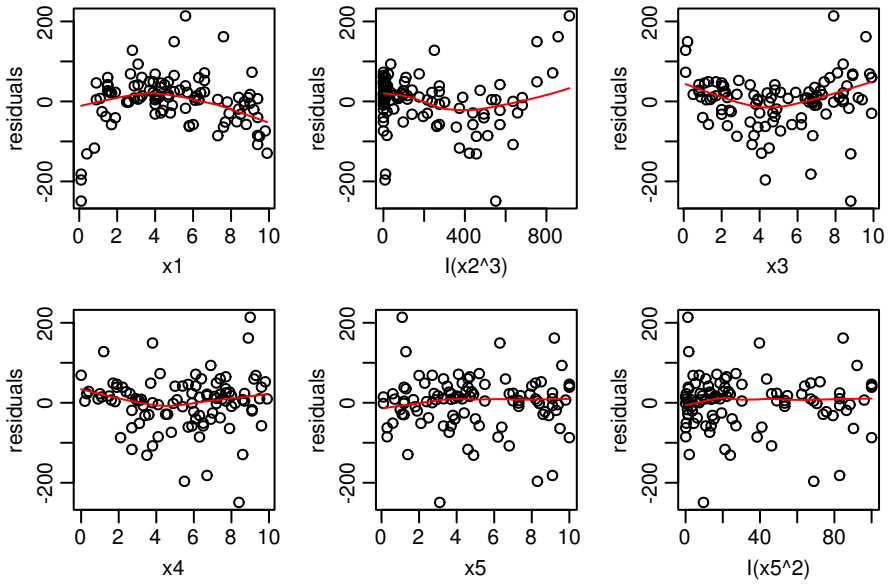



```
# Dans l'avant dernier graphique, la correction polynomiale a très bien fonctionné.
# On voit cependant que la relation est légèrement non linéaire pour X1, x3 et en
# particulier pour x2. La relation avec y est positive (cfr summary(mod),
# termplot etc.... ceci n'est évidemment pas visible sur ces graphiques de résidus) avec
# une courbure vers le bas. On se trouve donc dans le cadrant inférieur droit de la règle
# de Mosteller & Tukey. On peut donc essayer de transformer x2 avec des exposants >1.
```

```
mod <- lm(y ~ x1 + I(x2^2) + x3 + x4 + x5 + I(x5^2), data=d)
diagplot2(mod)
```

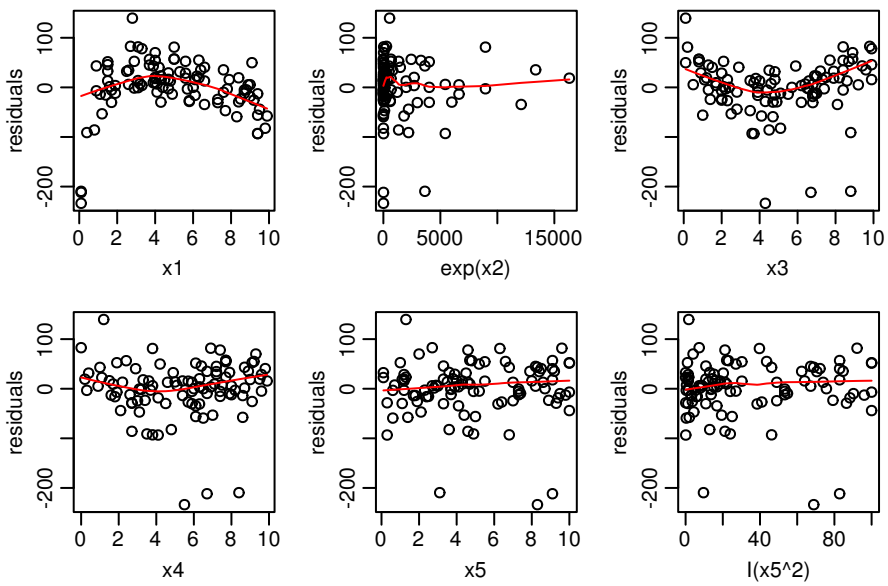



```
mod <- lm(y ~ x1 + I(x2^3) + x3 + x4 + x5 + I(x5^2), data=d)
diagplot2(mod)
```



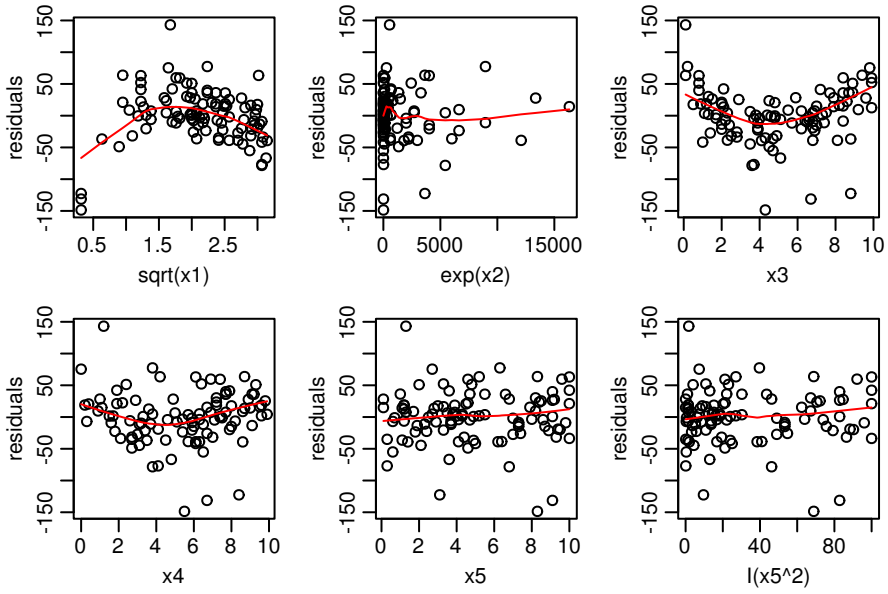

```
# Ca ne fonctionne pas très bien. Une autre possibilité serait de transformer les y avec
# une racine carrée, un log etc. Mais dans ce cas on modifierait aussi la relation avec
# les autres variables ce qui n'est pas idéal. Prendre le log de y revient à peu près
# à prendre exp(x) (e exposant x) ce qui fonctionne assez bien ici  même si çà a tendance
# à tasser les valeurs proches de 0 :
```

```
mod <- lm(y ~ x1 + exp(x2) + x3 + x4 + x5 + I(x5^2), data=d)
diagplot2(mod)
```

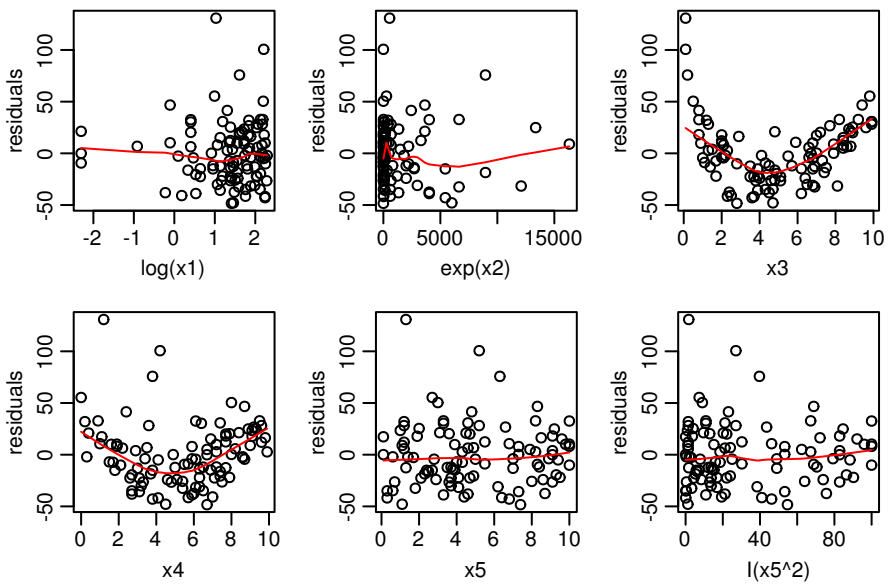



```
# La non linéarité dans la relation avec la variable x1 apparaît maintenant nettement.
# La relation est en cloche vers le haut et la relation y~x1 est positive, on se trouve
# donc dans le quadrant supérieur gauche du la règles de Mosteller & Tukey. On peut donc
# essayer des transformation log et racine carré (ou cubique etc) de x. Dans ce cas, la
# transformation racine carrée n'est pas assez forte et la transformation log semble
# bonne même si elle a tendance à ramasser les points vers la droite.
```

```
mod <- lm(y ~ sqrt(x1) + exp(x2) + x3 + x4 + x5 + I(x5^2), data=d)
diagplot2(mod)
```

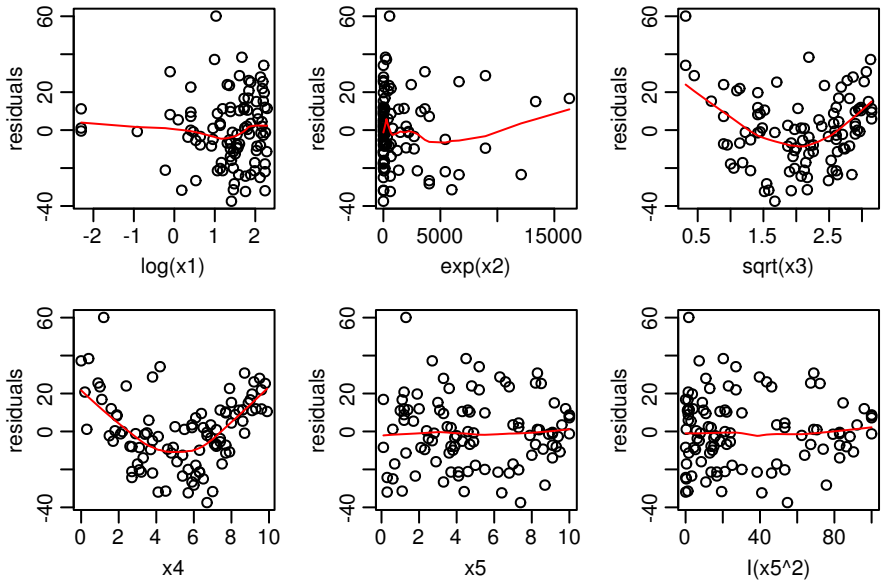



```
mod <- lm(y ~ log(x1) + exp(x2) + x3 + x4 + x5 + I(x5^2), data=d)
diagplot2(mod)
```



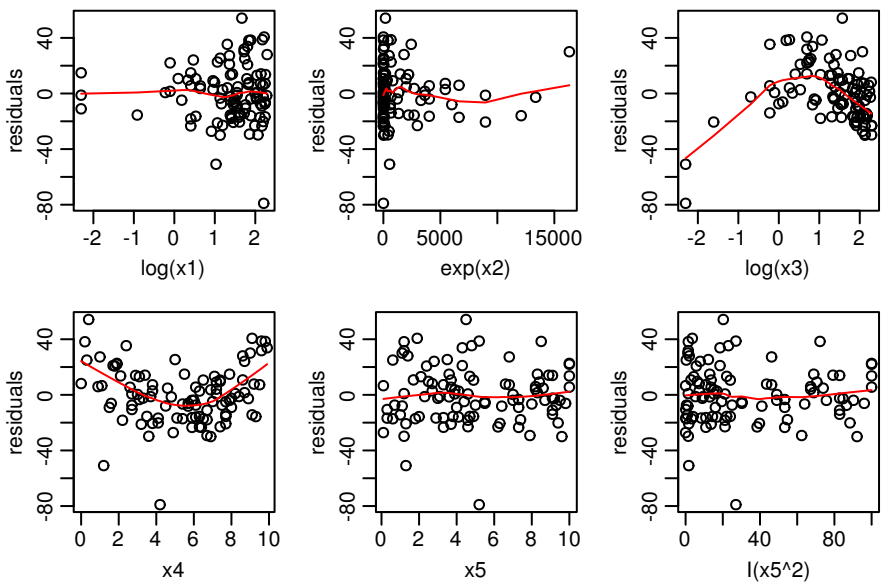

```
# Apparaissent maintenant des problèmes de non linéarité très clairs pour x3
# La relation y ~ x3 est plutôt négative, on se trouve donc
# ici dans le quadrant inférieur gauche de la règle de Mosteller & Tukey.
# On peut donc essayer une transformation racine carrée, qui semble insuffisante ici :
```

```
mod <- lm(y ~ log(x1) + exp(x2) + sqrt(x3) + x4 + x5 + I(x5^2), data=d)
diagplot2(mod)
```



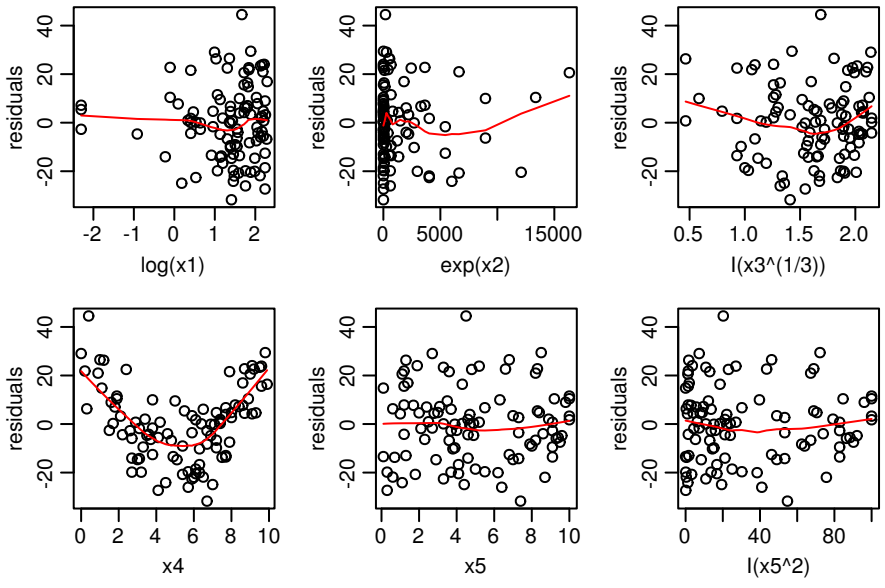

```
# On peut donc essayer une transformation log qui est plus forte, mais cette
# transformation est clairement trop forte ici  :
```

```
mod <- lm(y ~ log(x1) + exp(x2) + log(x3) + x4 + x5 + I(x5^2), data=d)
diagplot2(mod)
```

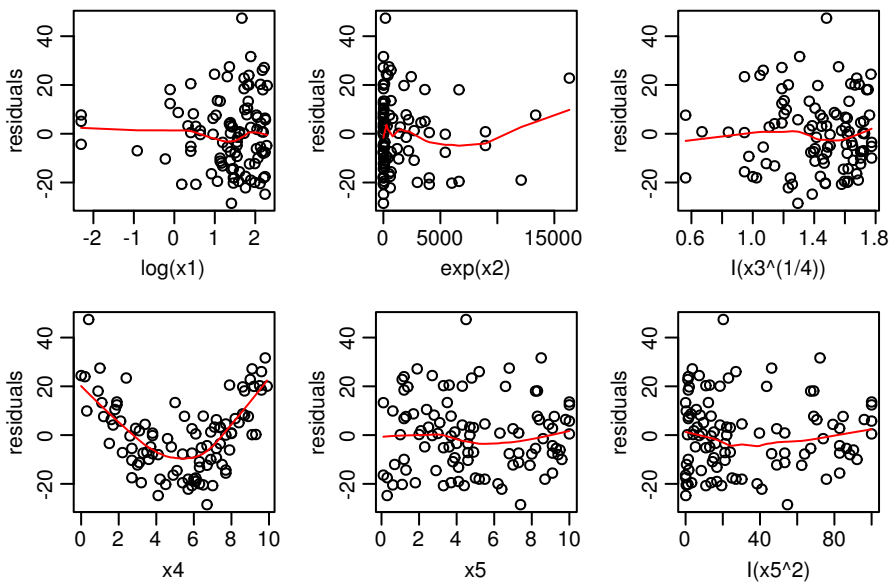



```
# La transformation racine carré est identique à un exposant 0.5 (1/2). On peut donc
# essayer d'autres fractions en exposant qui seront une transformation plus forte que
# la racine carrée mais moins forte que le log. Après quelques essais, on constate que
# l'exposant 0.25 fonctionne le mieux.
```

```
mod <- lm(y ~ log(x1) + exp(x2) + I(x3^(1/3)) + x4 + x5 + I(x5^2), data=d)
diagplot2(mod)
```




```
mod <- lm(y ~ log(x1) + exp(x2) + I(x3^(1/4)) + x4 + x5 + I(x5^2), data=d)
diagplot2(mod)
```

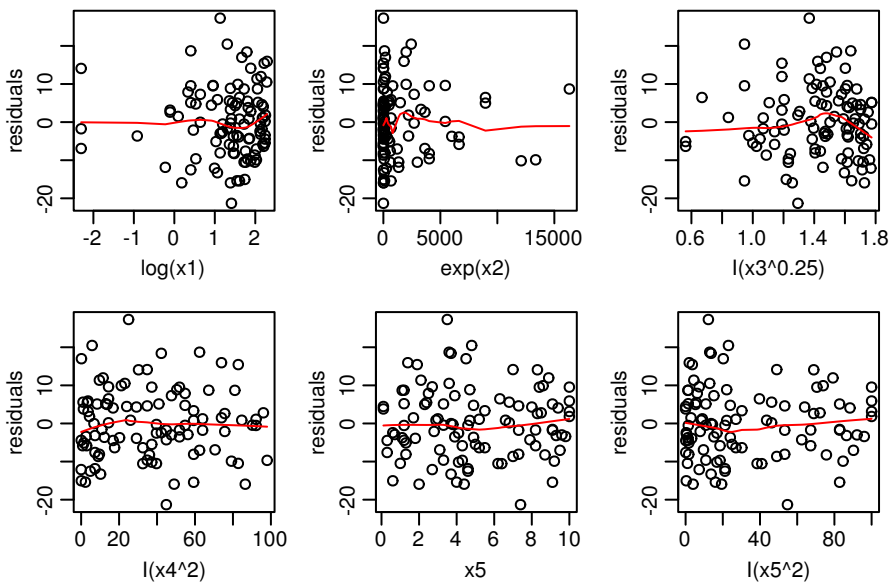



```
# Les problèmes apparaissent maintenant dans la variable x4 alors qu'ils étaient totalement
# invisibles au départ. La relation entre y etx4 est positive. Ce n'est pas très visible
# sur le scatterplot matrix car la relation y~x4 est masquée par les autres mais elle
# est très claire si on regarde le summary du modèle précédent :
summary(mod)
```

```
##
## Call:
## lm(formula = y ~ log(x1) + exp(x2) + I(x3^(1/4)) + x4 + x5 +
##     I(x5^2), data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -28.526 -10.992   0.088   7.888  47.409
##
## Coefficients:
##                  Estimate   Std. Error t value Pr(>|t|)
## (Intercept) -412.5362227   10.5042690  -39.27   <2e-16 ***
## log(x1)      105.2474442    1.7007979   61.88   <2e-16 ***
## exp(x2)        0.0251064    0.0005301   47.36   <2e-16 ***
## I(x3^(1/4)) -301.7969166    5.7285870  -52.68   <2e-16 ***
## x4            20.0196591    0.5843344   34.26   <2e-16 ***
## x5           147.5427312    2.1478155   68.69   <2e-16 ***
## I(x5^2)      -14.6906761    0.2002978  -73.34   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.3 on 93 degrees of freedom
## Multiple R-squared:  0.9942, Adjusted R-squared:  0.9938
## F-statistic:  2645 on 6 and 93 DF,  p-value: < 2.2e-16
```

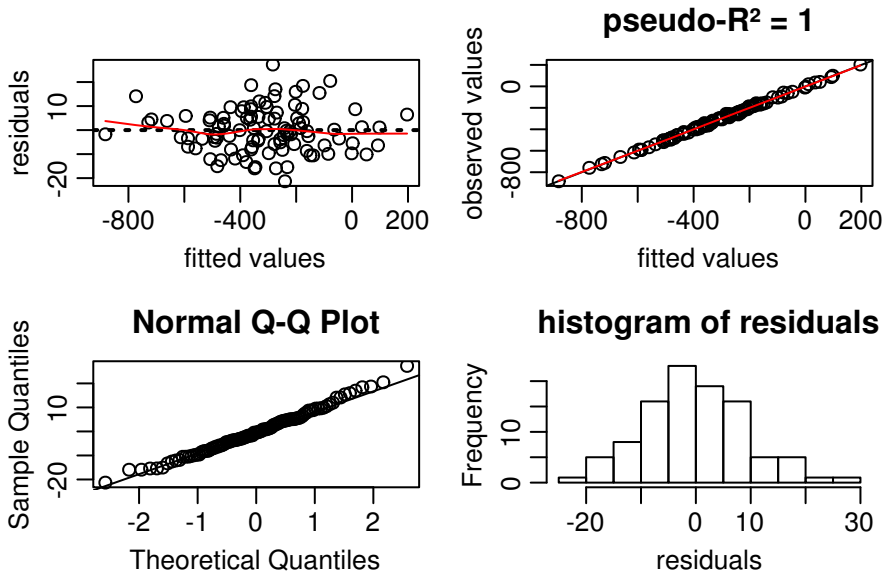
```
# On se trouve donc dans le quadrant inférieur droit de la règle de Mosteller & Tukey.
# On peut donc tenter divers exposants >1 pour x4. x4^2 semble parfaitement convenir.
```

```
mod <- lm(y ~ log(x1) + exp(x2) + I(x3^0.25) + I(x4^2) + x5 + I(x5^2), data=d)
diagplot2(mod)
```




```
# On a maintenant un modèle qui semble adéquat. Les résidus on maintenant une
# distribution parfaitement normale. Le R² est monté à 0.9978 et l'erreur standard
# résiduelle est descendue à 9.477. Le modèle est donc fortement amélioré.
```

```
diagplot(mod)
```



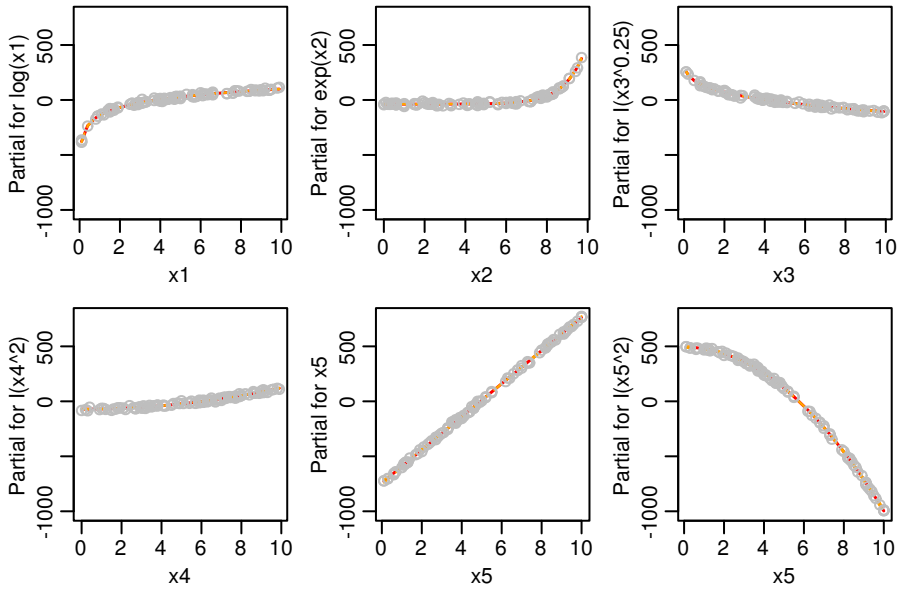

```
summary(mod)
```

```
##
## Call:
## lm(formula = y ~ log(x1) + exp(x2) + I(x3^0.25) + I(x4^2) + x5 +
##     I(x5^2), data = d)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -21.3073  -6.3601  -0.4747   5.2649  27.2865
```

```
##
## Coefficients:
##                 Estimate  Std. Error t value Pr(>|t|)
## (Intercept) -380.517307    6.425423  -59.22   <2e-16 ***
## log(x1)      103.743874    1.053162   98.51   <2e-16 ***
## exp(x2)        0.025421    0.000328   77.50   <2e-16 ***
## I(x3^0.25)  -302.067419    3.547687  -85.14   <2e-16 ***
## I(x4^2)        1.940917    0.034262   56.65   <2e-16 ***
## x5           150.467837    1.324836  113.58   <2e-16 ***
## I(x5^2)      -14.899024    0.123580 -120.56   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.477 on 93 degrees of freedom
## Multiple R-squared:  0.9978, Adjusted R-squared:  0.9976
## F-statistic:  6920 on 6 and 93 DF,  p-value: < 2.2e-16
```

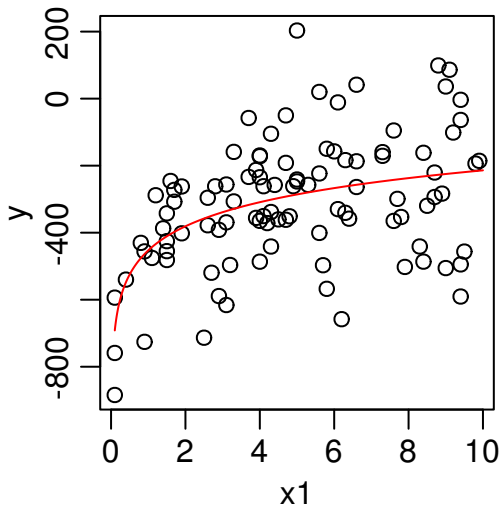
```
# Un termplot permet de visualiser rapidement les relations entre les variables
```

```
par(mfrow = c(2,3), mar = c(3,3,0.5,0.5), mgp = c(1.7, 0.6,0))
termplot(mod, partial.resid=T, se=TRUE)
```

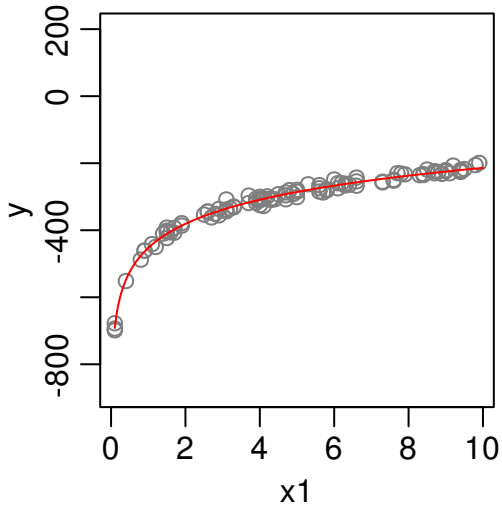



```
# Exemple simple avec juste x1
# On construit la matrice X qui contient les valeurs pour lesquelles on veut une
# prédiction. On fait varier uniquement x1 : 100 valeurs entre log(0.1) et log(10).
# pour les autres valeurs on prend leur moyenne.
X <- cbind(1, seq(log(0.1), log(10), length.out = 100),
           mean(exp(d$x2)), mean(d$x3^0.25), mean(d$x4^2), mean(d$x5), mean(d$x5^2))
# NB, de manière équivalente, on aurait aussi pu récupérer les moyennes des colonnes de
# mod$model
X <- apply(mod$model[,-1], 2, mean)
X <- sapply(X, rep, 100)
X[,1]<- seq(log(0.1), log(10), length.out = 100)
X <- cbind(1,X)

# Ensuite on trace le graphique. Notez que avant de tracer la ligne du modèle, on
# rétrotransforme x1 de façon à l'exprimer sur l'échelle d'origine (entre 0 et 10)
# plutôt que sur l'échelle log
```

```r
par(mfrow = c(1,1), mar = c(3,3,0.5,0.5), mgp = c(1.7, 0.6,0))
pred <- X %*% coef(mod)
plot(d$y ~ d$x1, ylab = "y", xlab = "x1")
lines(pred ~ exp(X[,2]), col = "red")
```



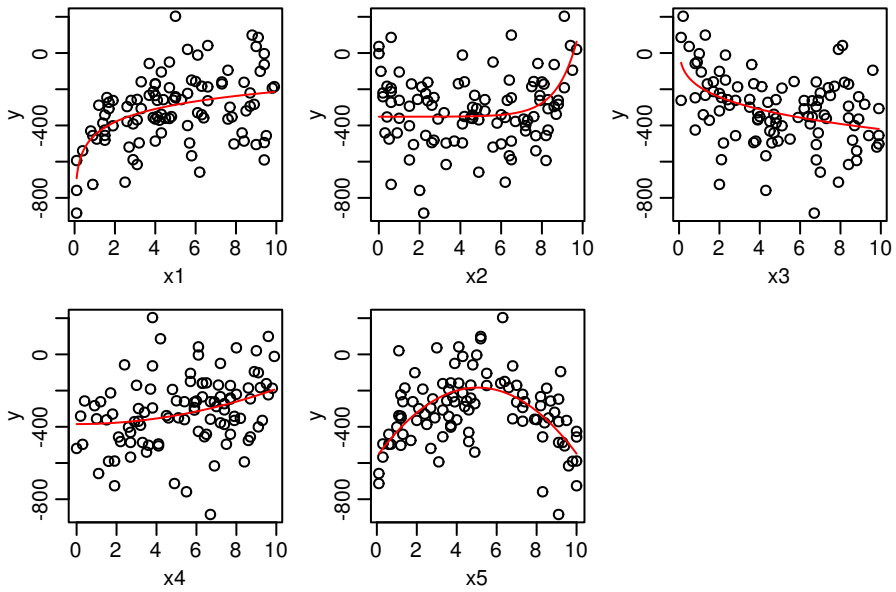

```r
# On peut aussi représenter les résidus partiels à la place des valeurs observées
Xpr <- X
Xpr[,2] <- mod$model[,2]
partialres <- Xpr %*% coef(mod) + resid(mod)

plot(d$y ~ d$x1, ylab = "y", xlab = "x1", type = "n")
points(partialres ~ d$x1, col = "grey50")
lines(pred ~ exp(X[,2]), col = "red")
```




```r
# Graphique du modèle et des données pour toutes les variables.
# On a adapté le code de l'exercice 5.
# La particularité est que avant de tracer les lignes de valeurs prédites, on
# rétro-transforme les x qui ont servi à faire la prédiction de façon à les présenter sur
# l'échelle d'origine. La régression polynomiale (x5) est un peu plus délicate et se règle
```

```r
# en dehors de la boucle

# Pour chaque variable explicative, on calcule les moyennes que l'on répète 1000 fois.
# Pour chaque variable, on génère 1000 nombres compris entre son maximum et son minimum.
# dans ce cas-ci, toutes les variables sont comprises entre 0 et 10, on aurait donc pu
# utiliser simplement ces valeurs.
means <- sapply(apply(mod$model[,-1], 2, mean), rep, times = 1000)
minmax <- rbind(apply(mod$model[,-1], 2, min), apply(mod$model[,-1], 2, max))
minmax <- apply(minmax, 2, function(x) seq(x[1], x[2], length.out = 1000))
obs <- d
```

```r
par(mfrow = c(2,3), mar = c(3,3,0.5,0.5), mgp = c(1.7, 0.6,0))

for( i in 2:5) {
    X <- cbind(1,means)
    X[,i] <- minmax[,i-1]
    pred <- X %*% coef(mod)
    se <- sqrt(diag(X %*% vcov(mod) %*% t(X)))
    X2 <- X
    plot(obs[,1] ~ obs[,i], ylab = colnames(obs)[1], xlab = colnames(obs[i]))
    X2[,2] <- exp(X2[,2]) ; X2[,3] <- log(X2[,3]) ; X2[,4] <- (X2[,4])^4 ;
        X2[,5] <- sqrt(X2[,5]); X2[,7] <- X2[,7]^0.5
    lines(pred ~ X2[, i], col = "red")
}


X <- cbind(1,means)
X[,6] <- minmax[,5]
X[,7] <- minmax[,5]^2
pred <- X %*% coef(mod)
plot(obs[,1] ~ obs[,6], ylab = colnames(obs)[1], xlab = colnames(obs[6]))
lines(pred ~ X[,6], col = "red")
```



Exercice 9a : GLM logistique à une variable explicative

On a relevé la présence ou l'abence d'une espèce de plante dans 100 sites pour lequels on dispose également d'une mesure de l'humidité du sol sur une échelle continue de 1 à 12 (très sec à aquatique).
Faites un modèle prédictif de la présence de cette espèce et une réprésentation graphique des données et du modèle.
Au moyen des valeurs prédites, déterminez dans quelle gamme de valeurs d'humidité, on a 80% de chance de trouver l'espèce (réponse : 7.3 - 9.5).

```
set.seed(1)
d <- data.frame(
    hum = round(runif(100, 1,12),2)
)
X <- model.matrix(~ hum + I(hum^2), data=d)
# B <- c(-1110, 315, -20)
# B <- c(-170, 33.6, -1.63)*0.5
B <- c(-60, 15.2, -0.92)*1

set.seed(2)
d$pres <- rbinom(100, size=1, prob= invlogit(X %*% B))
```

```
d <- data.frame(
    hum = c(3.92, 5.09, 7.3, 10.99, 3.22, 10.88, 11.39,
      8.27, 7.92, 1.68, 3.27, 2.94, 8.56, 5.23, 9.47, 6.47, 8.89, 11.91,
      5.18, 9.55, 11.28, 3.33, 8.17, 2.38, 3.94, 5.25, 1.15, 5.21,
      10.57, 4.74, 6.3, 7.6, 6.43, 3.05, 10.1, 8.35, 9.74, 2.19, 8.96,
      5.52, 10.03, 8.12, 9.61, 7.08, 6.83, 9.68, 1.26, 6.25, 9.06,
      8.62, 6.25, 10.47, 5.82, 3.69, 1.78, 2.09, 4.48, 6.7, 8.28, 5.48,
      11.04, 4.23, 6.05, 4.66, 8.16, 3.84, 6.26, 9.43, 1.93, 10.63,
      4.73, 10.23, 4.81, 4.67, 6.24, 10.81, 10.51, 5.29, 9.55, 11.57,
      5.78, 8.84, 5.4, 4.58, 9.33, 3.23, 8.82, 2.34, 3.7, 2.58, 3.64,
      1.65, 8.07, 10.64, 9.57, 9.77, 6.01, 5.51, 9.92, 7.65),
    pres = c(0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1,
     0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1,
     1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
     0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0,
     0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1)
    )
```

```
# La relation est ici clairement unimodale et correspond à un modèle polynomial d'ordre 2.
# Le message d'avis indique simplement que certaine valeurs prédites très petites ou
# très grandes ont été fixées à 0 ou 1 alors que c'est normalement impossible avec un
# lien logit. Il faut s'inquiéter si le mesage indique que l'algorithme n'a pas convergé.
mod <- glm(pres ~ hum + I(hum^2), data=d, family = binomial)
summary(mod)
```

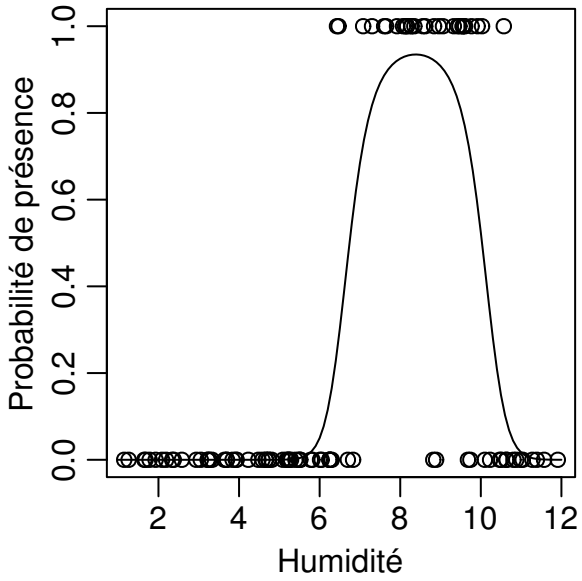
```
##
## Call:
## glm(formula = pres ~ hum + I(hum^2), family = binomial, data = d)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.26177  -0.11192  -0.00016   0.37565   2.07734
##
## Coefficients:
##             Estimate Std. Error z value  Pr(>|z|)
## (Intercept) -66.5830    15.6573  -4.253 0.0000211 ***
## hum          16.5150     3.8782   4.258 0.0000206 ***
## I(hum^2)     -0.9847     0.2330  -4.226 0.0000238 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 122.173  on 99  degrees of freedom
## Residual deviance:  42.683  on 97  degrees of freedom
## AIC: 48.683
##
## Number of Fisher Scoring iterations: 9
```

```
# Il s'agit d'un modèle basé sur des données binaires. On ne devrait donc pas avoir
# de surdispersion. On observe une légère "sous-dispersion". Les résultats obtenus sont
# donc conservateurs. En général on ne corrige pas la sous-disspersion
overdisp(mod)
```

```
## pearsonresid     deviance
##    0.5399798    0.4400286
```

```
# prédictions et représentation graphique
X <- cbind(1, seq(1, 12, 0.1), seq(1, 12, 0.1)^2)
pred <- invlogit(X %*% coef(mod))

par(mfrow = c(1,1), mar = c(3,3,0.5,0.5), mgp = c(1.7, 0.6,0))
plot(pres ~ hum, data=d, xlab = "Humidité", ylab = "Probabilité de présence")
lines(pred ~ X[,2])
```




```
# Dans quelle gamme de valeurs d'humidité, a-t-on une probabilité de préence >= 0.8 ?
pred <- data.frame( hum = seq(1, 12, 0.1), prob = pred)
min(pred[pred$prob >= 0.80,"hum"])
```

```
## [1] 7.3
```

```
max(pred[pred$prob >= 0.80,"hum"])
```

```
## [1] 9.5
```

Exercice 9b : GLM logistique à deux variables explicatives

On a relevé la présence ou l'abence d'une espèce de plante dans 300 sites pour lequels on dispose également d'une mesure de l'humidité du sol sur une échelle continue de 1 à 12 (très sec à aquatique) et d'acidité du sol sur une échelle continue de 1 à 9 (très acide à très basique). NB : Dans ce cas l'analyse par GLM reste faiable mais pour ce genre de modèles les Generalized Additive Models (GAM) peuvent vite se montrer assez utiles.

- Faites un modèle prédictif de la présence de cette espèce et une réprésentation graphique classique des données et du modèle sous forme de courbe (fonction lines).
  Pour la représentation graphique, vous pouvez par exemple représenter les valeurs prédites en fonction de l'humidité pour des valeurs de ph de 3, 5, 7 et en fonction du ph pour des valeurs d'humidité de 5, 7, 9 et 11.
- la représentation graphique précédente n'est pas optimale dans ce cas. L'idéal est une représentation en pseudo-3D. Pour ce faire estimez les valeurs prédites de probabilité présence en fonction de toutes les combinaisons possibles de humidité et ph. Pour construire une matrice avec toutes ces combinaisons possibles, vous pouvez vous aider de la fonction expand.grid. Vous pouvez enuite faire eune représentation avec hum en x, ph en y et des couleurs pour représenter la probabilité prédite. Vous pouvez utiliser la fonction image ou filled.contour. Vous pouvez aussi explorer les fonctions persp ainsi que persp3D du package rgl. Voir les exemples dans les transsparents de la formation R (partie sur les graphiques). Pour utiliser ces fonctions vous devrez transformer votre vecteur de valeurs prédites en une matrice dont les colonnes correspondent aux valeurs de hum, le lignes aux valeurs de ph et la matrice elle même contient les valeurs prédites. (voir exemples de image avec le jeu de données volcano)
- Au moyen des valeurs prédites, déterminez dans quelle gamme de valeurs d'humidité, et de pH on a 80% de chance de trouver l'espèce (réponse : humidité : 7.1 - 10.2, ph : 1.8 - 4.6).

```
d <- data.frame(
    hum = round(runif(300, 1,12),2),
    ph = round(runif(300, 1,9),2)
)
X <- model.matrix(~ (hum + I(hum^2) + ph + I(ph^2) + hum:ph + I(hum^2):ph + hum:I(ph^2) +
                         I(hum^2):I(ph^2)) , data=d)
B <- c(-19.33, 3.57, -0.21, -14.1, 1.76, 4.07, -0.23, -0.53, 0.03)*1

d$pres <- rbinom(300, size=1, prob= invlogit(X %*% B))
```

```
d <- structure(list(hum = c(4.55, 1.49, 9.05, 6.42, 3.59, 2.39, 4.08,
11.14, 7.44, 4.86, 10.83, 4.03, 3.43, 4.31, 10.76, 2.88, 2.39,
2.6, 11.22, 1.58, 1.73, 9.69, 7.12, 3.83, 1.66, 9.38, 11.84,
3.56, 7.78, 10.11, 8.39, 4.98, 11.62, 10.44, 9.11, 8.96, 6.45,
1.1, 9.7, 1.85, 7.16, 8.67, 9.72, 9.32, 8.24, 6.65, 7.12, 10.9,
6.05, 11.98, 9.35, 5.03, 10.36, 8.84, 6.72, 1.47, 11.56, 5.98,
6.73, 3.06, 1.21, 3.12, 4.91, 4.17, 11.5, 7.54, 10.62, 8.96,
6.04, 8, 4.33, 11.54, 11.3, 2.72, 5.09, 10.08, 10.84, 1.16, 9.17,
1.71, 6.9, 6.49, 6.22, 3.95, 3.22, 9.51, 11.54, 7.35, 2.83, 1.9,
8.86, 2.6, 4.59, 11.7, 2.5, 4.45, 2.15, 2.27, 11.83, 7.79, 7.96,
3.84, 8.75, 3.2, 10.14, 3.64, 11.15, 11.94, 5.77, 6.08, 9.63,
10.34, 3.92, 3.03, 11.76, 4.47, 11.24, 6.06, 7.22, 9.31, 9.48,
5.25, 3.45, 2.07, 11.21, 4.71, 1.97, 2.25, 2.23, 3.64, 11.17,
10.4, 11.96, 7.31, 11.1, 11.69, 10.51, 5.11, 5.32, 4.33, 10.63,
11.12, 9.41, 11.5, 6.78, 9.94, 6.19, 11.01, 10.72, 9.64, 2.41,
4.19, 10.42, 4.7, 6.72, 2.99, 9.69, 7.2, 11.01, 11.16, 4.2, 5.82,
9.11, 11.38, 5.1, 1.22, 9.81, 1.09, 6.57, 2.14, 5.1, 9.78, 3.07,
8.78, 5.07, 4.87, 6.45, 4.48, 11.17, 5.48, 3.48, 2.73, 8.59,
5.49, 3.12, 9.95, 6.01, 9.87, 7.37, 11.68, 1.55, 3.73, 9.62,
7.28, 5.83, 7.04, 1.65, 5.72, 2.57, 2.03, 2.46, 5.02, 6, 6.91,
3.04, 5.94, 3.82, 7.8, 10.69, 4.5, 10.12, 11.58, 6.97, 10.03,
11.38, 10.17, 9.75, 3.51, 3.01, 3.24, 4.87, 10.74, 2.23, 11.25,
8.68, 9.47, 8.77, 9.58, 7.07, 5.65, 5.42, 9.52, 10.15, 2.38,
2.12, 11.6, 4.15, 7.82, 7.73, 7.18, 4.24, 9.13, 9.55, 7.71, 7.02,
10.47, 6.9, 4.6, 10.95, 1.04, 8.01, 10.6, 3.34, 2.3, 1.37, 9.61,
10.51, 8.69, 1.8, 10.72, 5.61, 11.18, 9.61, 1.81, 2.76, 11.94,
10.88, 1.25, 7.43, 10.26, 3.63, 3.8, 11.78, 7.03, 3.29, 9.21,
6.38, 3.88, 9.85, 8.52, 4.54, 1.97, 1.27, 1.8, 1.71, 2.72, 3.14,
6.87, 4.46, 5.17, 9.58, 8.48, 2.29, 10.2, 8.1, 11.19, 2.9, 6.76,
4, 10.18), ph = c(8.24, 1.99, 8.67, 4.76, 6.33, 8.5, 2.18, 2.86,
8.47, 8.73, 5.39, 2.91, 3.88, 6.53, 2.08, 8.24, 2.11, 3.83, 2.31,
3.08, 2.5, 7.44, 8.81, 4.63, 8.66, 6.61, 2.35, 5.06, 8.01, 1.72,
3.21, 8.95, 7, 3.02, 3.19, 4.4, 1.38, 5.68, 8.15, 2.51, 1.5,
1.06, 6.11, 5.27, 3.04, 6, 8.92, 6.04, 4.95, 2.6, 2.75, 1.7,
2.89, 2.58, 5.28, 4.49, 3.55, 1.33, 1.33, 1.54, 1.43, 2.48, 5.76,
4.54, 5.49, 1.23, 3.21, 7.77, 3.21, 2.1, 6.3, 2.26, 3.55, 2.22,
4.08, 5.71, 3.73, 4.67, 4.23, 2.36, 5.46, 1.47, 2.61, 7.56, 7.71,
8.69, 3.61, 5.75, 5.46, 8.92, 5.15, 4.67, 2.79, 1.61, 3.64, 8.39,
7.15, 1.8, 6.9, 1.06, 8.14, 1.82, 5.73, 3.73, 4.01, 8.12, 8.47,
4.3, 5.22, 1.69, 8.27, 7.21, 5.58, 8.11, 4.69, 4.54, 2.58, 4.02,
5.85, 5.35, 1.1, 8.96, 8.05, 6.99, 7.53, 7.27, 6.88, 4.44, 5.79,
1.29, 6.68, 4.06, 7.63, 5.94, 2.9, 2.8, 8.77, 2.74, 8.35, 3.46,
5.36, 3.87, 6.17, 8.22, 8.43, 7.44, 1.86, 5.46, 2.93, 8.06, 7.46,
5.7, 4.7, 7.78, 8.17, 2.03, 1.03, 2.03, 2.46, 8.88, 7.09, 7.94,
8.86, 1.17, 7.96, 4.6, 6.18, 6.51, 7.93, 4.28, 6.91, 6.39, 3.88,
2.96, 1.33, 8.23, 8.77, 5.92, 3.96, 5.94, 5.66, 1.48, 8.67, 6.73,
7.69, 4.35, 6.7, 3.62, 3.86, 2.54, 3.63, 3.33, 3.8, 4.9, 8.73,
```

```
5.19, 7.41, 5.65, 6.04, 7.75, 6.31, 5.16, 3.65, 5.93, 5.65, 5.76,
1.47, 6.92, 4.94, 8.99, 5.33, 1.33, 2.62, 3.88, 1.93, 5.47, 2.88,
3.5, 5.01, 3.99, 3.93, 1.02, 6.07, 4.29, 3.84, 5.89, 4.95, 6.7,
8.96, 4.97, 6.99, 7.73, 5.44, 5.74, 2.22, 1.15, 4.09, 7.72, 2.58,
6.75, 7.97, 2.79, 4.25, 3.7, 3.25, 8.94, 6.15, 1.23, 3.27, 7.52,
6.43, 2.48, 1.23, 7.85, 5.65, 2.56, 4.74, 3.86, 1.52, 8.46, 4.96,
2.71, 2, 6.9, 5.37, 4.84, 6.09, 3.16, 6.66, 5.34, 3.42, 3.73,
8.22, 7.93, 7.64, 5.86, 4.69, 5.43, 6.23, 7.92, 8.5, 1.05, 6.15,
4.1, 2.04, 6.17, 4.38, 2.18, 4.17, 3.12, 3.79, 6.55, 5.06, 1.7,
4.08, 8.88, 8.03, 5.98, 8.94, 7.91), pres = c(0L, 0L, 0L, 1L,
0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 1L, 0L, 0L, 0L, 0L, 0L,
0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 1L, 0L, 0L, 1L, 1L, 1L,
0L, 0L, 0L, 0L, 0L, 1L, 0L, 0L, 1L, 0L, 0L, 0L, 0L, 0L, 1L, 0L,
1L, 1L, 0L, 0L, 0L, 0L, 1L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L,
0L, 1L, 0L, 0L, 1L, 0L, 0L, 0L, 1L, 0L, 1L, 0L, 1L, 0L, 0L, 0L,
0L, 0L, 0L, 0L, 0L, 0L, 1L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L,
0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L,
0L, 1L, 0L, 0L, 1L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 1L,
0L, 0L, 0L, 1L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L,
0L, 0L, 0L, 0L, 1L, 0L, 0L, 0L, 0L, 1L, 0L, 0L, 0L, 0L, 0L, 0L, 0L,
0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 1L, 0L, 0L, 0L, 0L, 0L, 0L, 0L,
0L, 0L, 0L, 0L, 0L, 1L, 0L, 1L, 1L, 0L, 0L, 0L, 1L, 0L, 0L, 0L,
0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L,
0L, 1L, 0L, 0L, 1L, 0L, 0L, 0L, 0L, 0L, 0L, 1L, 1L, 0L, 0L, 0L,
0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 1L, 0L, 0L, 1L, 0L, 1L,
1L, 0L, 0L, 0L, 0L, 0L, 0L, 1L, 0L, 0L, 0L, 1L, 0L, 1L, 0L, 0L,
0L, 0L, 1L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 1L,
0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 1L, 0L,
0L, 1L, 1L, 0L, 0L, 0L, 0L, 0L)), .Names = c("hum", "ph", "pres"
), row.names = c(NA, -300L), class = "data.frame")
```

```
# On a ici ausi clairement une relation polynomiale d'ordre 2 pour les deux variables
# explicatives
mod <- glm(pres ~ hum + I(hum^2) + ph + I(ph^2), data = d, family = binomial)
summary(mod)
```

```
##
## Call:
## glm(formula = pres ~ hum + I(hum^2) + ph + I(ph^2), family = binomial,
##     data = d)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.92410  -0.13467  -0.00058   0.00000   2.29936
##
## Coefficients:
##             Estimate Std. Error z value    Pr(>|z|)
## (Intercept) -47.6544     9.0551  -5.263 0.000000142 ***
## hum          10.0770     2.0139   5.004 0.000000562 ***
## I(hum^2)     -0.5824     0.1150  -5.066 0.000000407 ***
## ph            4.3659     1.0303   4.237 0.000022616 ***
## I(ph^2)      -0.6834     0.1488  -4.591 0.000004405 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 253.625  on 299  degrees of freedom
## Residual deviance:  95.347  on 295  degrees of freedom
```

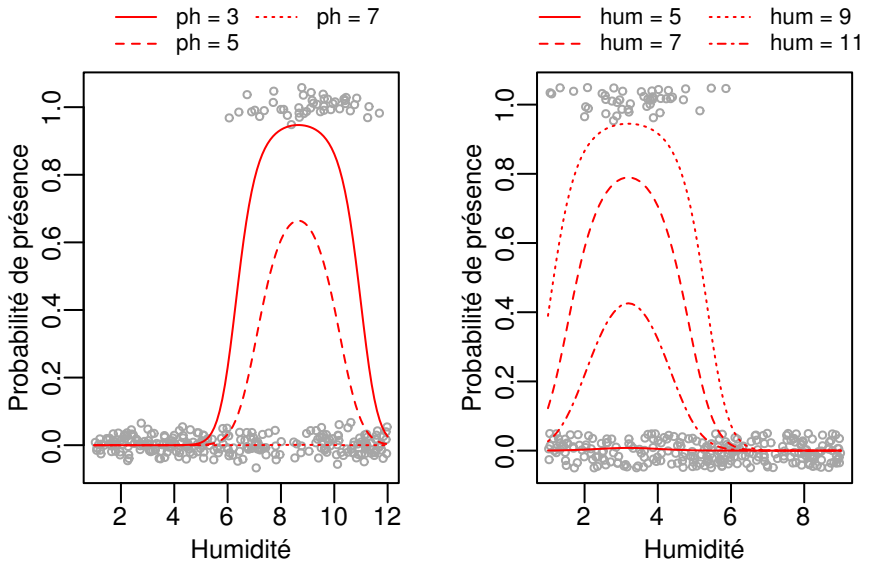
```
## AIC: 105.35
##
## Number of Fisher Scoring iterations: 9
```

```r
overdisp(mod)
```

```
## pearsonresid     deviance
##    0.3548289    0.3232112
```

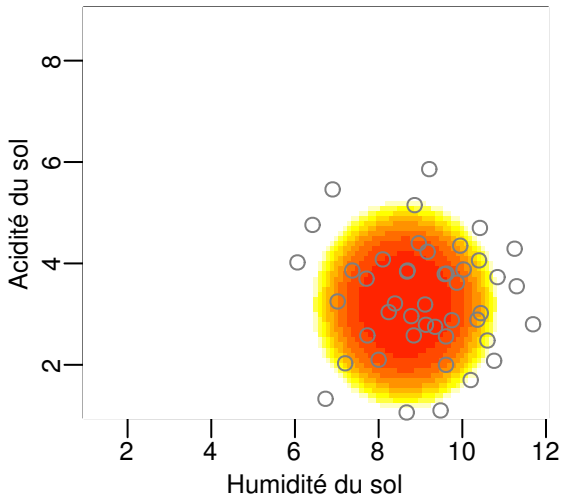
```r
# graphique classique avec laes valeurs prédites le long d'une variable explicative
# pour plusieurs valeurs fixées de l'autre variable.
```

```r
# dev.new(width = 12/2.54, height = 8/2.54)
par(mfrow = c(1,2), mar = c(2.5,2.5,2.5,1.1), mgp = c(1.2, 0.35,0), cex.axis = 0.8, cex.lab = 0.8)

X3 <- cbind(1, seq(1, 12, 0.1), seq(1, 12, 0.1)^2, 3, 3^2)
X5 <- cbind(1, seq(1, 12, 0.1), seq(1, 12, 0.1)^2, 5, 5^2)
X7 <- cbind(1, seq(1, 12, 0.1), seq(1, 12, 0.1)^2, 7, 7^2)
pred3 <- invlogit(X3 %*% coef(mod))
pred5 <- invlogit(X5 %*% coef(mod))
pred7 <- invlogit(X7 %*% coef(mod))

plot(I(pres+ rnorm(nrow(d), 0, 0.025)) ~ hum, data=d, col = "grey65", cex = 0.5,
     xlab = "Humidité", ylab = "Probabilité de présence")
lines(y = pred3, x = X3[,2], lty = 1, col = "red")
lines(y = pred5, x = X5[,2], lty = 2, col = "red")
lines(y = pred7, x = X7[,2], lty = 3, col = "red")

legend(x = "top", inset = -0.2, xpd = NA, bty = "n", lty = 1:3, ncol = 2, col = "red",
       legend = c("ph = 3", "ph = 5", "ph = 7"), cex = 0.7)

X5 <- cbind(1, 5, 5^2, seq(1, 9, 0.1), seq(1, 9, 0.1)^2)
X7 <- cbind(1, 7, 7^2, seq(1, 9, 0.1), seq(1, 9, 0.1)^2)
X9 <- cbind(1, 9, 9^2, seq(1, 9, 0.1), seq(1, 9, 0.1)^2)
X11 <- cbind(1, 11, 11^2, seq(1, 9, 0.1), seq(1, 9, 0.1)^2)

pred5 <- invlogit(X5 %*% coef(mod))
pred7 <- invlogit(X7 %*% coef(mod))
pred9 <- invlogit(X9 %*% coef(mod))
pred11 <- invlogit(X11 %*% coef(mod))

plot(jitter(pres,amount=0.05) ~ ph, data=d, col = "grey65", cex = 0.5,
     xlab = "Humidité", ylab = "Probabilité de présence")
lines(y = pred5, x = X5[,4], lty = 1, col = "red")
lines(y = pred7, x = X7[,4], lty = 2, col = "red")
lines(y = pred9, x = X9[,4], lty = 3, col = "red")
lines(y = pred11, x = X11[,4], lty = 4, col = "red")

legend(x = "top", inset = -0.2, xpd = NA, bty = "n", lty = 1:4, ncol = 2, col = "red",
       legend = c("hum = 5", "hum = 7", "hum = 9", "hum = 11"), cex = 0.7)
```



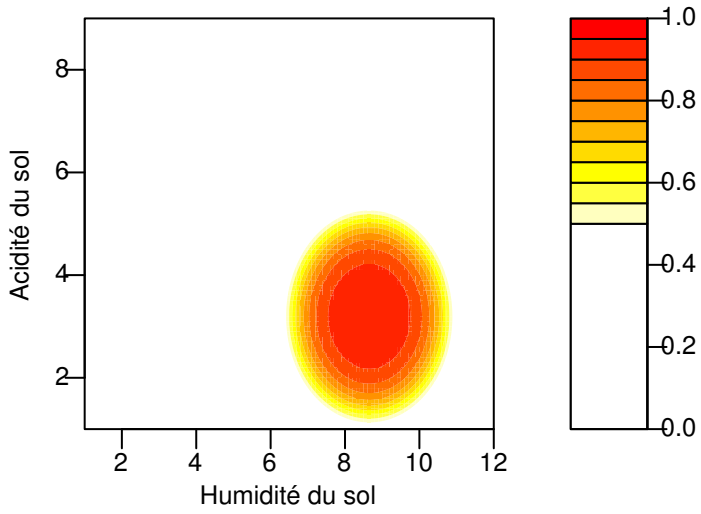

```r
# Prédiction de toutes les combinaisons de valeurs posibles pour hum et ph
hum <- seq(1,12, 0.1)
ph <- seq(1,9, 0.1)
X <- expand.grid(hum, ph)
colnames(X) <- c("hum", "ph")
X <- model.matrix(~ hum + I(hum^2) + ph + I(ph^2), data=X)
pred <- round(invlogit(X %*% coef(mod)),5)

# Pour pouvoir utiliser les représentations graphiques en 3D et pseudo 3D, il faut
# transformer les valeurs prédites en matrice
z <- matrix(pred, nrow = length(unique(X[,"hum"])))

# Image colorée
```

```r
par(mfrow = c(1,1), mar = c(2.5,2.5,2.5,1.1), mgp = c(1.2, 0.35,0), cex.axis = 0.8, cex.lab = 0.8)
image(x = seq(1,12, 0.1), y = seq(1,9, 0.1), z = z,
      xlab = "Humidité du sol", ylab = "Acidité du sol",
      col = c("white", rev(heat.colors(10))), breaks = c(0, seq(0.5, 1, 0.05)))
points(x = d[d$pres == 1,"hum"], y = d[d$pres == 1,"ph"], col = "grey50")
```




```
# Image colorée avec échelle
```

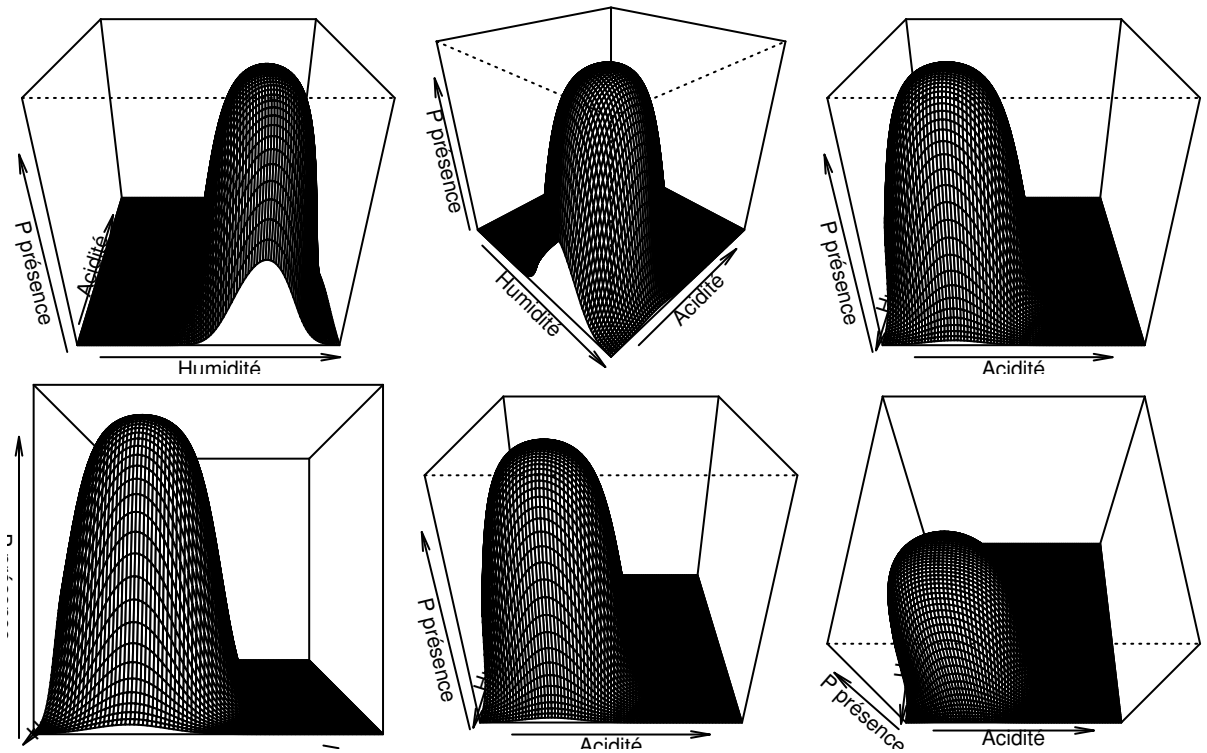
```
par(mfrow = c(1,1), mar = c(2.5,2.5,2.5,1.1), mgp = c(1.2, 0.35,0), cex.axis = 0.8, cex.lab = 0.8)
filled.contour(x = seq(1,12, 0.1), y = seq(1,9, 0.1), z = z,
               xlab = "Humidité du sol", ylab = "Acidité du sol",
               levels = c(0, seq(0.5, 1, 0.05)),col = c("white", rev(heat.colors(10))) )
```




```
# PSeudo-3D
```

```
par(mfrow = c(2,3), mar = c(0,0,0,0))
persp(x = seq(1,12, 0.1), y = seq(1,9, 0.1), z = z,
      xlab = "Humidité", ylab = "Acidité", zlab = "P présence",
      phi = 30, theta = 0)
persp(x = seq(1,12, 0.1), y = seq(1,9, 0.1), z = z,
      xlab = "Humidité", ylab = "Acidité", zlab = "P présence",
```

```
      phi = 30, theta = 45)
persp(x = seq(1,12, 0.1), y = seq(1,9, 0.1), z = z,
      xlab = "Humidité", ylab = "Acidité", zlab = "P présence",
      phi = 30, theta = 90)
persp(x = seq(1,12, 0.1), y = seq(1,9, 0.1), z = z,
      xlab = "Humidité", ylab = "Acidité", zlab = "P présence",
      phi = 0, theta = 90)
persp(x = seq(1,12, 0.1), y = seq(1,9, 0.1), z = z,
      xlab = "Humidité", ylab = "Acidité", zlab = "P présence",
      phi = 30, theta = 90)
persp(x = seq(1,12, 0.1), y = seq(1,9, 0.1), z = z,
      xlab = "Humidité", ylab = "Acidité", zlab = "P présence",
      phi = 60, theta = 90)
```




```
# Il est possible aussi d'obtenir des graphiques en 3D interactifs (non visible dans le pdf):
```

```
library(rgl)
persp3d(x = seq(1,12, 0.1), y = seq(1,9, 0.1), z = z, col = "gray", aspect = TRUE)


# prédictions des valeurs d'humidité et ph pour P > 0.8
pred <- data.frame( X, prob = pred)
apply(pred[pred$prob >= 0.80,c("hum", "ph")], 2, min)
```

```
## hum  ph
## 7.1 1.8
```

```
apply(pred[pred$prob >= 0.80,c("hum", "ph")], 2, max)
```

```
##  hum   ph
## 10.2  4.6
```

Exercice 10 : Vaches laitières - modèle mixte gaussien en carré latin

Note préliminaire
Les premiers exercices de cette série correspondent à des designs expérimentaux classiques avec à la fois des facteurs fixes et des facteurs aléatoires. Pour pouvoir comparer les différentes approches on a tenté de les analyser à la fois avec un modèle fixe, une anova mixte classique (fonction aov) et des modèles mixtes. Dans ces cas parfaitement ballancés, avec des erreurs à distribution normale, et des variables explicatives parfaitement indépendantes, les anova mixtes et les modèles mixtes devraient donner des résultats proches.

Ennoncé de l'exercice 10
On veut comparer l'effet de deux aliments B et C par rapport à un aliment témoin A sur la production laitière. On a fourni ces aliments à 6 vaches pendant 3 périodes. Chaque vache a reçu chaque aliment une seule fois pendant une des 3 périodes dans un orde différent. On s'est arrangé pourqu'à chaque période chaque aliment soit attribué à 2 vaches sur les six (design en carré latin). Chaque période représente 4 semaines d'observations avec une période d'adaptation préalable. C'est un exemple Tiré de Dagnelie 2003 Principes d'expérimentation Exemple 8.6. L'ouvrage est disponible en ligne : http://www.dagnelie.be/

```
d <- data.frame(
    aliment = c("B", "C", "A", "B", "A", "C", "A", "A", "B", "C", "C",
                "B", "C", "B", "C", "A", "B", "A"),
    periode = paste("per", rep(1:3, each = 6), sep="_"),
    vache = paste("vache", rep(1:6, times = 3), sep="_"),
    lait = c(21.7,20.4,25.9,23.5,19.1,19.0,19.1,19.9,24.0,20.5,17.0,19.1,16.4,
             19.5,22.2,21.9,20.0,17.6),
    aliment_t0 = c( rep(0,6), "B", "C", "A", "B", "A", "C", "A", "A",
                    "B", "C", "C", "B")
)
```

```
# On a a ici un facteur fixe à 3 niveaux (aliment) et deux facteurs aléatoires :
# vache (6 niveaux) et période (3 niveaux). Si on pense qu'il y a une tendance
# systématique due à la période, on pourrait considérer ce facteur comme fixe. Dans notre
# cas il s'agit clairement d'une répétition destinée à tester chaque aliment sur chaque
# vache dans un ordre différent.
# Tous ces facteurs sont croisés : chaque vache a reçu les 3 aliments et a été observée
# pendant les 3 périodes.
# On a cependant pratiquement aucune répétition des combinaisons de ces facteurs à part
# l'aliment qui a été testé chez 2 vaches à chaque période. On ne peut donc estimer
# aucune interaction à part aliment:periode.
# Avec une analyse de la variance classique on est cependant en général obligé de traiter
# les deux facteurs aléatoires comme si ils étaient fixes en supposant que toutes les
# interactions sont nulles (il existe des méthodes pour vérifier si cette hypothèse est
# tenable - voir Dagnelie 2003). En effet même dans le cas où on peut estimer les
# interactions il n'existe aucun carré moyen dans cette configuration contre lequel on
# peut tester l'effet fixe pour obtenir un F correct.

# On peut visualiser le design en carré latin comme suit :
library(reshape)
cast(periode ~ vache, value = "aliment", data=d)
```

```
##   periode vache_1 vache_2 vache_3 vache_4 vache_5 vache_6
## 1   per_1       B       C       A       B       A       C
## 2   per_2       A       A       B       C       C       B
## 3   per_3       C       B       C       A       B       A
```

```
# L'analyse en effets fixes.
# Il y a un effet très significatif des 3 variables explicatives
# On peut constater que les différents types de somme des carrés des écarts sont
```

```r
# identiques dans ce cas expérimental balancé et sans interactions.
mod <- lm(lait ~ aliment + periode + vache, data = d)
anova(mod) # type I
```

```
## Analysis of Variance Table
##
## Response: lait
##           Df Sum Sq Mean Sq F value    Pr(>F)
## aliment    2 12.988  6.4939  10.014 0.0066370 **
## periode    2 13.778  6.8889  10.623 0.0055984 **
## vache      5 71.698 14.3396  22.113 0.0001721 ***
## Residuals  8  5.188  0.6485
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
library(car)
Anova(mod) # type II
```

```
## Anova Table (Type II tests)
##
## Response: lait
##           Sum Sq Df F value    Pr(>F)
## aliment   12.988  2  10.014 0.0066370 **
## periode   13.778  2  10.623 0.0055984 **
## vache     71.698  5  22.113 0.0001721 ***
## Residuals  5.188  8
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
Anova(mod, type = 3) # type III
```

```
## Anova Table (Type III tests)
##
## Response: lait
##             Sum Sq Df  F value    Pr(>F)
## (Intercept) 756.04  1 1165.879 5.914e-10 ***
## aliment      12.99  2   10.014 0.0066370 **
## periode      13.78  2   10.623 0.0055984 **
## vache        71.70  5   22.113 0.0001721 ***
## Residuals     5.19  8
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Il est intéressant de comparer ces résultats avec une approche AIC.
# Les conclusions sont tout à fait différentes. Clairement les deux approches
# ne cherchent pas les mêmes choses. L'AIC cherche quelles sont les variables les plus
# importantes pour expliquer un phénomène tout en gardant un modèle suffisamment
# parcimonieux. Ici on a clairement pas suffisemment de données par rapport au nombre
# d'hypothèses que l'on veut tester du point de vue de l'AIC.
res <- model.select(mod)
res[[1]][,-c(2:5)]
```

```
##                    model   AICc AICc.delta AICc.w sum.w
## 5                  vache 86.612      0.000  0.492 0.492
## 1                        87.394      0.782  0.333 0.825
## 3                periode 91.103      4.491  0.052 0.877
## 2                aliment 91.261      4.649  0.048 0.925
```

```
## 7          periode+ vache 91.756      5.144  0.038 0.963
## 6          aliment+ vache 92.522      5.910  0.026 0.988
## 8 aliment+ periode+ vache 94.689      8.077  0.009 0.997
## 4        aliment+ periode 96.853     10.241  0.003 1.000
```

```
res[[2]]
```

```
##             freq     w
## (Intercept)  1.0 1.000
## vache        0.5 0.564
## periode      0.5 0.101
## aliment      0.5 0.085
```

```
# On peut utiliser aov et spécifer un terme d'erreur mais comme indiqué plus haut,
# on ne peut pas tester l'effet fixe si on considère periode et vache comme aléatoires.
# La fonction aov donne donc les "Sum of Squares" mais pas les F et les p valeurs.
aov(lait ~ aliment + Error(periode + vache), data = d)
```

```
##
## Call:
## aov(formula = lait ~ aliment + Error(periode + vache), data = d)
##
## Grand Mean: 20.37778
##
## Stratum 1: periode
##
## Terms:
##                 Residuals
## Sum of Squares   13.77778
## Deg. of Freedom         2
##
## Residual standard error: 2.624669
##
## Stratum 2: vache
##
## Terms:
##                 Residuals
## Sum of Squares   71.69778
## Deg. of Freedom         5
##
## Residual standard error: 3.786761
##
## Stratum 3: Within
##
## Terms:
##                   aliment Residuals
## Sum of Squares  12.987778  5.187778
## Deg. of Freedom         2         8
##
## Residual standard error: 0.8052777
## Estimated effects may be unbalanced
```

```
# Avec les modèles mixtes on peut spécifier des effets aléatoires croisés
lmm <- lmer(lait ~ aliment + (1|periode) + (1|vache), data=d)
summary(lmm)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: lait ~ aliment + (1 | periode) + (1 | vache)
##    Data: d
```

```
##
## REML criterion at convergence: 61.6533
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  vache    (Intercept) 4.5637   2.1363
##  periode  (Intercept) 1.0401   1.0198
##  Residual             0.6485   0.8053
## Number of obs: 18, groups: vache, 6; periode, 3
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)  20.5833     1.1024  18.671
## alimentB      0.7167     0.4649   1.541
## alimentC     -1.3333     0.4649  -2.868
##
## Correlation of Fixed Effects:
##          (Intr) almntB
## alimentB -0.211
## alimentC -0.211  0.500
```

```
# On peut constater que la majeure partie de la variation (après avoir enlevé la
# variation dûe aux aliments) est due à la différence entre vaches (74%) contre 16.5%
# pour la différence entre périodes. Voir les variances dans la partie "Random effects"
# du summary du modèle.
(varcomp <- c(unlist(VarCorr(lmm)), sigmasq = sigma(lmm)^2))
```

```
##     vache   periode   sigmasq
## 4.5636818 1.0400705 0.6484731
```

```
varcomp[1] / sum(varcomp)
```

```
##     vache
## 0.7299292
```

```
varcomp[2] / sum(varcomp)
```

```
##  periode
## 0.166352
```

```
varcomp[3] / sum(varcomp)
```

```
##   sigmasq
## 0.1037188
```

```
# On peut tester l'effet fixe avec un Test de Rapport de Vraisemblance
# Le nombre d'observations est faible ici et les conditions asymptotiques du test
# ne sont donc pas remplies. La p valeur est donc potentiellement trop basse.
# Ceci dit comme elle est très faible on peut raisonablement penser que l'effet est
# réellement significatif. On peut obtenir la p valeur par parametric bootsrap par aquis
# de conscience.
Anova.lmer(lmm)
```

```
##              LR df p(>Chisq)
## aliment 12.46175  2   0.00197
```

```r
system.time(res <- Anova.lmer(lmm, nsimul = 500))
```

```
##    user  system elapsed
##  21.789   0.384  22.223
```

```r
res
```

```
##              LR df p(>Chisq) simul p
## aliment 12.46175  2   0.00197    0.01
```

```r
# On peut aussi faire des tests de wald et des tests de F avec des degrés de liberté
# de Kenward-Roger puisqu'on est dans un modèle gaussien et un design ballancé.
# (packages car et afex)
Anova(lmm, test = "F")
```

```
## Analysis of Deviance Table (Type II Wald F tests with Kenward-Roger df)
##
## Response: lait
##              F Df Df.res   Pr(>F)
## aliment 10.014  2      8 0.006637 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
mixed(lait ~ aliment + (1|periode) + (1|vache), data=d)
```

```
## Fitting 3 (g)lmer() models:
## [...]
## Obtaining 2 p-values:
## [..]
```

```
##        Effect     stat ndf    ddf F.scaling p.value
## 1 (Intercept) 363.1968   1 6.5254         1  0.0000
## 2     aliment  10.0141   2 8.0000         1  0.0066
```

```r
# Le but était comparer les aliments B et C à l'aliment A.
# Les effets fixes de ce modèle effectuent déjà cette comparaison, l'intercept
# correspondant à l'alliment A.
# La valeur de t suggère que la différence A - C est estimée avec suffisemment
# de précision. On peut obtenir des intervalles de confiance par parametric bootstrap.
# Pour une raison que j'ignore la fonction confint ne fonctionne pas chez moi.
# Mais on peut utiliser directement bootMer
# confint(lmm, method = "boot", nsim = 200) # ne fonctionne pas

f <- function(m) {return(c(getME(m, "theta")*sigma(m), sigma = sigma(m), fixef(m)))}
b <- bootMer(lmm, FUN = f, nsim = 200)
(CI <- t(apply(b$t, 2, quantile, probs = c(0.025, 0.975), na.rm = TRUE)))
```

```
##                           2.5%      97.5%
## vache.(Intercept)    0.8028078  3.4677448
## periode.(Intercept)  0.0000000  1.8189694
## sigma                0.3996722  1.3094432
## (Intercept)         18.6630307 22.6502640
## alimentB            -0.1003013  1.5287327
## alimentC            -2.2873922 -0.4061776
```

```r
# On estime donc que les vaches nourries avec l'aliment C ont produit `r abs(round(CI[6,1],2))` à r abs(rou
# moins que les vaches nourries avec l'alliment A. Pour l'alliment B la production est
# "augmentée" de `r round(CI[5,1],2)` à `r round(CI[5,1],2)` l. L'intervalle de confiance contient donc le
# Ces estimations ne font aucun ajustement pour la multiplicité des "tests".
# Le package multcomp permet d'avoir une approche plus classique mais basée sur une
# approximation normale
C <- rbind("A - B" = c(0,-1,0), "A - C" = c(0,0,-1))
library(multcomp)
modmc <- glht(lmm, linfct=C)

# Sans correction et avec l'approximation normale utilisée dans ce package,
# la différence B-A n'est déjà pas significative
summary(modmc, test = adjusted("none"))
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Fit: lmer(formula = lait ~ aliment + (1 | periode) + (1 | vache),
##     data = d)
##
## Linear Hypotheses:
##            Estimate Std. Error z value Pr(>|z|)
## A - B == 0  -0.7167     0.4649  -1.541  0.12321
## A - C == 0   1.3333     0.4649   2.868  0.00413 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- none method)
```

```r
# Avec une correction classique, la différence A C est toujours significative et également
# avec la correction de Bonferroni qui est considérée comme une correction très
# conservative
summary(modmc)
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Fit: lmer(formula = lait ~ aliment + (1 | periode) + (1 | vache),
##     data = d)
##
## Linear Hypotheses:
##            Estimate Std. Error z value Pr(>|z|)
## A - B == 0  -0.7167     0.4649  -1.541  0.21346
## A - C == 0   1.3333     0.4649   2.868  0.00797 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

```r
summary(modmc, test = adjusted("bonferroni"))
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Fit: lmer(formula = lait ~ aliment + (1 | periode) + (1 | vache),
##     data = d)
##
## Linear Hypotheses:
##            Estimate Std. Error z value Pr(>|z|)
## A - B == 0  -0.7167     0.4649  -1.541  0.24641
```

```
## A - C == 0   1.3333     0.4649   2.868  0.00827 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- bonferroni method)
```

Exercice 11 : Modèle mixte binomial et sélection de modèle

On veut étudier les facteurs environnementaux qui peuvent expliquer (ou du moins prédire) la présence de la sole dans un estuaire portugais. (données du chapitre 21, de Zuur et al. 2007).
La présence de l'espèce et les variables environnementales ont été notés sur 65 points d'échantillonage répartis dans 4 zones (Area).
Construisez un modèle prédictif de la présence de l'espèce. Est-ce qu'un modèle mixte serait adapté dans ce cas et si oui lequel ? Faut-il utiliser toutes ces variables explicatives ? N'y a-t-il pas des variables redondantes/colinéaires ? Les relations sont-elles bien linéaires ? N'y a-t-il pas de points extrèmes ?
Une fois que vous avez un modèle complet satisfaisant, utilisez les méthodes de sélection de modèles pour déterminer quelles sont les variables les plus importantes pour prédire la présence de cette espèce.
Faites une représentation graphique du modèle sur base des "model averaged coefficients".

```
d <- structure(list(Sample = 1:65, season = c(1L, 1L, 1L, 1L, 1L,
1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L,
1L, 1L, 1L, 1L, 1L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L,
2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L,
2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L), month = c(5L,
5L, 5L, 5L, 5L, 5L, 5L, 5L, 5L, 5L, 5L, 5L, 6L, 6L, 6L, 6L,
6L, 6L, 6L, 6L, 6L, 6L, 6L, 6L, 6L, 7L, 7L, 7L, 7L, 7L, 7L, 7L,
7L, 7L, 7L, 7L, 7L, 7L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L,
8L, 8L, 8L, 9L, 9L, 9L, 9L, 9L, 9L, 9L, 9L, 9L, 9L, 9L, 9L, 9L
), Area = c(2L, 2L, 2L, 4L, 4L, 4L, 3L, 3L, 3L, 1L, 1L, 1L, 1L,
2L, 2L, 2L, 4L, 4L, 4L, 3L, 3L, 3L, 1L, 1L, 1L, 1L, 2L, 2L, 2L,
4L, 4L, 4L, 3L, 3L, 3L, 1L, 1L, 1L, 1L, 4L, 4L, 4L, 3L, 3L, 3L,
2L, 2L, 2L, 1L, 1L, 1L, 1L, 3L, 3L, 3L, 2L, 2L, 2L, 4L, 4L, 4L,
1L, 1L, 1L, 1L), depth = c(3, 2.6, 2.6, 2.1, 3.2, 3.5, 1.6, 1.7,
1.8, 4.5, 6, 4, 4, 2.7, 2.2, 2.5, 2.8, 4, 3.2, 2.3, 1.7, 2, 6.5,
3.5, 9, 2.5, 2.3, 2, 2.4, 2.4, 2, 1.6, 1.3, 1.25, 1.7, 4.5, 4.25,
4, 5, 2.25, 3.7, 2.2, 2, 1.7, 1.7, 2.7, 2.5, 2, 7, 5, 6, 6.5,
1.5, 1.5, 1.5, 3.2, 2.5, 3.5, 2.8, 2.1, 4, 5, 4.8, 1.4, 2.5),
temperature = c(20L, 18L, 19L, 20L, 20L, 20L, 19L, 17L, 19L,
21L, 22L, 22L, 22L, 21L, 21L, 21L, 21L, 21L, 22L, 20L, 20L,
21L, 22L, 24L, 23L, 23L, 23L, 23L, 23L, 24L, 24L, 25L, 27L,
26L, 26L, 25L, 25L, 25L, 25L, 24L, 24L, 24L, 24L, 24L, 24L,
22L, 22L, 23L, 25L, 25L, 25L, 25L, 23L, 23L, 23L, 19L, 19L,
18L, 17L, 19L, 19L, 21L, 21L, 22L, 21L), salinity = c(30L,
29L, 30L, 29L, 30L, 32L, 29L, 28L, 29L, 12L, 17L, 3L, 2L,
28L, 29L, 28L, 30L, 30L, 30L, 30L, 30L, 31L, 4L, 7L, 12L,
16L, 29L, 29L, 30L, 25L, 27L, 29L, 27L, 25L, 25L, 10L, 12L,
14L, 4L, 30L, 32L, 32L, 30L, 32L, 34L, 30L, 30L, 30L, 14L,
15L, 7L, 5L, 27L, 30L, 31L, 30L, 34L, 33L, 33L, 31L, 31L,
10L, 16L, 19L, 5L), transparency = c(15L, 15L, 15L, 15L,
15L, 7L, 15L, 10L, 10L, 35L, 30L, 35L, 35L, 10L, 10L, 5L,
10L, 5L, 40L, 20L, 15L, 10L, 40L, 20L, 35L, 40L, 25L, 10L,
15L, 25L, 20L, 30L, 20L, 20L, 20L, 60L, 80L, 50L, 60L, 25L,
25L, 40L, 20L, 15L, 15L, 30L, 30L, 30L, 70L, 80L, 50L, 40L,
20L, 15L, 15L, 20L, 20L, 5L, 15L, 7L, 10L, 30L, 50L, 30L,
25L), gravel = c(3.74, 1.94, 2.88, 11.06, 9.87, 32.45, 6.77,
22.61, 4.45, 3.54, 2.14, 3.1, 2.96, 3.64, 1.03, 1.33, 10.84,
8.43, 31.49, 5.08, 24.13, 3.37, 1.42, 1.77, 3.72, 0.62, 0.46,
1.63, 0.04, 7.88, 6.6, 31.29, 8.59, 28.59, 7.15, 1.96, 1.2,
6.85, 1.05, 2.62, 2.82, 1.96, 11.94, 7.66, 27.23, 9.2, 30.08,
4.9, 6.85, 1.41, 2.73, 3.2, 2.01, 0.21, 5.49, 7.33, 10.68,
26.22, 6.45, 30, 8.13, 10.47, 3.62, 1.51, 1.27), large_sand = c(13.15,
4.99, 8.98, 11.96, 28.6, 7.39, 14.55, 34.15, 15.43, 15.16,
36.89, 37.9, 40.24, 13.08, 3.14, 7.08, 11.22, 26.45, 6.46,
12.11, 37.24, 17.03, 12.53, 37.71, 39.08, 35.59, 16.22, 3.98,
```

```
3.01, 14.37, 23.7, 5.81, 9.33, 33.23, 13.12, 9.55, 40.71,
43.24, 32.19, 13.1, 2.33, 0.34, 16.88, 25.9, 10.34, 13.43,
35.5, 10.79, 5.99, 36.57, 41.21, 31.68, 8.92, 0.01, 4.03,
13.31, 24.88, 10.57, 9.2, 35.74, 7.88, 10.34, 35.92, 44.54,
27.4), med_fine_sand = c(11.93, 5.43, 16.85, 21.95, 19.49,
9.43, 9.88, 6.5, 7.88, 44.88, 44.33, 35.27, 36.81, 10.34,
4.42, 14.11, 20.7, 19.06, 9.38, 11.78, 6.44, 4.47, 41.57,
49.1, 38.03, 34.54, 7.08, 1.57, 13.51, 21.93, 21.41, 12.04,
14.51, 9.31, 0.91, 44.92, 51.56, 37.79, 31.19, 9.48, 0.55,
8.58, 22.42, 25.87, 11.2, 13.07, 8.53, 3.7, 40.65, 50.33,
36.82, 30.84, 7.87, 2.49, 5.5, 26.61, 26.46, 12.83, 16.02,
8.3, 1.29, 43.04, 45.72, 34.83, 29.44), mud = c(71.18, 87.63,
71.29, 55.03, 42.04, 50.72, 68.8, 36.75, 72.24, 36.42, 16.65,
23.73, 20, 72.94, 91.4, 77.48, 57.24, 46.06, 52.67, 71.03,
32.19, 75.13, 44.48, 11.43, 19.17, 29.25, 76.24, 92.82, 83.44,
55.82, 48.29, 50.86, 67.57, 28.87, 78.82, 43.57, 6.53, 12.12,
35.57, 74.8, 94.31, 89.11, 48.76, 40.57, 51.23, 64.31, 25.89,
80.6, 46.52, 11.69, 19.24, 34.28, 81.19, 97.29, 84.98, 52.76,
37.98, 50.38, 68.33, 25.96, 82.7, 36.14, 14.75, 19.12, 41.89
), Solea_solea = c(0L, 0L, 1L, 0L, 0L, 0L, 1L, 1L, 0L, 1L,
0L, 1L, 1L, 0L, 1L, 1L, 0L, 0L, 0L, 0L, 1L, 0L, 1L, 1L, 1L,
1L, 1L, 0L, 0L, 0L, 0L, 0L, 0L, 1L, 0L, 1L, 0L, 1L, 1L, 1L,
0L, 0L, 0L, 0L, 0L, 0L, 1L, 0L, 1L, 1L, 1L, 1L, 0L, 0L, 0L,
0L, 0L, 1L, 0L, 0L, 0L, 1L, 0L, 0L, 0L)), .Names = c("Sample",
"season", "month", "Area", "depth", "temperature", "salinity",
"transparency", "gravel", "large_sand", "med_fine_sand", "mud",
"Solea_solea"), class = "data.frame", row.names = c(NA, -65L))
```

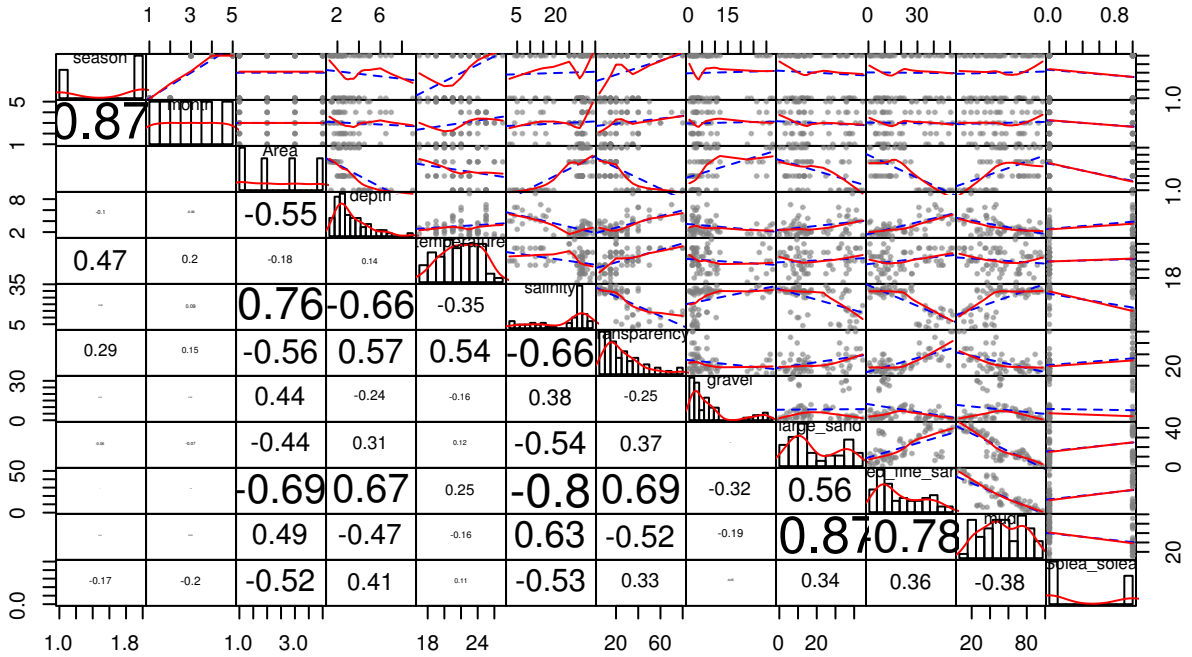
```
# On charge les scripts contenant diverses fonctions utiles
source("/home/gilles/stats/mytoolbox.R")
source("/home/gilles/stats/model.select_0.4.R")
par(mar = c(3, 3, 0.5, 0.5), mgp = c(1.75, 0.6, 0))

# le summary indique que certaines variables doivent être transformées en facteurs.
summary(d)
```

```
##      Sample       season        month          Area           depth        temperature       salinity
##  Min.   : 1   Min.   :1.0   Min.   :5   Min.   :1.000   Min.   :1.250   Min.   :17.00   Min.   : 2.00
##  1st Qu.:17   1st Qu.:1.0   1st Qu.:6   1st Qu.:1.000   1st Qu.:2.000   1st Qu.:20.00   1st Qu.:16.00
##  Median :33   Median :2.0   Median :7   Median :2.000   Median :2.500   Median :22.00   Median :29.00
##  Mean   :33   Mean   :1.6   Mean   :7   Mean   :2.385   Mean   :3.104   Mean   :22.09   Mean   :23.72
##  3rd Qu.:49   3rd Qu.:2.0   3rd Qu.:8   3rd Qu.:3.000   3rd Qu.:4.000   3rd Qu.:24.00   3rd Qu.:30.00
##  Max.   :65   Max.   :2.0   Max.   :9   Max.   :4.000   Max.   :9.000   Max.   :27.00   Max.   :34.00
##   transparency       gravel        large_sand    med_fine_sand        mud         Solea_solea
##  Min.   : 5.00   Min.   : 0.04   Min.   : 0.01   Min.   : 0.55   Min.   : 6.53   Min.   :0.0
##  1st Qu.:15.00   1st Qu.: 1.96   1st Qu.: 9.20   1st Qu.: 8.53   1st Qu.:34.28   1st Qu.:0.0
##  Median :20.00   Median : 4.45   Median :13.43   Median :14.51   Median :50.72   Median :0.0
##  Mean   :25.83   Mean   : 8.21   Mean   :19.28   Mean   :20.38   Mean   :52.13   Mean   :0.4
##  3rd Qu.:35.00   3rd Qu.: 9.20   3rd Qu.:33.23   3rd Qu.:34.54   3rd Qu.:72.94   3rd Qu.:1.0
##  Max.   :80.00   Max.   :32.45   Max.   :44.54   Max.   :51.56   Max.   :97.29   Max.   :1.0
```

```
d[, c("Sample", "season", "month", "Area")] <-
    lapply(d[, c("Sample", "season", "month", "Area")], as.factor)

# Un simple scatterplot matrix peut déjà nous apprendre beaucoup de choses ...
# Les 4 dernières variables concernant la granulométrie du fond sont fortement corrélées.
# Les choix possibles sont assez subjectifs. Ici on va garder mud et gravel et éliminer
```

```
# les deux autres. Mud est en effet fortement corrélé avec les deux variables supprimées
# on ne perd donc pas beaucoup d'information. Par ailleurs ces deux variables sont
# également fortement corrélées avec d'autres variables. En les suprimant on règle donc
# ces problèmes. Il faudra cependant garder en tête pour l'interprétation que si "mud"
# ressort comme variable explicative dans l'analyse ce sera peut-être dû à ces deux
# variables suprimées.
# On peut voir aussi que season et month sont redondants. On va garder dans une
# premier temps la variable month. C'est assez subjectif à ce stade
# et sans connaitre la biologie de l'espèce.
# On voit aussi qu'il y a des fortes différences entre sites (Area). Le site est typiquement
# un effet aléatoire ici. On veut contrôler le fait que les observations d'un même site ne
# sont vraisemblablement pas indépendantes mais on est pas particulièrement intéressé par
# une comparaison entre les sites. Quatre sites, ce n'est pas beaucoup pour estimer une
# variance. Il faudra donc  vérifier que le modèle arrive bien à estimer une variance.
# si non on a toujours la possibilité de l'utiliser comme variable fixe.
```

```
pairs2(d[,-1], pt.cex=0.6, gap=0)
```

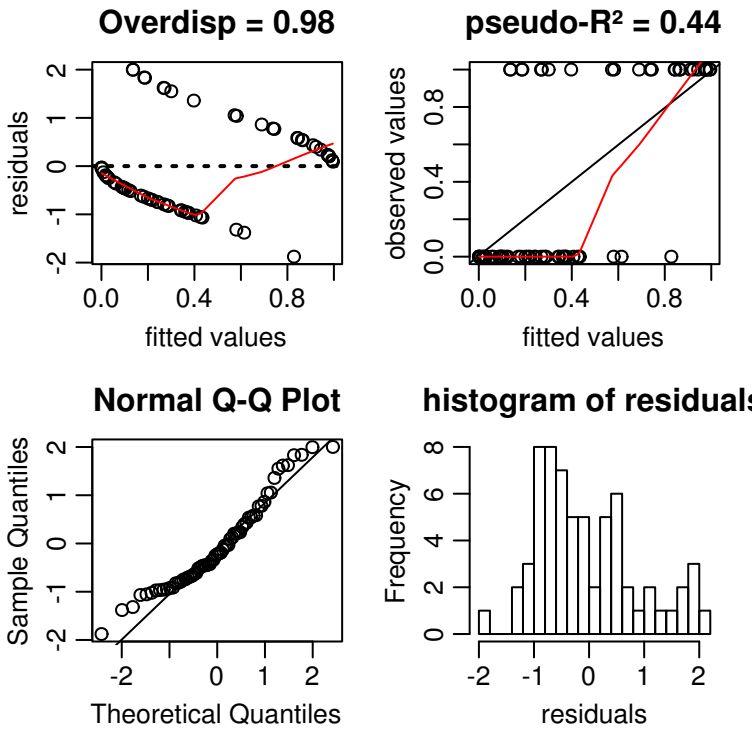



```
# Voici le modèle mixte avec Area comme variable aléatoire. On va rester avec un simple
# modèle random intercept. Le but principal est de contrôler pour la non indépendance
# des observations et on a assez peu de données et il faut essayer d'avoir un modèle
# pas trop complexe.
library(lme4)
mod <- glmer(Solea_solea ~ month + depth + temperature + salinity + transparency +
                gravel + mud + (1| Area), data=d, family = binomial)

# Si on regarde les sorties du summary, on constate qu'en effet, l'effet aléatoire Area
# est estimé à 0. On garde çà en tête pour le moment.
summary(mod)
```

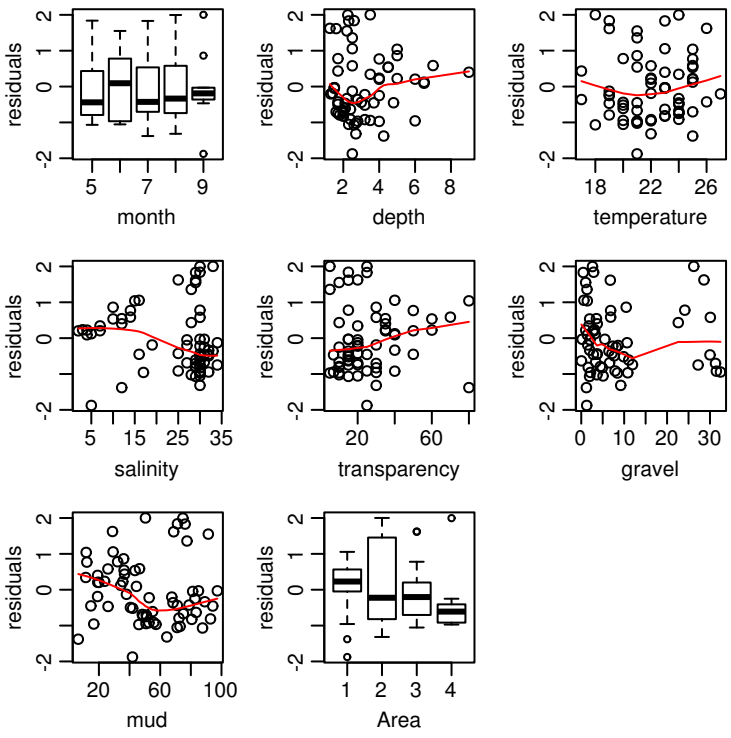
```
## Generalized linear mixed model fit by maximum likelihood ['glmerMod']
##  Family: binomial ( logit )
```

```
## Formula: Solea_solea ~ month + depth + temperature + salinity + transparency +      gravel + mud + (1 | Area)
##    Data: d
##
##      AIC      BIC   logLik deviance
##  77.3763 103.4689 -26.6881  53.3763
##
## Random effects:
##  Groups Name        Variance  Std.Dev.
##  Area   (Intercept) 1.981e-11 0.000004451
## Number of obs: 65, groups: Area, 4
##
## Fixed effects:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  24.616994  10.395585   2.368  0.01788 *
## month6        2.090692   1.239344   1.687  0.09162 .
## month7        3.780143   2.297627   1.645  0.09992 .
## month8        4.466173   2.236827   1.997  0.04586 *
## month9       -2.362834   1.463200  -1.615  0.10635
## depth         0.096921   0.363332   0.267  0.78966
## temperature  -0.904918   0.435530  -2.078  0.03773 *
## salinity     -0.310628   0.105398  -2.947  0.00321 **
## transparency -0.026449   0.034421  -0.768  0.44225
## gravel        0.075594   0.048954   1.544  0.12255
## mud           0.004724   0.024947   0.189  0.84980
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) month6 month7 month8 month9 depth  tmprtr salnty trnspr gravel
## month6       0.538
## month7       0.793  0.723
## month8       0.814  0.712  0.872
## month9      -0.304  0.153 -0.018 -0.056
## depth       -0.093  0.100  0.236  0.134  0.012
## temperature -0.957 -0.601 -0.858 -0.824  0.216 -0.071
## salinity    -0.621 -0.316 -0.384 -0.520  0.357  0.168  0.472
## transparncy -0.266 -0.137 -0.302 -0.402  0.152 -0.272  0.122  0.383
## gravel       0.182  0.145  0.133  0.161 -0.229  0.068 -0.198 -0.559  0.037
## mud         -0.181 -0.072 -0.155 -0.135 -0.036  0.079  0.121 -0.377  0.260  0.581
```

```
# Les plots de résidus sont à peu près OK. On voir qu'il y a des relations
# légèrement non linéaires pour certaines variables en particulier "mud" mais çà reste
# très raisonable et quelques essais de transformations ne changent pas grand chose.
# En cas de doute, on pourrait s'orienter vers des modèles additifs (GAM)
diagplot(mod)
```




```
diagplot2(mod)
```




```
# On peut aussi faire un essai avec un glm simple pour pouvoir visualiser un termplot
# et les vifs. Le termplot confirme que la non linéarité n'est pas dramatique.
# Les vifs montrent qu'il y a encore pas mal de colinéarité dans les variables
# explicatives. Il faut garder çà en tête pour la suite.
```

```
modglm <- glm(Solea_solea ~ month + depth + temperature + salinity + transparency +
                  gravel + mud , data=d, family = binomial)
par(mfrow=c(3,3))
par(mar = c(3, 3, 0.5, 0.5), mgp = c(1.75, 0.6, 0))
termplot(modglm, partial.resid=TRUE)
library(car)
vif(modglm)
```

```
##                   GVIF Df GVIF^(1/(2*Df))
## month        11.342256  4        1.354682
## depth         2.063838  1        1.436606
## temperature   8.047952  1        2.836891
## salinity      5.833663  1        2.415298
## transparency  3.655825  1        1.912021
## gravel        2.076346  1        1.440953
## mud           3.089789  1        1.757780
```

```
# On clacule les AICc, poids de modèles, de variables etc...
# Seule la variable salinité semble clairement supportée par les données avec un poids
# w+ de 0.954 (pour une fréquence initiale de 0.5). Les variables gravel et temperature
# sont faiblement supportées par les données (w+ = 0.54, 0.42), et les données suggèrent
# fortement que depth, mud et transparency complexifient inutilement les modèles.
# Pour la variable "month", on voit clairement que le mois de septembre est for différent
# des autres (signe négatif) mais l'estimation de ces paramètres est très peu précise
# (erreurs standard élevées par rapport aux coefficients). Cette seule variable ajoute
# cependant 4 paramètres au modèle ce qui peut être assez pénalisant quand on a
# relativement peu de données. On va donc essayer de remplacer month par season qui ne
# contient que 2 niveaux.
system.time(res <- model.select(mod))
```

```
##    user  system elapsed
##  39.630   0.024  39.738
```

```
res$AICtab[1:20, c("model", "AICc.delta", "AICc.w", "sum.w")]
```

```
##                                                 model AICc.delta AICc.w sum.w
## 41                                 salinity+ gravel      0.000  0.105 0.105
## 9                                         salinity      0.615  0.077 0.183
## 46               month+ temperature+ salinity+ gravel      1.115  0.060 0.243
## 14                       month+ temperature+ salinity      1.350  0.054 0.297
## 45                      temperature+ salinity+ gravel      1.944  0.040 0.337
## 43                           depth+ salinity+ gravel      1.945  0.040 0.377
## 105                            salinity+ gravel+ mud      2.170  0.036 0.412
## 57                    salinity+ transparency+ gravel      2.260  0.034 0.446
## 13                             temperature+ salinity      2.333  0.033 0.479
## 11                                   depth+ salinity      2.528  0.030 0.509
## 73                                     salinity+ mud      2.592  0.029 0.538
## 25                           salinity+ transparency      2.781  0.026 0.564
## 42                           month+ salinity+ gravel      2.949  0.024 0.588
## 62  month+ temperature+ salinity+ transparency+ gravel      3.250  0.021 0.609
## 78                  month+ temperature+ salinity+ mud      3.563  0.018 0.627
## 30          month+ temperature+ salinity+ transparency      3.604  0.017 0.644
## 110          month+ temperature+ salinity+ gravel+ mud      3.758  0.016 0.660
## 10                                   month+ salinity      3.817  0.016 0.676
## 48         month+ depth+ temperature+ salinity+ gravel      3.917  0.015 0.691
## 16                 month+ depth+ temperature+ salinity      4.051  0.014 0.704
```

```
res$mod.av[rev(order(res$mod.av$w)),-c(5,6)]
```

```
##              freq     w av.coef av.se
## (Intercept)   1.0 1.000   8.027 8.388
## salinity      0.5 0.954  -0.177 0.073
## gravel        0.5 0.544   0.037 0.027
## temperature   0.5 0.427  -0.226 0.220
## month9        0.5 0.340  -0.719 0.650
## month8        0.5 0.340   0.970 0.990
## month7        0.5 0.340   0.744 0.975
## month6        0.5 0.340   0.526 0.547
## depth         0.5 0.259   0.042 0.087
## mud           0.5 0.253   0.000 0.006
## transparency  0.5 0.244  -0.003 0.007
```

```
# Voici le modèle avec season à la place de month.
# on constate que avec ce modèle la composante de la variance associée au site (Area)
# n'est plus 0 ce qui est une bonne chose.
mod2 <- glmer(Solea_solea ~ season + depth + temperature + salinity + transparency +
                  gravel + mud + (1| Area), data=d, family = binomial)
summary(mod2)
```

```
## Generalized linear mixed model fit by maximum likelihood ['glmerMod']
##  Family: binomial ( logit )
## Formula: Solea_solea ~ season + depth + temperature + salinity + transparency +      gravel + mud + (1 | Area)
##    Data: d
##
##      AIC      BIC   logLik deviance
##  80.1585  99.7280 -31.0792  62.1585
##
## Random effects:
##  Groups Name        Variance Std.Dev.
##  Area   (Intercept) 0.7714   0.8783
## Number of obs: 65, groups: Area, 4
##
## Fixed effects:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   1.582396   4.924064   0.321   0.7479
## season2      -1.350566   0.843908  -1.600   0.1095
## depth         0.116148   0.334303   0.347   0.7283
## temperature   0.053424   0.195276   0.274   0.7844
## salinity     -0.185450   0.082441  -2.249   0.0245 *
## transparency  0.010553   0.031048   0.340   0.7339
## gravel        0.091040   0.046507   1.958   0.0503 .
## mud           0.009448   0.023483   0.402   0.6874
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) seasn2 depth  tmprtr salnty trnspr gravel
## season2      0.304
## depth       -0.421  0.143
## temperature -0.838 -0.435  0.155
## salinity    -0.539 -0.007  0.294  0.237
## transparncy  0.131 -0.200 -0.308 -0.376  0.031
## gravel       0.006 -0.146 -0.043 -0.082 -0.464  0.206
## mud         -0.002  0.018  0.020 -0.218 -0.411  0.338  0.608
```

```r
# Voici les résultats de la sélection de modèles.
# La variable température est maintenant plus du tout supportée par les données.
# Cette variable est redondante en partie avec la saison et la salinité qui semblent
# plus importantes ici. Ce set de modèles semble meilleur dans le sens où l'incertitude
# sur la sélection de modèle est moins grande : l'AICc weight chute plus vite et on a
# clairement deux groupes de variables, dont 4 variables clairement peu supportées
# par les données (w+ <0.271). Gravel et season on un poids plus élevé mais çà reste très
# limite. Les erreurs standrad non conditionnelles (av.se) de ces deux variables
# sont assez élevées en particulier pour la variable "season".
# NB Si on veut être strict, on ne devrait pouvoir prendre en colmpte dans l'incertitude
# ces différents essais avec season ou avec month.
# NB2 : l'utilisation de system.time n'est pas du tout obligatoire. Elle permet juste
# d'avoir une estimation du temps de calcul. A chaque fois qu'on ajoute une variable
# explicative au modèle global, le temps de calcul double à peu près.
system.time(res2 <- model.select(mod2))
```

```
##    user  system elapsed
##  24.922   0.000  24.967
```

```r
res2$AICtab[1:20, c("model", "k", "AICc.delta", "AICc.w", "sum.w")]
```

```
##                                      model k AICc.delta AICc.w sum.w
## 42                season+ salinity+ gravel 5      0.000  0.101 0.101
## 41                       salinity+ gravel 4      0.530  0.077 0.178
## 9                                 salinity 3      1.145  0.057 0.235
## 10                       season+ salinity 4      1.237  0.054 0.290
## 58  season+ salinity+ transparency+ gravel 6      2.221  0.033 0.323
## 46   season+ temperature+ salinity+ gravel 6      2.242  0.033 0.356
## 44         season+ depth+ salinity+ gravel 6      2.256  0.033 0.389
## 106          season+ salinity+ gravel+ mud 6      2.313  0.032 0.420
## 45           temperature+ salinity+ gravel 5      2.474  0.029 0.450
## 43                 depth+ salinity+ gravel 5      2.475  0.029 0.479
## 105                  salinity+ gravel+ mud 5      2.700  0.026 0.505
## 57          salinity+ transparency+ gravel 5      2.791  0.025 0.530
## 13                   temperature+ salinity 4      2.863  0.024 0.554
## 11                         depth+ salinity 4      3.059  0.022 0.576
## 73                           salinity+ mud 4      3.122  0.021 0.597
## 74                   season+ salinity+ mud 5      3.215  0.020 0.618
## 25                  salinity+ transparency 4      3.312  0.019 0.637
## 12                 season+ depth+ salinity 5      3.412  0.018 0.655
## 26          season+ salinity+ transparency 5      3.484  0.018 0.673
## 14           season+ temperature+ salinity 5      3.576  0.017 0.690
```

```r
res2$mod.av[rev(order(res2$mod.av$w)),-c(5,6)]
```

```
##              freq     w av.coef av.se
## (Intercept)   1.0 1.000   3.008 2.649
## salinity      0.5 0.932  -0.153 0.061
## gravel        0.5 0.589   0.043 0.029
## season2       0.5 0.515  -0.579 0.462
## depth         0.5 0.271   0.051 0.091
## mud           0.5 0.261   0.000 0.006
## temperature   0.5 0.257  -0.002 0.047
## transparency  0.5 0.245   0.001 0.007
```

```r
# Vu l'effet de la saison, on peut se demander si il n'y aurait pas une interaction
# entre celle-ci et les variables les plus importantes (salinity et gravel).
# Clairement non !
```

```
mod3 <- glmer(Solea_solea ~ season + depth + temperature + salinity + transparency +
                  gravel + mud + salinity:season + gravel:season + (1| Area), data=d, family = binomial)
system.time(res3 <- model.select(mod3))
```

```
##    user  system elapsed
##   57.10    0.00   57.21
```

```
res3$AICtab[1:20, c("model", "k", "AICc.delta", "AICc.w", "sum.w")]
```

```
##                                       model k AICc.delta AICc.w sum.w
## 42                    season+ salinity+ gravel 5      0.000  0.074 0.074
## 41                            salinity+ gravel 4      0.530  0.057 0.132
## 9                                    salinity 3      1.145  0.042 0.174
## 10                            season+ salinity 4      1.237  0.040 0.214
## 165   season+ salinity+ gravel+ season:gravel 6      1.536  0.035 0.248
## 137 season+ salinity+ gravel+ season:salinity 6      2.058  0.027 0.275
## 58     season+ salinity+ transparency+ gravel 6      2.221  0.025 0.299
## 46      season+ temperature+ salinity+ gravel 6      2.242  0.024 0.324
## 44             season+ depth+ salinity+ gravel 6      2.256  0.024 0.348
## 106              season+ salinity+ gravel+ mud 6      2.313  0.023 0.371
## 45               temperature+ salinity+ gravel 5      2.474  0.022 0.393
## 43                     depth+ salinity+ gravel 5      2.475  0.022 0.414
## 105                      salinity+ gravel+ mud 5      2.700  0.019 0.434
## 57             salinity+ transparency+ gravel 5      2.791  0.018 0.452
## 13                       temperature+ salinity 4      2.863  0.018 0.470
## 11                             depth+ salinity 4      3.059  0.016 0.486
## 73                               salinity+ mud 4      3.122  0.016 0.502
## 129         season+ salinity+ season:salinity 5      3.150  0.015 0.517
## 74                       season+ salinity+ mud 5      3.215  0.015 0.532
## 25                     salinity+ transparency 4      3.312  0.014 0.546
```

```
res3$mod.av[rev(order(res3$mod.av$w)),-c(5,6)]
```

```
##                   freq     w av.coef av.se
## (Intercept)      1.000 1.000   3.288 2.815
## salinity         0.615 0.944  -0.164 0.070
## gravel           0.615 0.651   0.043 0.033
## season2          0.692 0.642  -1.029 0.905
## depth            0.500 0.262   0.046 0.088
## mud              0.500 0.255   0.000 0.006
## temperature      0.500 0.248   0.001 0.046
## transparency     0.500 0.247   0.002 0.007
## season2:salinity 0.231 0.154   0.008 0.017
## season2:gravel   0.231 0.138   0.009 0.012
```

```
# Une pratique courrante consiste à prendre le meilleur modèle ou un modèle final
# avec les variables qui semblent les plus importantes et de voir si elles sont
# "significatives". Cependant rappelons que ces inférences ne tiennent pas compte de
# la sélection de modèle et sont conditionnelles à ce modèle
# (ie valides uniquement si ce modèle est vraiment le bon). Ceci dit, si on garde en tête
# ces limitations, cet examen peut néanmoins être instructif.
# On voit ici que l'effet "gravel" est limite significatif et la variable season l'est
# encore moins. Sachant que les tests de z et de rapport de vraissemblance donnent
# en général des p valeurs trop petites et que le fait de ne pas prendre en compte
# l'incertitude sur la sélection de modèle accentue encore ce fait, ces deux variables
# ne sont clairement pas suffisamment supportées par les données. Il est probable
# qu'elles aient un effet mais on a pas assez de données pour en être certain.
# Par contre l'effet de la salinité semble clair
```

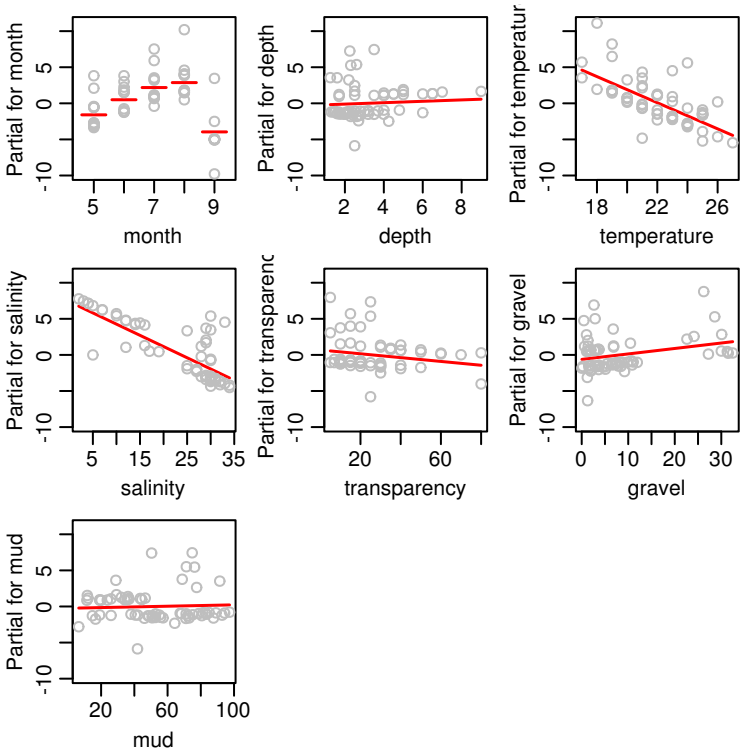
```
mod4 <- glmer(Solea_solea ~ season + salinity + gravel + (1| Area), data=d, family = binomial)
summary(mod4)
```

```
## Generalized linear mixed model fit by maximum likelihood ['glmerMod']
##  Family: binomial ( logit )
## Formula: Solea_solea ~ season + salinity + gravel + (1 | Area)
##    Data: d
##
##      AIC      BIC   logLik deviance
##  72.7551  83.6271 -31.3776  62.7551
##
## Random effects:
##  Groups Name        Variance Std.Dev.
##  Area   (Intercept) 0.6496   0.806
## Number of obs: 65, groups: Area, 4
##
## Fixed effects:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  4.11652    1.65506   2.487  0.01287 *
## season2     -1.13586    0.67440  -1.684  0.09213 .
## salinity    -0.19493    0.06264  -3.112  0.00186 **
## gravel       0.07761    0.03646   2.129  0.03327 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##          (Intr) seasn2 salnty
## season2  -0.364
## salinity -0.907  0.202
## gravel    0.134 -0.193 -0.343
```

```
Anova.lmer(mod4)
```

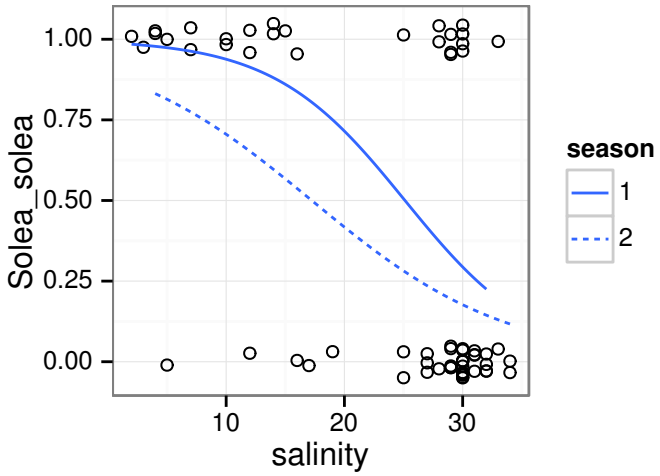
```
##                LR df p(>Chisq)
## season   2.880739  1    0.0896
## salinity 9.428085  1   0.00214
## gravel   3.587451  1    0.0582
```

```
# On pourrait calculer des intervales de confiance (conditionnels au modèle) comme ceci
# (mais c'est lent). Dans ce cas on peut franchement se contenter du coefficient + ou -
# 2 * l'erreur standard, en utilisant en particulier les model averaged coefficients et
# les erreurs standard non conditionnelles
# CI <- confint(mod4, method = "boot", nsim = 100, .progress="txt", PBargs=list(style=3))
```



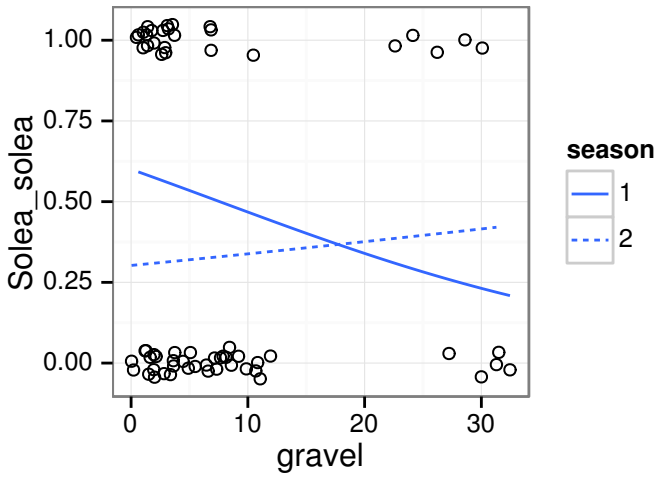

```
# On peut rappidement explorer les relations qui ressortent plus ou moins avec ggplot.
# Attention les lignes ne représentent pas notre modèle directement.
# Il s'agit de glm binomiaux univariés sur des sous ensemble des données.
library(ggplot2)
```

```
ggplot(d, aes(y = Solea_solea, x = salinity)) +
    geom_point(shape = 1, position = position_jitter(0, 0.05)) +
    stat_smooth(aes(group = season, linetype = season), method = "glm",
                se=FALSE, family = "binomial") +
    theme_bw()
```

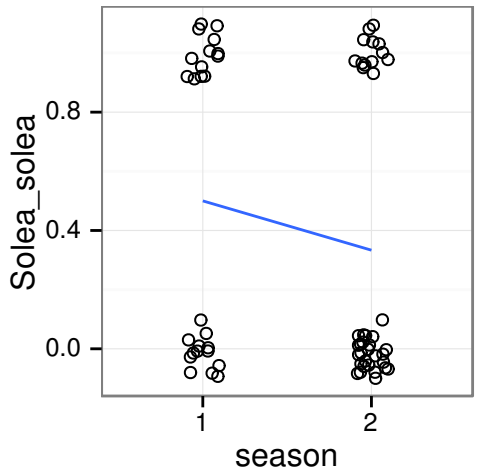



```
ggplot(d, aes(y = Solea_solea, x = gravel)) +
    geom_point(shape = 1, position = position_jitter(0, 0.05)) +
```

```
    stat_smooth(aes(group = season, linetype = season), method = "glm",
                se=FALSE, family = "binomial") +
    theme_bw()
```



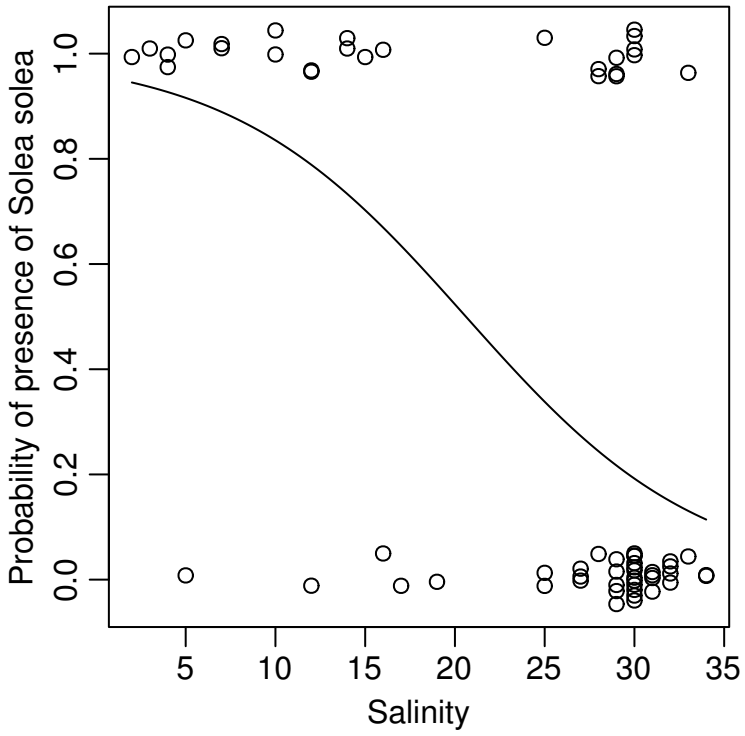

```
ggplot(d, aes(y = Solea_solea, x = season)) +
    geom_point(shape = 1, position = position_jitter(0.1, 0.1)) +
    stat_smooth(aes(group = 1), method = "glm",
                se=FALSE, family = "binomial") +
    theme_bw()
```




```
# Mais on peut aussi utiliser les "model averaged coefficients" pour faire une
# représentation graphique directe de notre modèle.
# On représente ici la relation avec la salinité après avoir contrôlé les autres
# variables explicatives (à leur valeur moyenne)
```

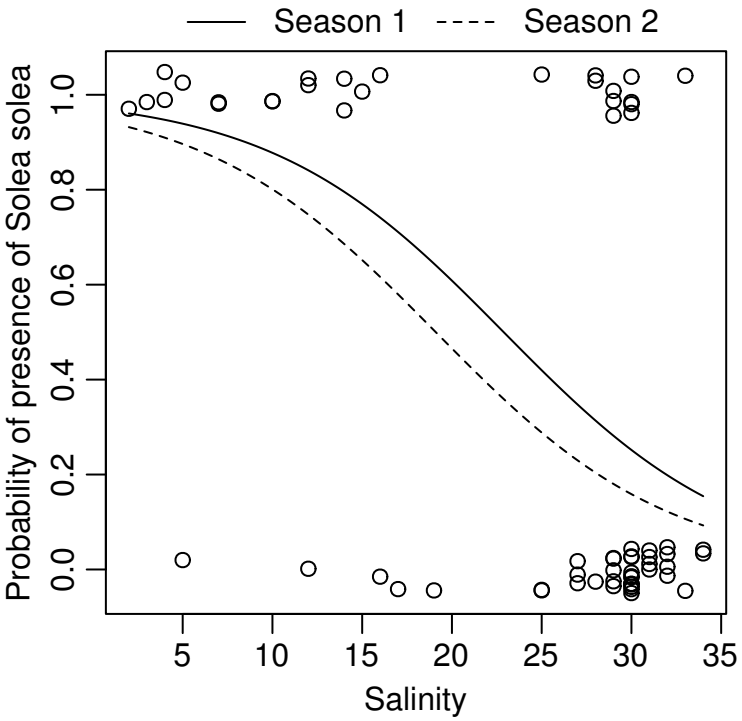
```
X <- model.matrix(~ season + depth + temperature + salinity + transparency +
                    gravel + mud , data=d)
newd <- as.data.frame(sapply(colMeans(X), rep, times = 100))
newd$salinity <- seq(min(X[,"salinity"]),max(X[,"salinity"]), length.out=100 )
```

```
pred <- invlogit(as.matrix(newd) %*% res2$mod.av$av.coef)

par(mar = c(3, 3, 0.5, 0.5), mgp = c(1.75, 0.6, 0))
plot(jitter(Solea_solea, amount = 0.05) ~ X[, "salinity"], data=d,
     xlab= "Salinity", ylab = "Probability of presence of Solea solea")
lines(pred ~ newd[, "salinity"])
```




```
newd <- as.data.frame(sapply(colMeans(X), rep, times = 100))
newd$salinity <- seq(min(X[,"salinity"]),max(X[,"salinity"]), length.out=100 )

Xs0 <- Xs1 <- newd
Xs0$season2 <- 0
Xs1$season2 <- 1
preds0 <- invlogit(as.matrix(Xs0) %*% res2$mod.av$av.coef)
preds1 <- invlogit(as.matrix(Xs1) %*% res2$mod.av$av.coef)

# On peut aussi représenter une courbe séparée pour chaque saison par exemple.
# On voit bien que l'effet de la saison est assez limité.
# On pourrait faire de même avec la variable gravel.
```

```
par(mar = c(3, 3, 2, 0.5), mgp = c(1.75, 0.6, 0))
plot(jitter(Solea_solea, amount = 0.05) ~ X[, "salinity"], data=d,
     xlab= "Salinity", ylab = "Probability of presence of Solea solea")
lines(preds0 ~ newd[, "salinity"])
lines(preds1 ~ newd[, "salinity"], lty=2)

legend("top", xpd = NA, bty = "n", inset = -0.12,
       lty = 1:2, legend = c("Season 1", "Season 2"), horiz = TRUE)
```


Season 1
Season 2
Probability of presence of Solea solea
0.0
0.2
0.4
0.6
0.8
1.0
5
10
15
20
25
30
35
Salinity