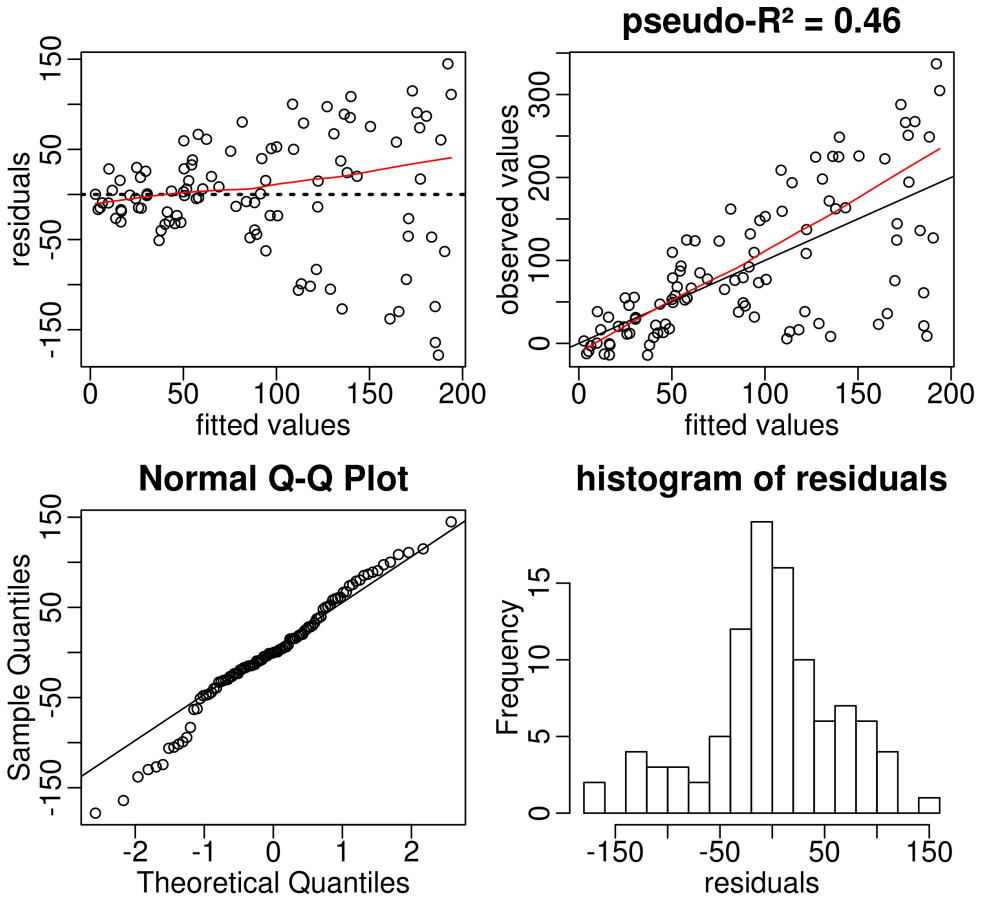
GLMs : En pratique



G. San Martin

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

cra-w

Conditions d'application des modèles

Les présupposés des GLMs

"model assumptions"

Par ordre d'importance (selon Gelman & Hill)

1) **adéquation**/validité
2) **linéarité** - additivité
3) **indépendance** des résidus
4) hypothèses sur la **variance** des résidus
5) hypothèses sur **distribution** des résidus

+ l'erreur de mesure des X doit être négligeable

Autres problèmes à garder en tête :

- **Outliers** : influence des données extrêmes (outliers)
- **Multicolinéarité** : indépendance des variables explicatives
- **Overfitting** : sélection de modèle nécessaire ?
- Faut-il **centrer** ou **standardiser** les données ?
- Quel **type de tests** (type I, II, III, ...), simulations, permutations,... ?

Conditions d'application des modèles

Les présupposés des GLMs

"model assumptions"
Par ordre d'importance (selon Gelman & Hill)

Les sources les plus fréquentes de problèmes à diagnostiquer et solutionner

1) **adéquation**/validité
2) **linéarité** - additivité
3) **indépendance** des résidus
4) hypothèses sur la **variance** des résidus
5) hypothèses sur **distribution** des résidus

+ l'erreur de mesure des X doit être négligeable

Autres problèmes à garder en tête :

- **Outliers** : influence des données extrêmes (outliers)
- **Multicolinéarité** : indépendance des variables explicatives
- **Overfitting** : sélection de modèle nécessaire ?
- Faut-il **centrer** ou **standardiser** les données ?
- Quel **type de tests** (type I, II, III, ...), simulations, permutations,... ?

Conditions d'application des modèles

Pour chaque type de présupposé/problème on va tenter de :

1) Expliquer quelle est la **nature du problème**
2) Montrer comment on peut **diagnostiquer**/détecter les problèmes
3) Proposer quelques pistes de **solutions**

Le diagnostique se fera le plus souvent sur base de graphiques des résidus (nuages de points, QQ-Plots, histogrammes) et par le calcul de statistiques descriptives (VIFs, coefficient de surdispersion,...)

Les graphiques de résidus sont particulièrement utiles.
Ils permettent de détecter de nombreux problèmes différents mais il est parfois difficile de connaître la cause exacte du problème

Conditions d'application des modèles

Il est impossible de donner une procédure linéaire pour examiner les problèmes ni de recettes automatiques pour les solutionner...

L'ordre présenté ci-après est donc assez arbitraire et n'est certainement pas à appliquer tel quel.

Certains diagnostiques se font avant d'estimer le premier modèle lors de la phase d'exploration des données
(SPLOMs, histogrammes, matrices de corrélation,...)

D'autres se font après avoir estimé un premier modèle
(VIFs, surdispersion, résidus)
Il faut régler tous ces problèmes avant de regarder les inférences !
Il s'agit d'un processus itératif qui permet d'améliorer progressivement le modèle en regardant les données.

Cette approche est pratiquement inévitable mais le danger est d'adapter complètement le modèle au jeu de données et que le résultat ne soit pas extrapolable à d'autres données (overfitting / data dredging). Idéalement il faudrait un jeu de données indépendant pour tester/valider le modèle (pex via Cross-Validation)

Valeurs extrêmes

Conditions d'application des modèles

Les présupposés des GLMs

("model assumptions")

par ordre d'importance (selon Gelman & Hill)

1) **adéquation**/validité
2) **linéarité** - additivité
3) **indépendance** des résidus
4) hypothèses sur la **variance** des résidus
5) hypothèses sur **distribution** des résidus

+ l'erreur de mesure des X doit être négligeable

Autres problèmes à garder en tête :

- **Outliers** : influence des données extrêmes
- **Multicolinéarité** : indépendance des variables explicatives
- **Overfitting** : sélection de modèle nécessaire ?
- Faut-il **centrer** ou **standardiser** les données ?
- Quel **type de tests** (type I, II, III, ...), simulations, permutations,... ?

-

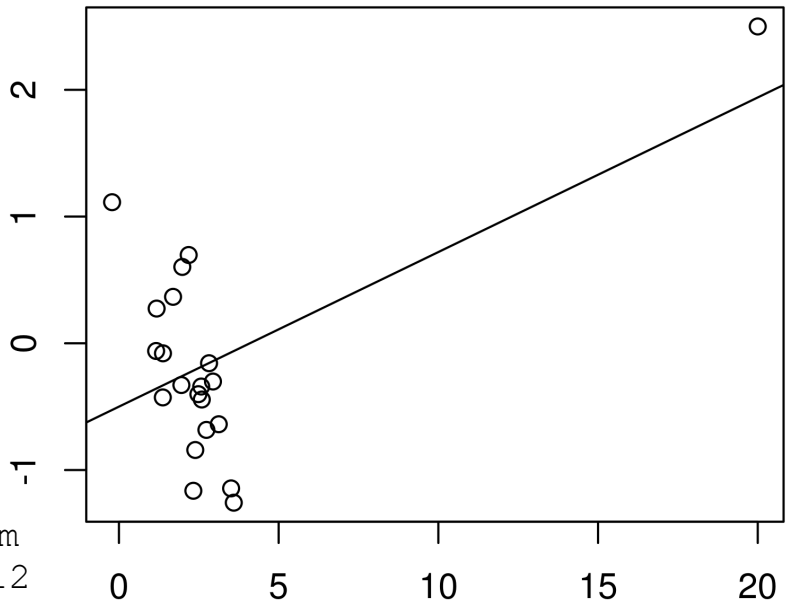
Valeurs extrêmes

Le problème

Les résultats du GLM peuvent être très fortement influencées par une ou plusieurs valeurs beaucoup plus grandes que les autres dans les variables explicatives.

Dans l'exemple suivant, on a une relation significativement positive...

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

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

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

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

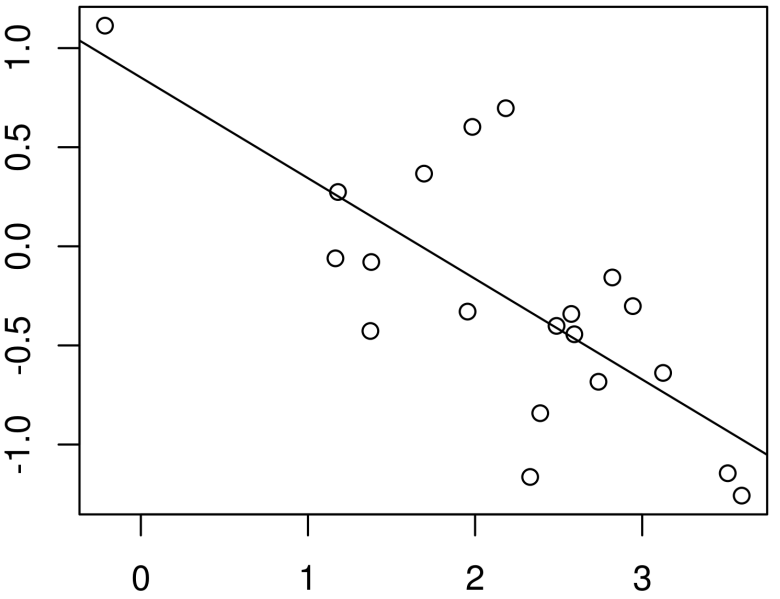


Valeurs extrêmes

Le problème

Après élimination de la valeur extrême on a une relation significativement négative ...
Cette relation est valide pour la gamme de valeurs de x entre 0 et 4 mais pas au delà...

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

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

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



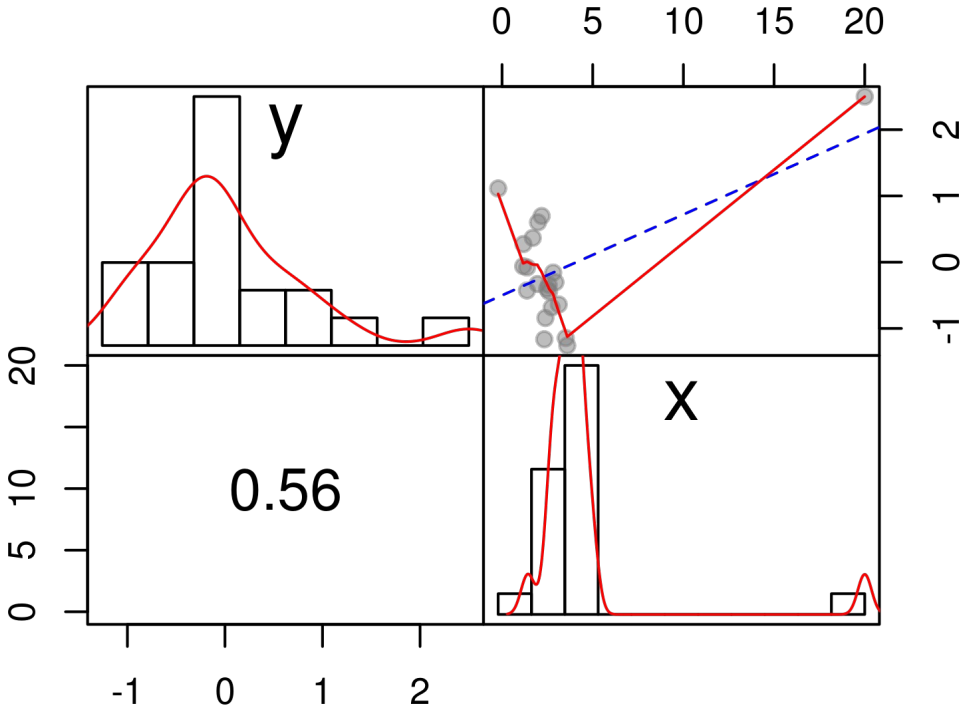
Valeurs extrêmes

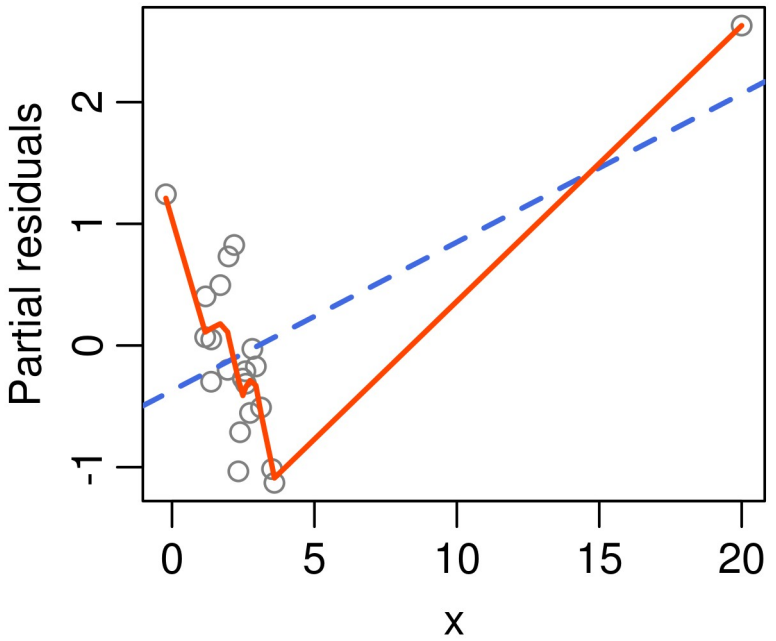
Diagnostique

Toujours faire une représentation graphique des données !!

SPLOM de y et des x (`pairs2`)
Histogrammes des x (diagonale de `pairs2`)
Graphiques des résidus vs variables explicatives (`diagplot2`)

```
pairs2(d)
diagplot2(m)
```





Valeurs extrêmes

Diagnostique

Il existe des statistiques descriptives des valeurs extrêmes
Les principales statistiques sont :

hat values : mesurent le leverage c.à d. à quel point une valeur de x est loin des autres (idéalement h < 2 * k/n ou h < 3*k/n pour petits n)

dfbetas : mesurent la différence de la valeur de chaque coefficient entre un modèle avec toutes les données et un modèle où on a enlevé une observations (les x sont standardisés)
idéalement dfbetas < |1|

Cook's distance : mesure plus synthétique de la précédente de l'effet de la suppression d'une observation sur l'ensemble des coefficients.
idéalement Cook.dist D < 4/(n-k-1) ou pf(D, k, n - k) < 0.5

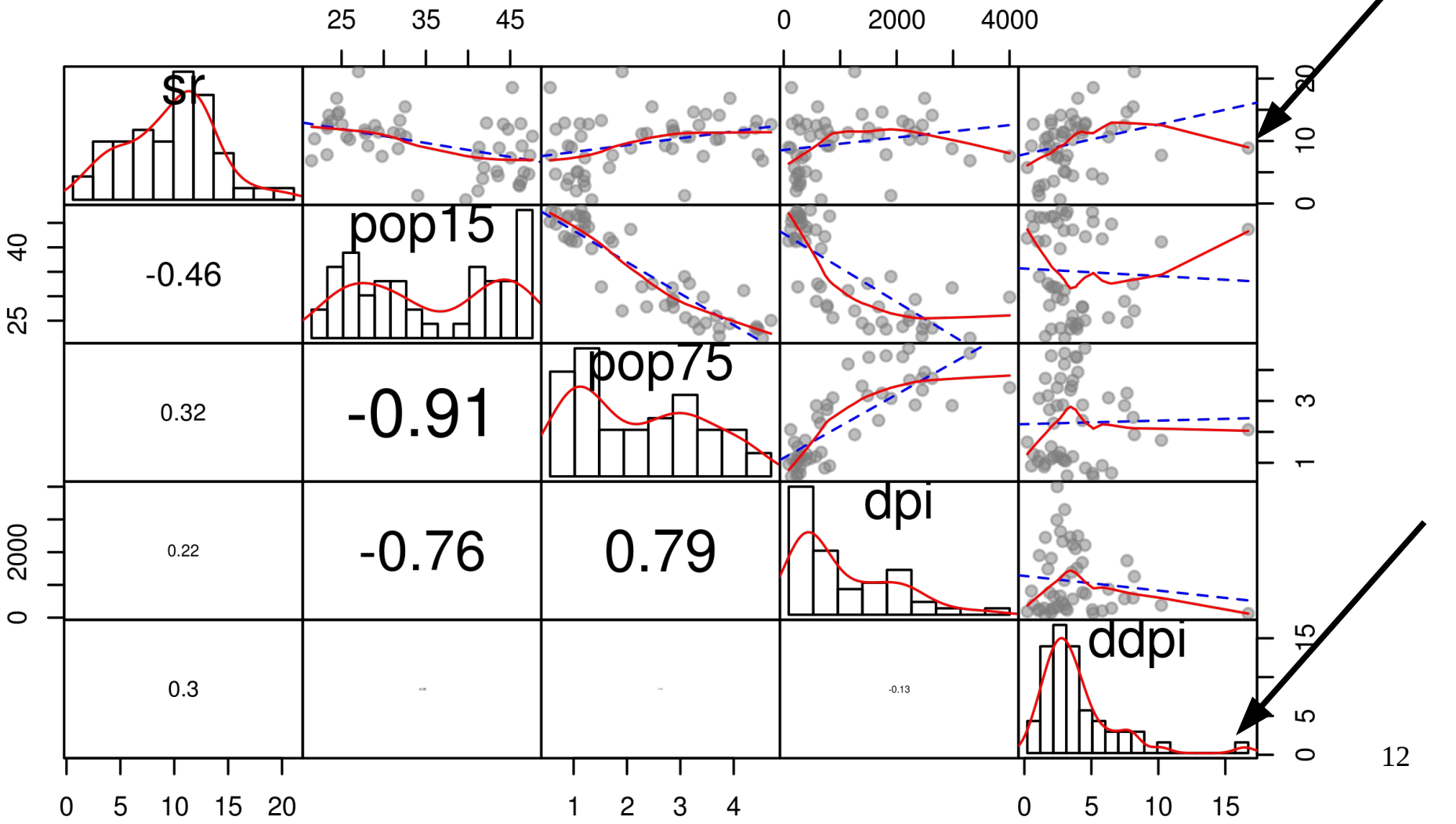
n = nombre d'observations - k = nombre de coefficients du modèle

Si il y a deux valeurs extrêmes au même endroit, dfbetas et Cook ne peuvent les détecter
Dans R : fonctions `hatvalues(m)`, `dfbetas(m)`, `cooks.distance(m)`
`summary(influence.measures(m))`
Dans mytoolbox.R, graphiques : `influence.plot(m)`, `dfbetas.plot(m)`

Valeurs extrêmes

Diagnostique

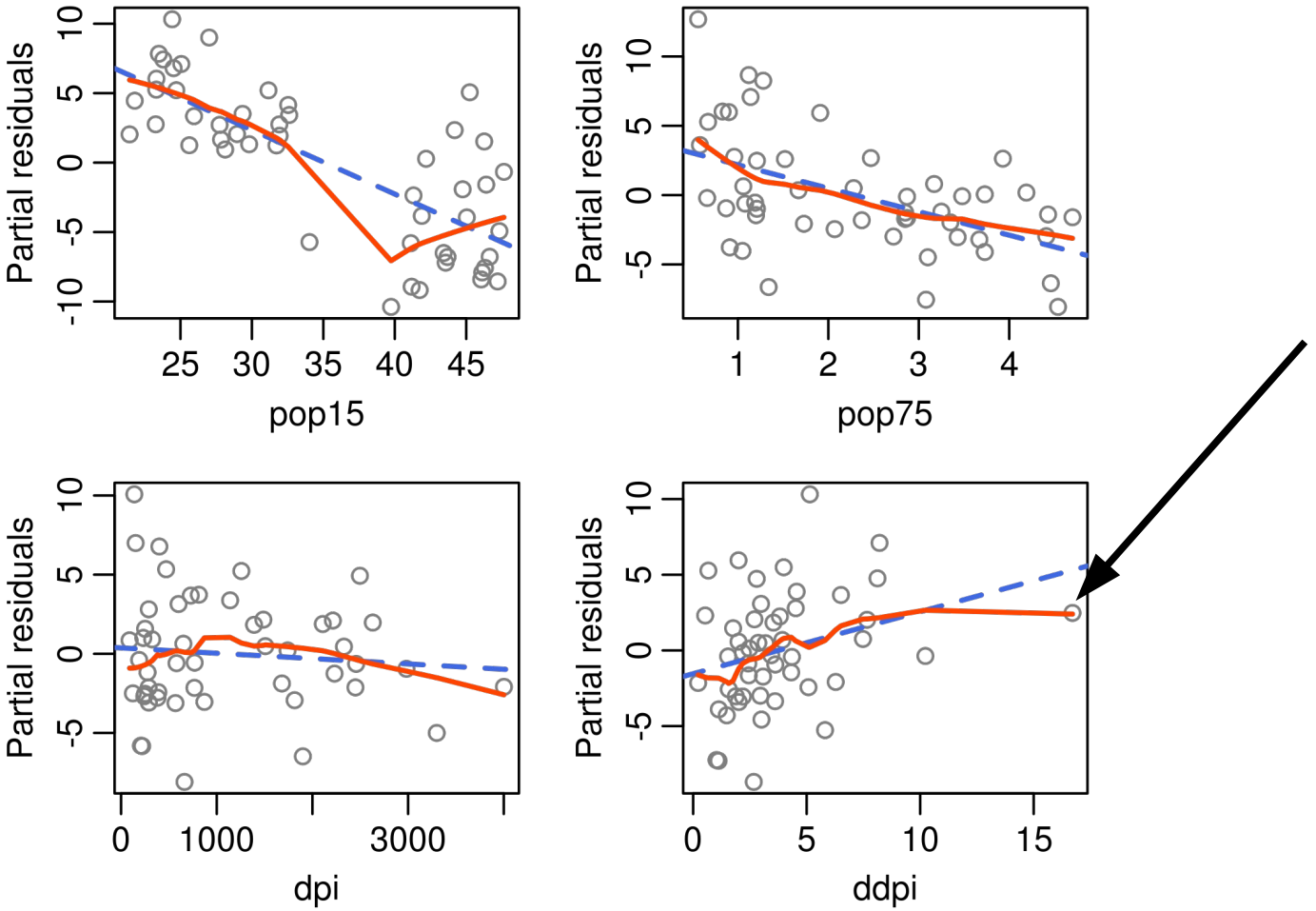
```
m <- lm(sr ~ pop15 + pop75 + dpi + ddpi, data = LifeCycleSavings)
pairs2(LifeCycleSavings)
diagplot2(m)
```



Valeurs extrêmes

Diagnostique

```
m <- lm(sr ~ pop15 + pop75 + dpi + ddpi, data = LifeCycleSavings)
pairs2(LifeCycleSavings)
diagplot2(m)
```



Valeurs extrêmes

Diagnostique

```
> infl <- influence.measures(m)

> summary(infl)
Potentially influential observations of
     lm(formula = sr ~ pop15 + pop75 + dpi + ddpi, data = LifeCycleSavings) :

               dfb.1_ dfb.pp15 dfb.pp75 dfb.dpi dfb.ddpi dffit   cov.r   cook.d hat
Chile          -0.20   0.13     0.22    -0.02    0.12    -0.46    0.65_*  0.04   0.04
United States   0.07  -0.07     0.04    -0.23   -0.03    -0.25    1.66_*  0.01   0.33_*
Zambia          0.16  -0.08    -0.34     0.09    0.23     0.75    0.51_*  0.10   0.06
Libya           0.55  -0.48    -0.38    -0.02   -1.02_*  -1.16_*  2.09_*  0.27   0.53_*

> infl
Influence measures of
     lm(formula = sr ~ pop15 + pop75 + dpi + ddpi, data = LifeCycleSavings) :

                dfb.1_ dfb.pp15 dfb.pp75  dfb.dpi  dfb.ddpi   dffit cov.r    cook.d    hat inf
Australia       0.01232 -0.01044 -0.02653  0.04534 -0.000159  0.0627 1.193 0.0008036 0.0677
(...)
Chile          -0.19941  0.13265  0.21979 -0.01998  0.120007 -0.4554 0.655 0.0378132 0.0373   *
China           0.02112 -0.00573 -0.08311  0.05180  0.110627  0.2008 1.150 0.0081570 0.0780
(...)
United Kingdom  0.04671 -0.03584 -0.17129  0.12554  0.100314 -0.2722 1.189 0.0149663 0.1165
United States   0.06910 -0.07289  0.03745 -0.23312 -0.032729 -0.2510 1.655 0.0128448 0.3337   *
Venezuela      -0.05083  0.10080 -0.03366  0.11366 -0.124486  0.3071 1.095 0.0188614 0.0863
Zambia          0.16361 -0.07917 -0.33899  0.09406  0.228232  0.7482 0.512 0.0966328 0.0643   *
Jamaica         0.10958 -0.10022 -0.05722 -0.00703 -0.295461 -0.3456 1.200 0.0240268 0.1408
Uruguay        -0.13403  0.12880  0.02953  0.13132  0.099591 -0.2051 1.187 0.0085323 0.0979
Libya           0.55074 -0.48324 -0.37974 -0.01937 -1.024477 -1.1601 2.091 0.2680704 0.5315   *
Malaysia        0.03684 -0.06113  0.03235 -0.04956 -0.072294 -0.2126 1.113 0.0091134 0.0652
(...)
```

Valeurs extrêmes

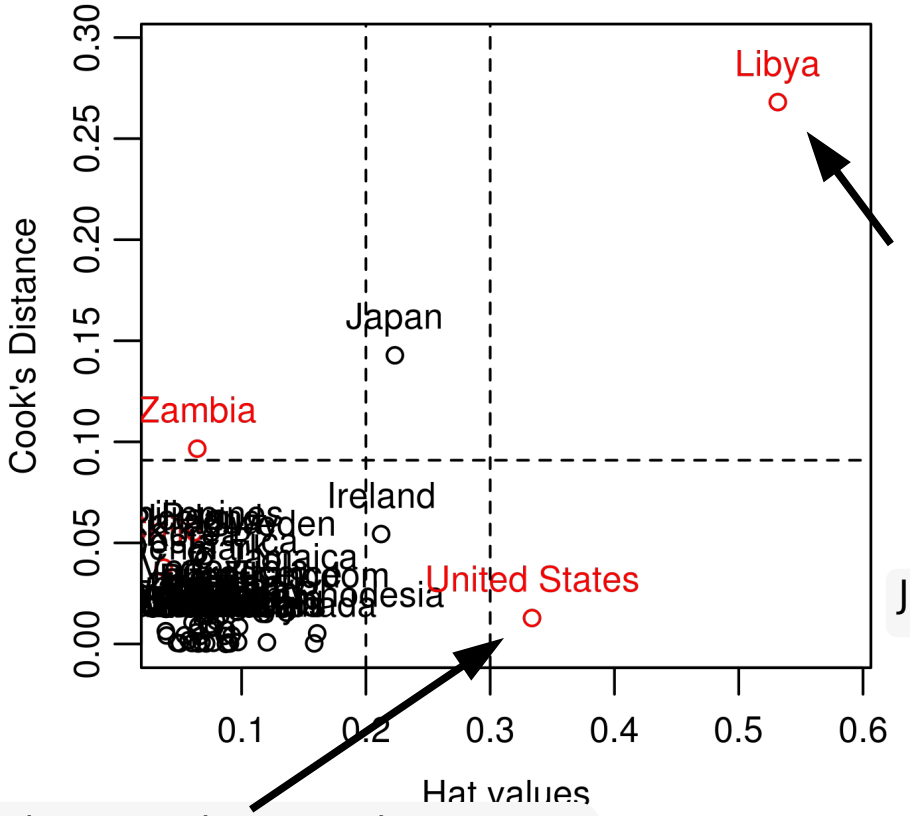
Diagnostique

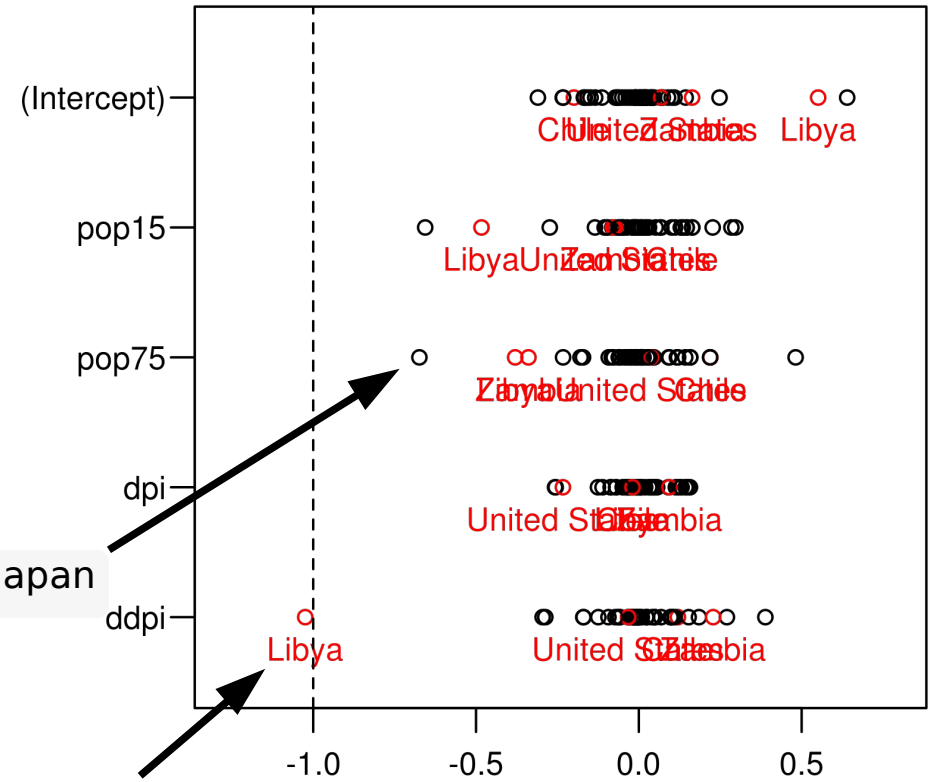
```
influence.plot(m)
dfbetas.plot(m)
```

```
> df <- dfbetas(m)
> row.names(df)[which.min(df[, "pop75"])]
[1] "Japan"
```

En rouge :
Les valeurs potentiellement influentes selon `summary(infl)`

NB : il y a une deuxième ligne pointillée horizontale hors du graphique
Quantile de la Loi de Fischer.





United States : leverage important mais malgré tout, effet limité sur les coefficients cfr Cook's D. faible

Seule la Lybie a un effet important sur le coefficient ddpi

Valeurs extrêmes

Solutions ?

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

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

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

Transformer les x pour réduire l'influence des grandes valeurs (log, sqrt). Attention aux conséquences sur la linéarité !

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

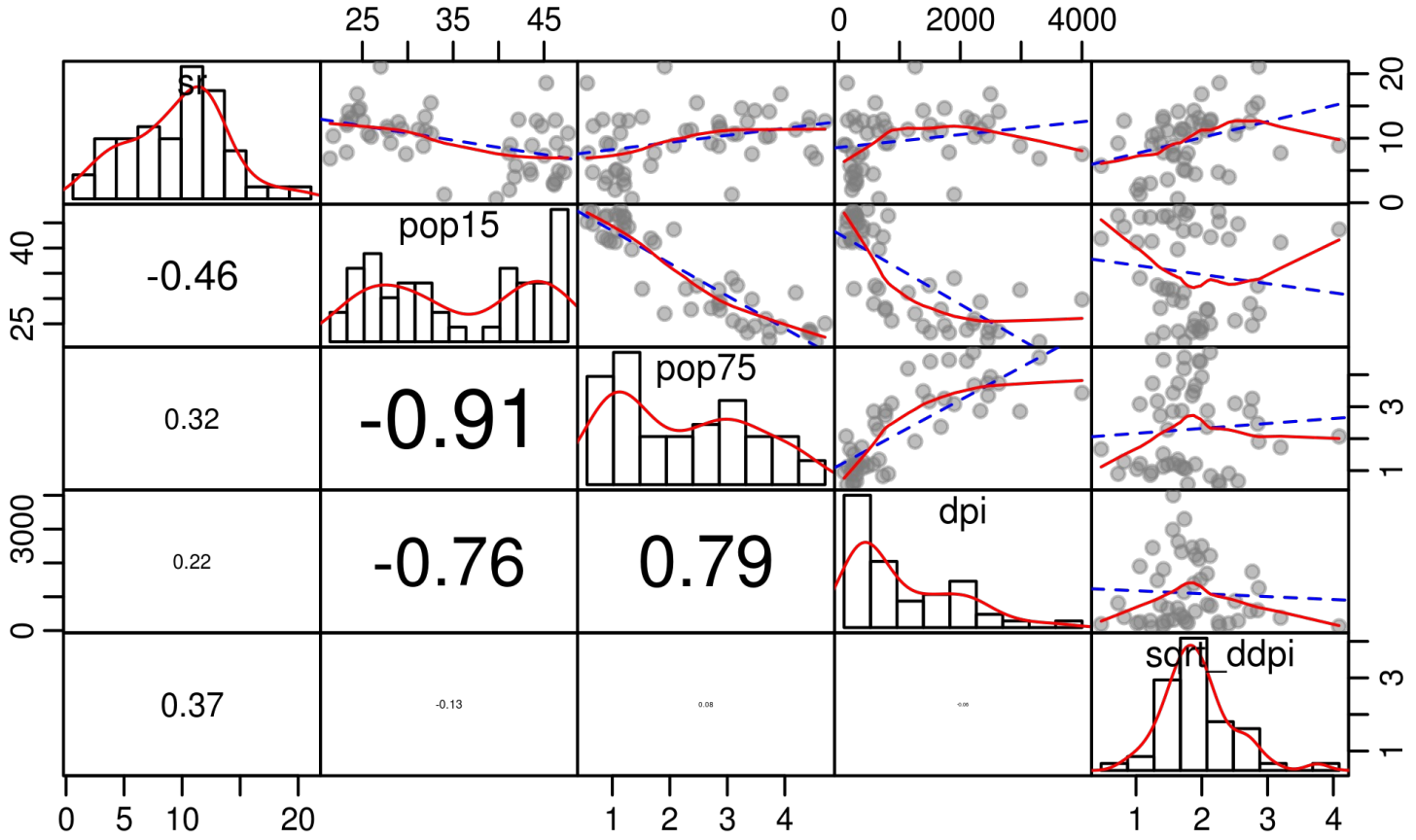
Utiliser des méthodes robustes aux valeurs extrêmes (robust regression pex)

Valeurs extrêmes

Solutions ?

```
d <- LifeCycleSavings
d$sqrt_ddpi <- sqrt(d$ddpi)
d <- d[,-5]

m <- lm(sr ~ pop15 + pop75 + dpi + sqrt_ddpi, data = d)
pairs2(d)
diagplot2(m)
```

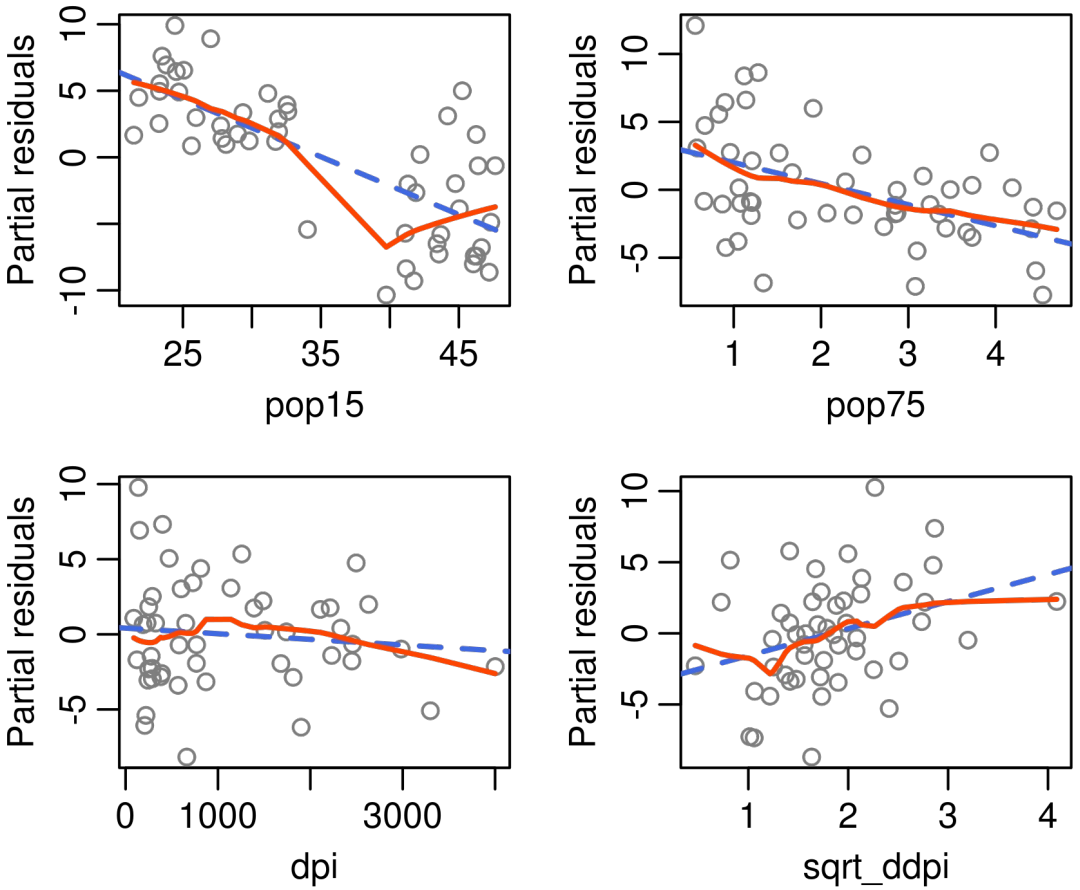
Transformation racine carrée



Valeurs extrêmes

Solutions ?

```
d <- LifeCycleSavings
d$sqrt_ddpi <- sqrt(d$ddpi)
d <- d[,-5]

m <- lm(sr ~ pop15 + pop75 + dpi + sqrt_ddpi, data = d)
pairs2(d)
diagplot2(m)
```



Valeurs extrêmes

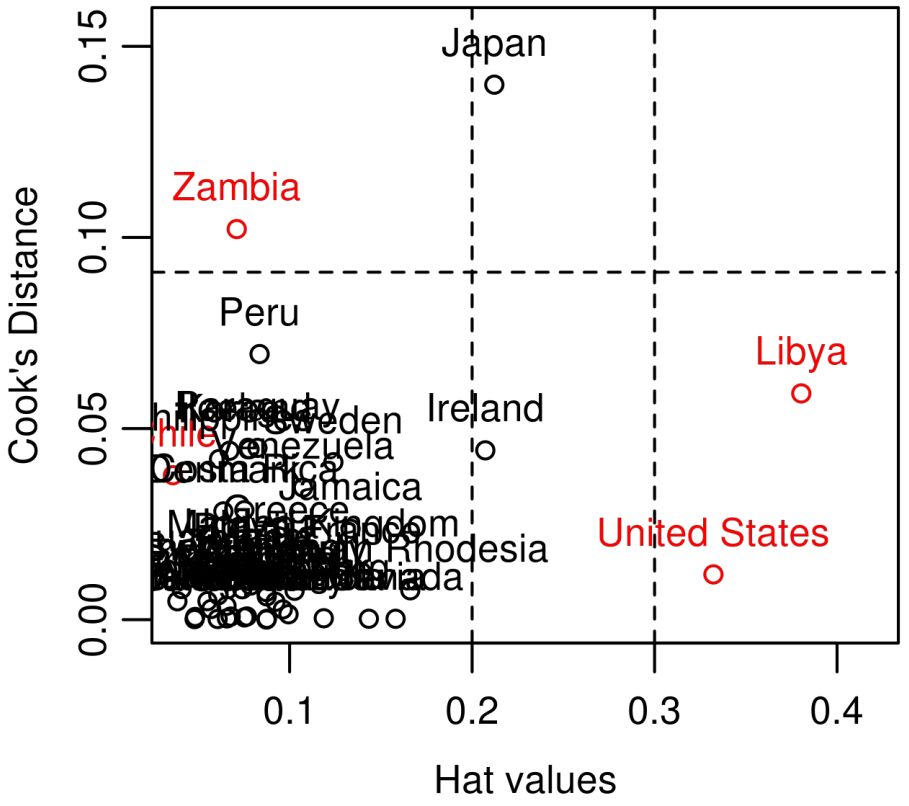
Solutions ?

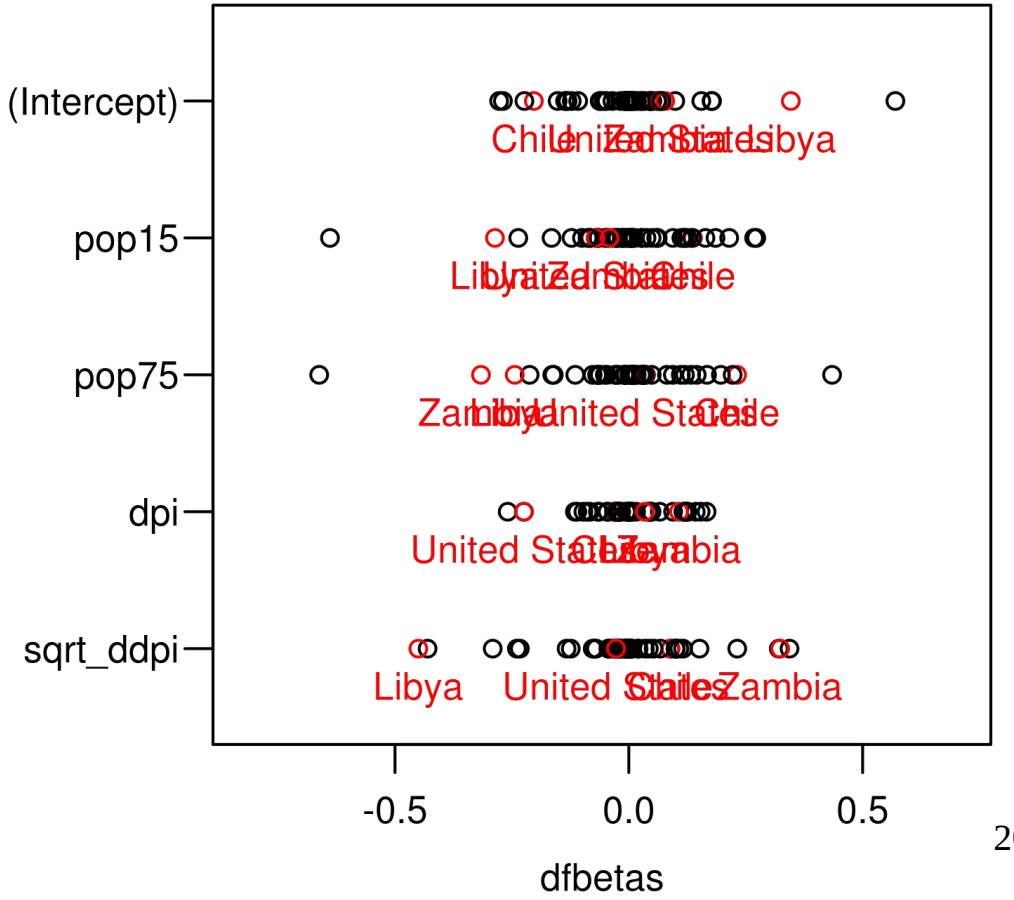
```
> infl <- influence.measures(m)
> summary(infl)
Potentially influential observations of
     lm(formula = sr ~ pop15 + pop75 + dpi + sqrt_ddpi, data = d) :

              dfb.1_ dfb.pp15 dfb.pp75 dfb.dpi dfb.sqr_ dffit cov.r   cook.d hat
Chile         -0.20   0.14     0.23    -0.03    0.09    -0.46  0.64_*  0.04   0.04
United States  0.07  -0.07     0.03    -0.22   -0.03    -0.24  1.65_*  0.01   0.33_*
Zambia         0.08  -0.04    -0.32     0.11    0.32     0.77  0.54_*  0.10   0.07
Libya          0.35  -0.29    -0.24     0.03   -0.45    -0.54  1.71_*  0.06   0.38_*
```

Valeurs extrêmes

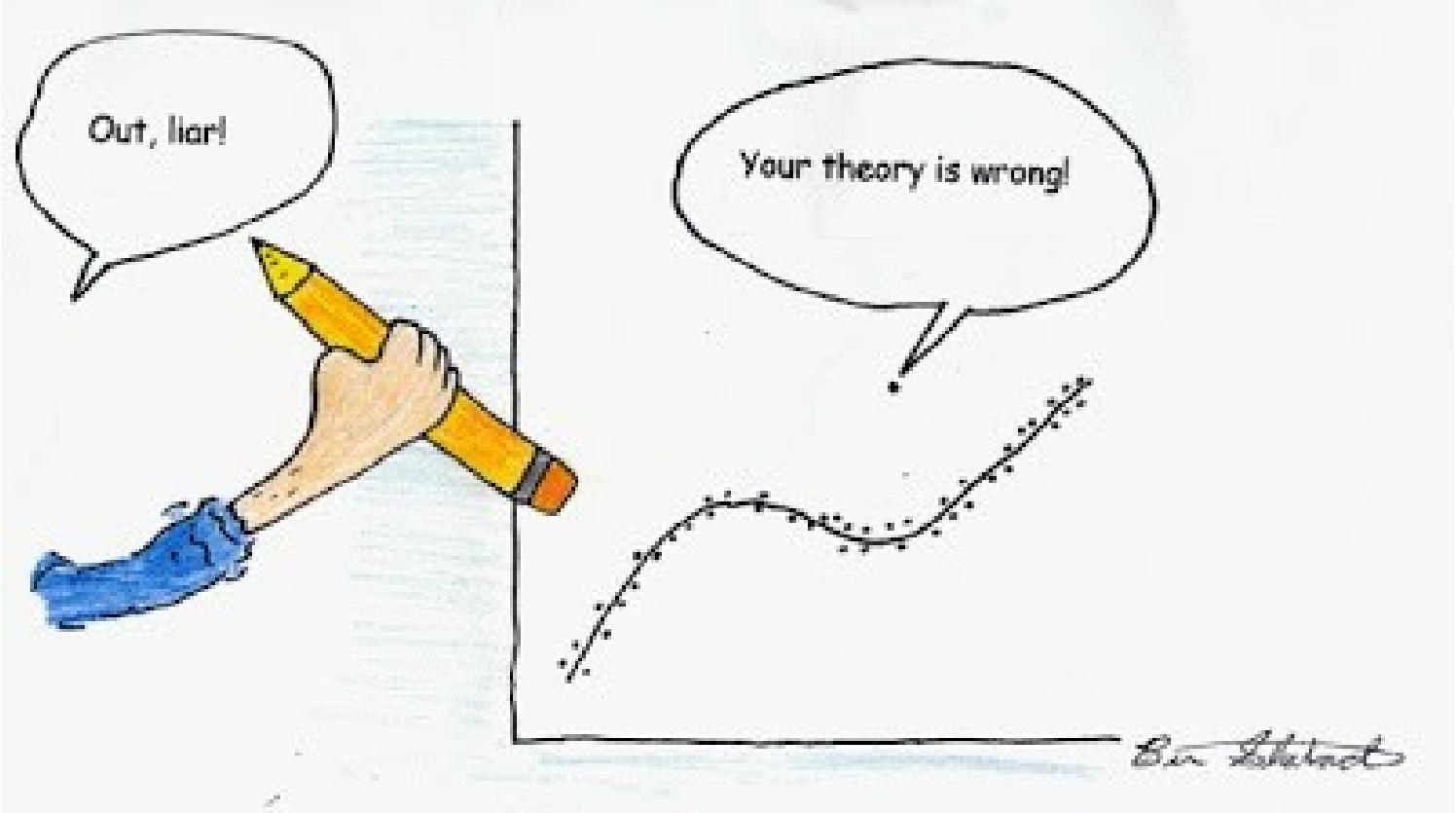
Solutions ?

```
influence.plot(m)
dfbetas.plot(m)
```





Valeurs extrêmes



Éliminer une valeur extrême n'est pas anodin !
Prudence et honnêteté !

Multicolinéarité entre variables explicatives

Conditions d'application des modèles

Les présupposés des GLMs

"model assumptions"

Par ordre d'importance (selon Gelman & Hill)

1) **adéquation**/validité
2) **linéarité** - additivité
3) **indépendance** des résidus
4) hypothèses sur la **variance** des résidus
5) hypothèses sur **distribution** des résidus

+ l'erreur de mesure des X doit être négligeable

Autres problèmes à garder en tête :

- **Outliers** : influence des données extrêmes (outliers)
- **Multicolinéarité** : indépendance des variables explicatives
- **Overfitting** : sélection de modèle nécessaire ?
- Faut-il **centrer** ou **standardiser** les données ?
- Quel **type de tests** (type I, II, III, ...), simulations, permutations,... ?

Multicolinéarité entre variables explicatives

Le problème

En régression multiple, on estime se qui se passe pour y quand x1 augmente d'une unité et que les autres variables explicatives restent inchangées.
Ceci est difficile voire impossible quand ces variables sont corrélées entre elles.

On parle de colinéarité lorsque deux variables sont corrélées.
On parle de multicolinéarité lorsqu'une variable est corrélée avec un ensemble d'autres variables.

Multicolinéarité entre variables explicatives

Le problème

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

1) les erreurs standard des paramètres augmentent
La précision des estimations est donc moindre et les p-valeurs augmentent (la puissance statistique diminue)

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

Multicolinéarité entre variables explicatives

Le problème

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

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

Multicolinéarité entre variables explicatives

Le problème

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

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

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

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

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

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

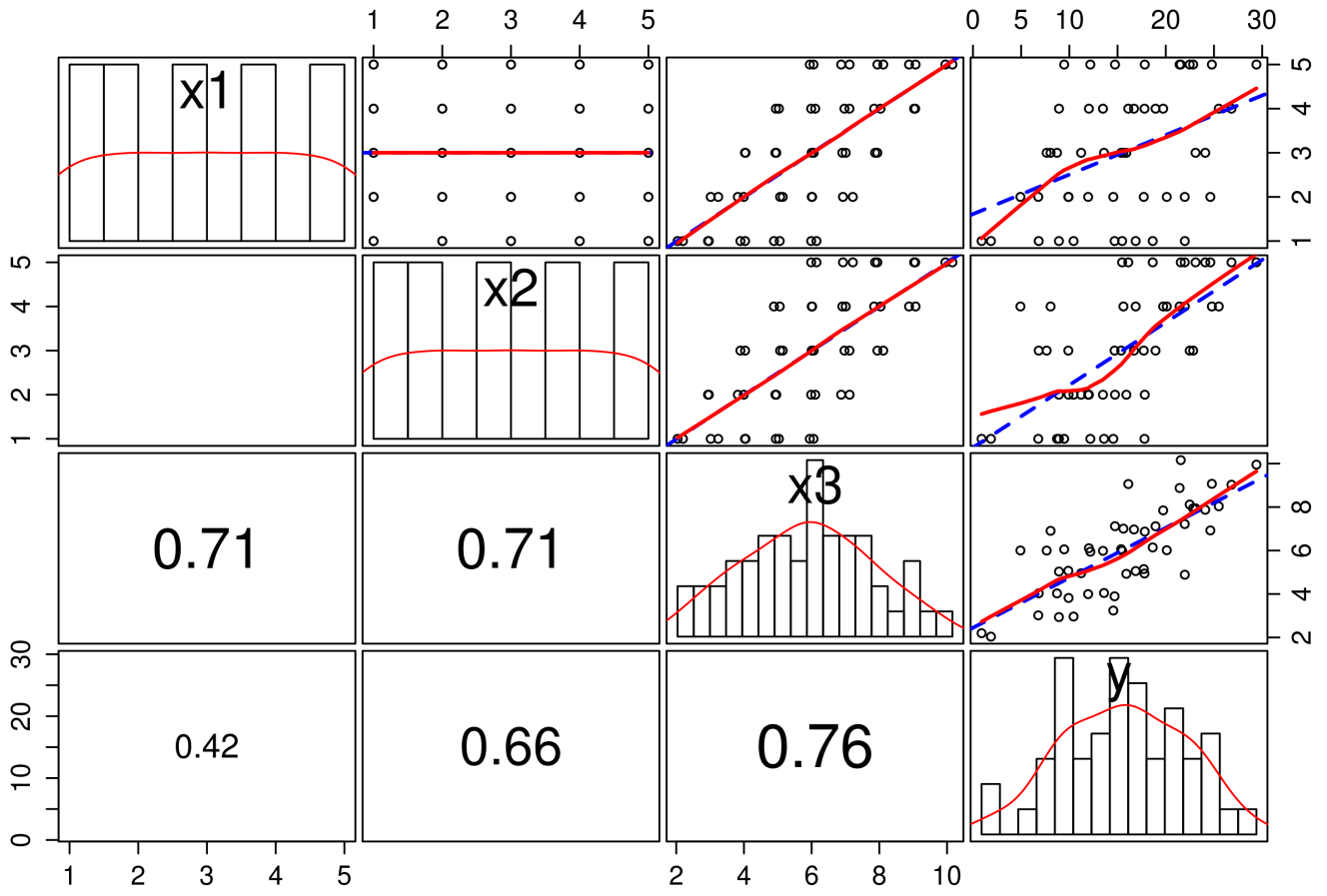
Si on les met dans le même modèle : les erreurs standard sont multipliées par 10 et on serait tenté de dire que la hauteur du corps est liée à la longueur du bras gauche mais pas du bras droit, ce qui est faux !

Multicolinéarité entre variables explicatives

Diagnostique

La visualisation des matrices de corrélations (SPLOMs, heatmaps) est très utile dans l'exploration préliminaire des données.
Mais elle n'est pas suffisante car elle ne montre que les corrélations 2 à 2.
Elle permet de voir la colinéarité mais pas la multicolinéarité.

```
> pairs2(cbind(x1, x2, x3, y))
```



Multicolinéarité entre variables explicatives

Diagnostique

Les VIFs ("Variance Inflation Factors") permettent de mesurer directement la multicolinéarité et d'interpréter l'effet sur le modèle :

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

```
> library(car)
> mod <- lm(y ~ x1 + x2)
> vif(mod)
      x1       x2
123.7848 123.7848
> sqrt(vif(mod))
      x1       x2
11.12586 11.12586
```

Lorsqu'il y a des variables qualitatives on peut utiliser les GVIFs (VIFs généralisés). Les erreurs standard sont multipliées par la 3ème colonne (sans en prendre la racine carrée)

```
> vif(mod)
                    GVIF Df GVIF^(1/(2*Df))
fertilizer             3  1        1.732051
variety                9  2        1.732051
fertilizer:variety    15  2        1.967990
```

Multicolinéarité entre variables explicatives

Diagnostique

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

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

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

Corrélation 2 à 2 relativement modérée

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

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

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

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

Ajout de x3 :
les erreurs standard sont multipliées par 14 à 20
Plus rien n'est significatif
Les coefficients changent

Multicolinéarité entre variables explicatives

Solutions ?

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

La colinéarité est une caractéristique des données.
Lorsque deux variables trop corrélées on ne peut tout simplement pas voir l'effet de l'une sur y indépendamment de l'autre...

Si pour répondre à la question il est nécessaire d'examiner l'effet de x1 indépendamment de x2 et que ces variables sont corrélées, la seule possibilité est de récolter un nouveau jeu de données où on manipule les valeurs des deux variables explicatives (approche expérimentale)

Lorsque ce n'est pas possible (souvent!) on doit reformuler la question et accepter les limites des données...

Multicolinéarité entre variables explicatives

Solutions ?

Lorsque ce n'est pas possible (souvent!) on doit reformuler la question et accepter les limites des données...

Si des variables corrélées posent problème :

1) les **grouper** : pex prendre la moyenne des bras gauche et droit, sommer leurs valeurs,...

2) **n'en garder qu'une**, celle qui est la plus biologiquement pertinente et garder en mémoire pour l'interprétation que toute relation pourrait être due en fait aux variables que l'on a retirées

3) **PCR (Principal Component Regression)**

Utiliser les composantes principales d'une PCA comme variables explicatives. L'avantage est qu'on garde toute l'information mais le gros inconvénient est l'interprétation biologique qui est plus difficile.

La PLS (Penalized Least Squares) peut être plus adaptée dans ce cas.
Elle produit de nouvelles variables non corrélées résumant les variables d'origine comme la PCA mais elles ont été construites de façon à maximiser la covariance avec Y

Multicolinéarité entre variables explicatives

Solutions ?

4) centrage

Lorsque des paramètres impliqués dans des interactions ou des termes polynomiaux sont fortement corrélés, le fait de **centrer ces variables** peut permettre de réduire les corrélations.

Le centrage peut aussi réduire les corrélations entre d'autres paramètres (pentes et intercepts pex) dans certaines situations (en particulier dans les modèles mixtes) et faciliter la convergence des algorithmes

Multicolinéarité entre variables explicatives

Solutions ?

Sélection automatique des variables : une fausse bonne idée ?

Il existe des méthodes qui permettent de réduire automatiquement les dimensions d'un modèle et donc de potentiellement éliminer automatiquement les variables corrélées avec d'autres.
(sélection de modèles par AIC, arbres de régression et extensions,...)

C'est une approche qui peut être efficace si le but est uniquement la prédiction.
Le but est alors d'avoir des estimations sans trop de variance ni biais quel que soit le moyen d'y arriver.

Si le but est un modèle explicatif (on veut interpréter les variables choisies)
c'est une approche qui peut être trompeuse...

1) Le choix de la variable corrélée qui reste dans le modèle est souvent assez arbitraire et masque l'importance potentielle des autres.

2) Ces méthodes permettent souvent de classer les variables par ordre d'importance. La corrélation va le plus souvent artificiellement diminuer ces mesures d'importance.

Overfitting

Conditions d'application des modèles

Les présupposés des GLMs

"model assumptions"

Par ordre d'importance (selon Gelman & Hill)

1) **adéquation**/validité
2) **linéarité** - additivité
3) **indépendance** des résidus
4) hypothèses sur la **variance** des résidus
5) hypothèses sur **distribution** des résidus

+ l'erreur de mesure des X doit être négligeable

Autres problèmes à garder en tête :

- **Outliers** : influence des données extrêmes (outliers)
- **Multicolinéarité** : indépendance des variables explicatives
- **Overfitting** : sélection de modèle nécessaire ?
- Faut-il **centrer** ou **standardiser** les données ?
- Quel **type de tests** (type I, II, III, ...), simulations, permutations,... ?

Overfitting

Le problème

Lorsque le nombre de variables explicatives augmente par rapport aux nombre de données (même si elles n'ont aucun lien avec y) :

les erreurs standard des coefficients augmentent
le R^2 augmente
la somme du carré des résidus (RSS) ou la déviance diminue

Lorsqu'on a autant de variables explicatives que de données même si elles n'ont aucun pouvoir explicatif, le R^2 est toujours = 1 et les RSS/déviance sont toujours = 0

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

On ne peut donc pas utiliser le R^2 pour comparer des modèles avec des nombres de paramètres différents !

Overfitting

Le problème

On dit que le modèle est "surparamétrisé" ("overfitted")

Le biais du modèle est faible
Mais sa variance est très élevée : un autre jeu de données donnera des résultats (coefficients) très différents

Des variables explicatives qui on un réel lien avec la variable dépendante peuvent devenir non significatives

Overfitting

Le problème

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

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

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

test pour x1 dans chacun des 10 modèles

Plus on a de variables (inutiles) dans le modèle plus l'erreur standard de x1 grimpe et moins elle est significative

Overfitting

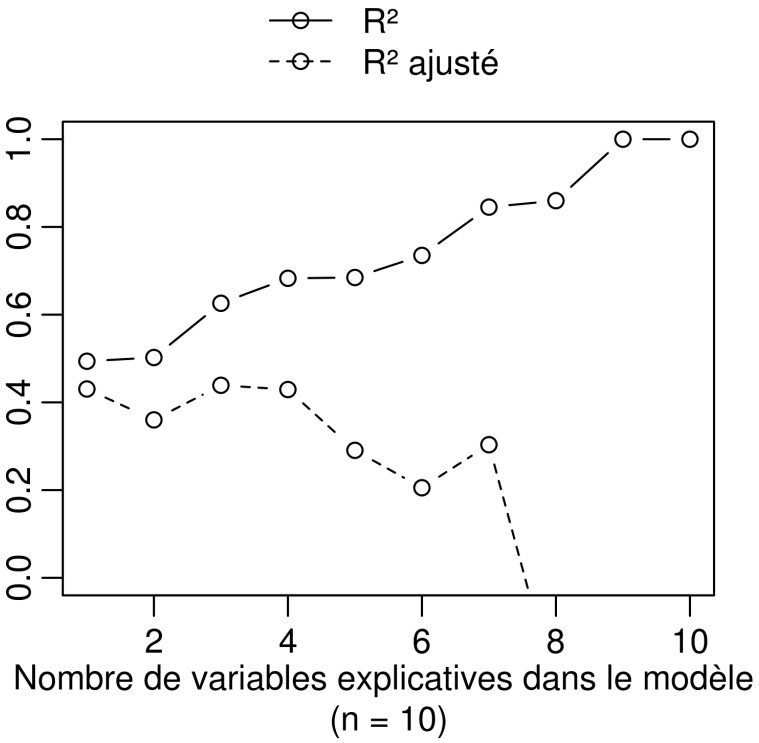
Le problème

Il existe une version ajustée du R^2 qui permet de comparer des modèles avec un nombre différent de paramètres et de données. Disponible uniquement pour les modèles gaussien...

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

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

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

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

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



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

Overfitting

Le problème

Pour cette raison, plusieurs statistiques estimant l'erreur faite par le modèle ne donnent pas une comparaison honnête entre modèles avec un nombre de paramètres différents :

R^2 - Déviance - RMSE - MAE - AUC

Certaines autres mesures tiennent compte du nombre de paramètres :

R^2 ajusté - AIC - BIC

Overfitting

Diagnostique

Il n'y a pas vraiment de méthode diagnostique directe

Pour les modèles gaussiens (lm), un R^2 ajusté très différent du R^2 est une indication de surparamétrisation.

On dit souvent (seuil arbitraire) qu'il faudrait idéalement avoir 10 observations indépendantes par variable explicative.

Overfitting

Solutions ?

Dans les approches expérimentales on essaye de **dimensionner son échantillonnage** de façon à pouvoir estimer correctement tous les paramètres nécessaires pour répondre à la question. (Analyse de puissance statistique)

Dans les méthodes plus observatives où le nombre de variables explicatives est potentiellement très grand, on applique en général des méthodes de **sélection de modèle** (pex par AIC) qui vont d'une manière ou d'une autre déduire les variables explicatives les plus importantes et réduire les dimensions du modèle.

L'objectif est alors de trouver le juste équilibre - pour un nombre d'observations donné - entre un modèle avec trop peu de variables (biais élevé mais variance faible) et un modèle avec trop de variables (biais faible mais variance élevée).

Overfitting

Solutions ?

Sélection de modèles
Model averaging / Shrinkage methods

Il existe de nombreuses méthodes qui permettent de réduire automatiquement les dimensions d'un modèle.

Un des problèmes de ces méthodes est qu'il existe souvent de nombreux modèles presque aussi bons (grande incertitude sur le meilleur modèle).

D'autres méthodes gardent toutes les variables mais tirent certains coefficients (shrinkage) vers 0 ce qui réduit la variance (des prédictions)

Stepwise sélection (AIC ou BIC)
Modèle avec AIC ou BIC minimum (parmi 2^k modèles)
Lasso Régression
Ridge regression
AIC/BIC model averaging

Centrer / standardiser les x ?

NB : il ne s'agit pas d'un problème mais plutôt de deux techniques qui peuvent faciliter l'interprétation et réduire certains problèmes

Conditions d'application des modèles

Les présupposés des GLMs

"model assumptions"

Par ordre d'importance (selon Gelman & Hill)

1) **adéquation**/validité
2) **linéarité** - additivité
3) **indépendance** des résidus
4) hypothèses sur la **variance** des résidus
5) hypothèses sur **distribution** des résidus

+ l'erreur de mesure des X doit être négligeable

Autres problèmes à garder en tête :

- **Outliers** : influence des données extrêmes (outliers)
- **Multicolinéarité** : indépendance des variables explicatives
- **Overfitting** : sélection de modèle nécessaire ?
- Faut-il **centrer** ou **standardiser** les données ?
- Quel **type de tests** (type I, II, III, ...), simulations, permutations,... ?

Centrer / standardiser les x ?

Quand faut-il centrer les données ?

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

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

Centrer peut être utile :
- Pour avoir un Intercept interprétable ou placé à un endroit où il est utile et où son erreur standard est raisonnable
- pour éviter une trop grande corrélation entre certains paramètres

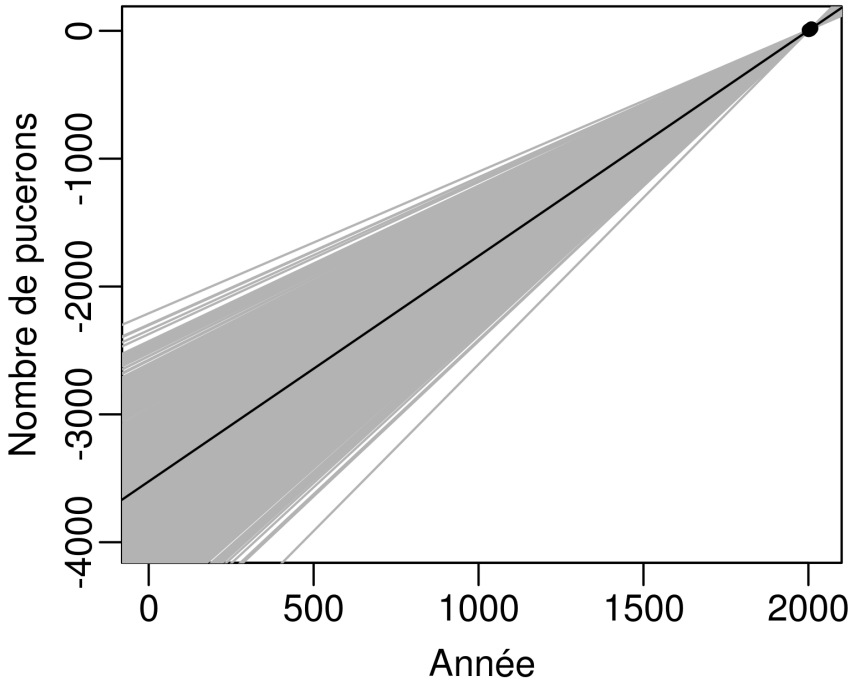
En pratique : surtout utile lorsqu'on a des interactions ou des termes polynomiaux pour réduire la corrélation entre ces termes et faciliter leur interprétation.

Centrer / standardiser les x ?

Quand faut-il centrer les données ?

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

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

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

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



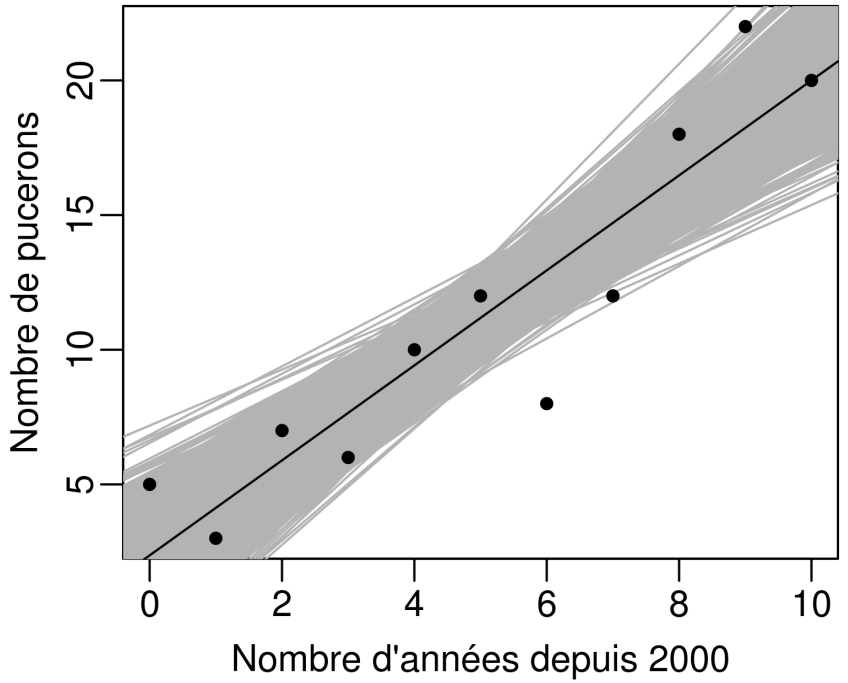
Centrer / standardiser les x ?

Quand faut-il centrer les données ?

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

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

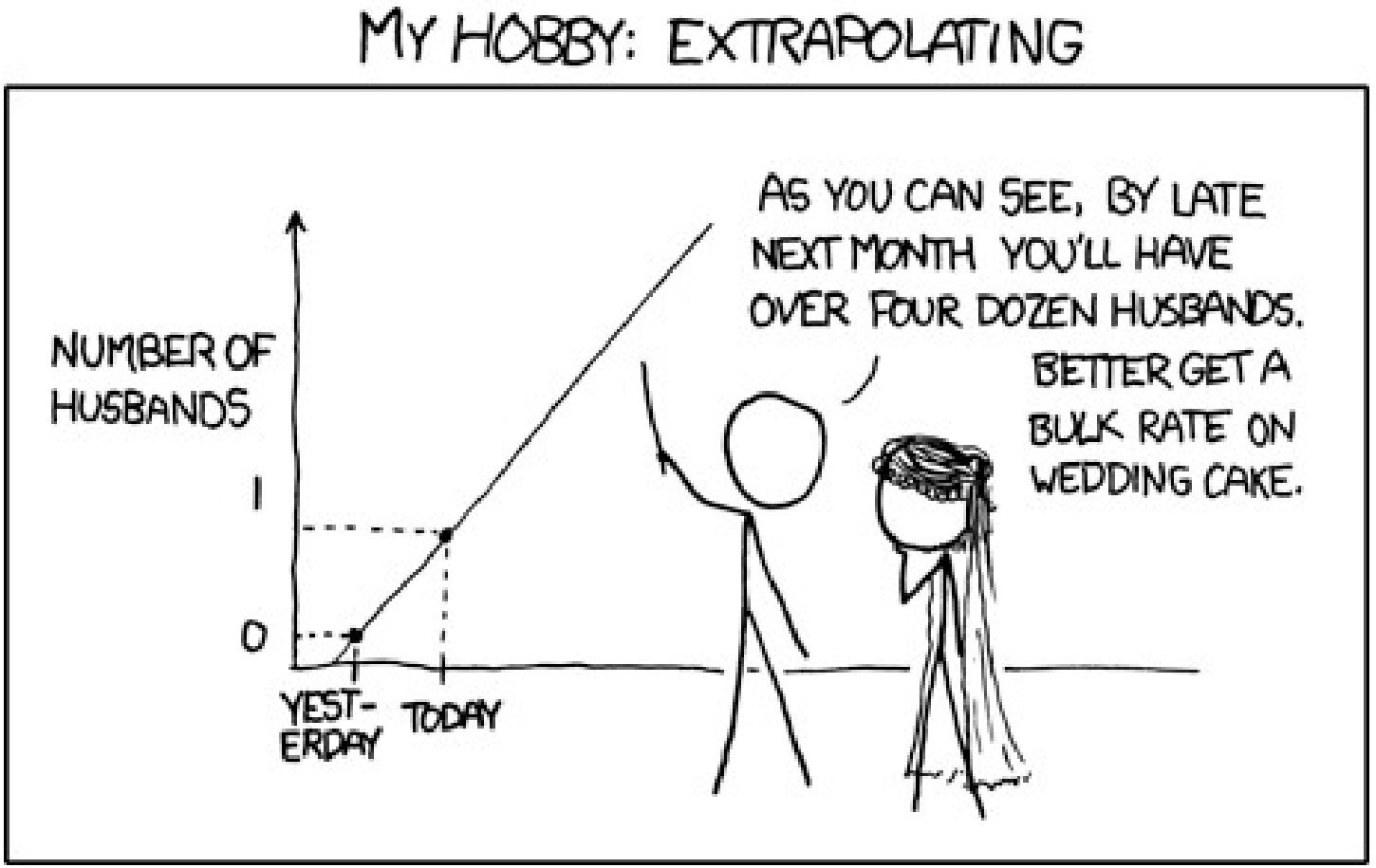
Si on centre la variable x
sur l'année 2000,
l'intercept estime maintenant
le nombre de pucerons en 2000
(~2.36 pucerons).

NB : la pente reste inchangée.



Centrer / standardiser les x ?

Attention à l'extrapolation !



Centrer / standardiser les x ?

Quand faut-il standardiser les variables explicatives ?

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

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

Centrer / standardiser les x ?

Quand faut-il standardiser les variables explicatives ?

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

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

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

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

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

Pour éviter la notation scientifique

Centrer / standardiser les x ?

Quand faut-il standardiser les variables explicatives ?

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

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

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

Centrer / standardiser les x ?

Quand faut-il standardiser les variables explicatives ?

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

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

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

Centrer / standardiser les x ?

Conclusion

Il n'est presque jamais obligatoire de centrer/standardiser les x

En particulier si vous interprétez les résultats finaux sur base des graphiques des modèles la facilité d'interprétation des coefficients est moins importante.
Les graphiques seront plus difficiles à interpréter ou a créer si les données sont centrées/standardisées.

Rien ne vous empêche d'estimer un modèle sur base des données standardisées pour l'interprétation directe des coefficients et un autre sur base des données brutes.

Le centrage peut être utile pour réduire la corrélation entre certains paramètres

Pour certaines méthodes autres que les GLM la standardisation est parfois quasi obligatoire (PCA sur variables dans des unités différentes, ridge regression,...)

Conditions d'application des modèles

Les présupposés des GLMs

"model assumptions"

Par ordre d'importance (selon Gelman & Hill)

1) **adéquation**/validité
2) **linéarité** - additivité
3) **indépendance** des résidus
4) hypothèses sur la **variance** des résidus
5) hypothèses sur **distribution** des résidus

+ l'erreur de mesure des X doit être négligeable

Autres problèmes à garder en tête :

- **Outliers** : influence des données extrêmes (outliers)
- **Multicolinéarité** : indépendance des variables explicatives
- **Overfitting** : sélection de modèle nécessaire ?
- Faut-il **centrer** ou **standardiser** les données ?
- Quel **type de tests** (type I, II, III, ...), simulations, permutations,... ?

Types de tests

Rappel : Méfiez vous des tests en présence d'interactions...

Ne pas utiliser les tests de type I (`anova(model)`) :
l'ordre des variables change les résultats
(sauf designs expérimentaux balancés)

En présence d'interactions, ne pas interpréter les effets principaux qui les composent (comme `drop1` le fait)

Si on veut quand même un tableau complet d'analyse de la variance (souvent assez pratique) utiliser plutôt les tests de type II (`car::Anova`)

Ne pas utiliser le type III (défaut dans de nombreux logiciels) à moins de savoir ce que vous faites

Si vous voulez être certain de ce que vous faites : construisez vous-même les modèles correspondant aux hypothèses que vous voulez tester et comparez les avec `anova(model1, model2)`

Adéquation / validité

Conditions d'application des modèles

Les présupposés des GLMs

"model assumptions"

Par ordre d'importance (selon Gelman & Hill)

1) **adéquation**/validité

2) **linéarité** - additivité

3) **indépendance** des résidus

4) hypothèses sur la **variance** des résidus

5) hypothèses sur **distribution** des résidus

+ l'erreur de mesure des X doit être négligeable

Autres problèmes à garder en tête :

- **Outliers** : influence des données extrêmes (outliers)
- **Multicolinéarité** : indépendance des variables explicatives
- **Overfitting** : sélection de modèle nécessaire ?
- Faut-il **centrer** ou **standardiser** les données ?
- Quel **type de tests** (type I, II, III, ...), simulations, permutations,... ?

Adéquation / validité

Hypothèse

Les données que l'on analyse doivent correspondre à la question posée...

La population échantillonnée doit correspondre au niveau auquel on veut pouvoir généraliser les données

Penser aussi aux bonnes pratiques expérimentales qui sont nécessaires autant que possible pour assurer la validité des conclusions :

randomisation des traitements et des mesures
contrôles pour éliminer certains facteurs de confusion
réplicats indépendants pour assurer des inférences valides

Adéquation / validité

C'est évidemment la condition la plus importante mais elle n'est pas toujours respectée ...

La variable dépendante doit être choisie et mesurée avec soin. Dans bien des cas on a cependant pas une mesure directe du phénomène que l'on veut étudier mais plutôt un indice plus ou moins représentatif (pex : détection d'une espèce, taille de population,...)

Les variables explicatives devraient aussi être sélectionnées avec soin. Dans les études observatives, n'utiliser que les variables dont on suspecte qu'elles ont réellement un lien biologique avec y

Adéquation / validité

Diagnostique / solutions

En général il est impossible de détecter des problèmes de validité/adéquation sur base des données.

Une bonne connaissance et un regard critique sur le phénomène étudié est nécessaire.
Du côté des solutions, la marge de manœuvre est en général très faible : il faut récolter de meilleures données ou tenir compte des limitations dans l'interprétation !

Dans certains cas des outils statistiques peuvent aider.
Pex lorsqu'on mesure la présence/absence d'une espèce sur différents sites on mesure en réalité une détection/non détection.
Certains types de modèles permettent de tenir compte de la probabilité de détection

Linéarité - additivité

Conditions d'application des modèles

Les présupposés des GLMs

"model assumptions"

Par ordre d'importance (selon Gelman & Hill)

1) **adéquation**/validité

2) **linéarité** - additivité

3) **indépendance** des résidus

4) hypothèses sur la **variance** des résidus

5) hypothèses sur **distribution** des résidus

+ l'erreur de mesure des X doit être négligeable

Autres problèmes à garder en tête :

- **Outliers** : influence des données extrêmes (outliers)
- **Multicolinéarité** : indépendance des variables explicatives
- **Overfitting** : sélection de modèle nécessaire ?
- Faut-il **centrer** ou **standardiser** les données ?
- Quel **type de tests** (type I, II, III, ...), simulations, permutations,... ?

Linéarité - additivité

Hypothèse

La variable dépendante est une fonction linéaire des variables explicatives de la forme :

$$y \sim a + b_1x_1 + b_2x_2 + b_3x_3 + ...$$

Ou pour les GLMs avec une fonction de lien g():

$$g^{-1}(y) \sim a + b_1x_1 + b_2x_2 + b_3x_3 + ...$$

Lorsqu'on ajoute des interactions ou des termes polynomiaux le modèle n'est plus uniquement "additif" mais aussi "multiplicatif"

Linéarité - additivité

Hypothèse

Par facilité*, on parlera de problèmes de linéarité quand la relation entre y et x ne suit pas une ligne droite.

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

La non linéarité affecte directement l'estimation des paramètres, la forme du modèle et les prédictions.

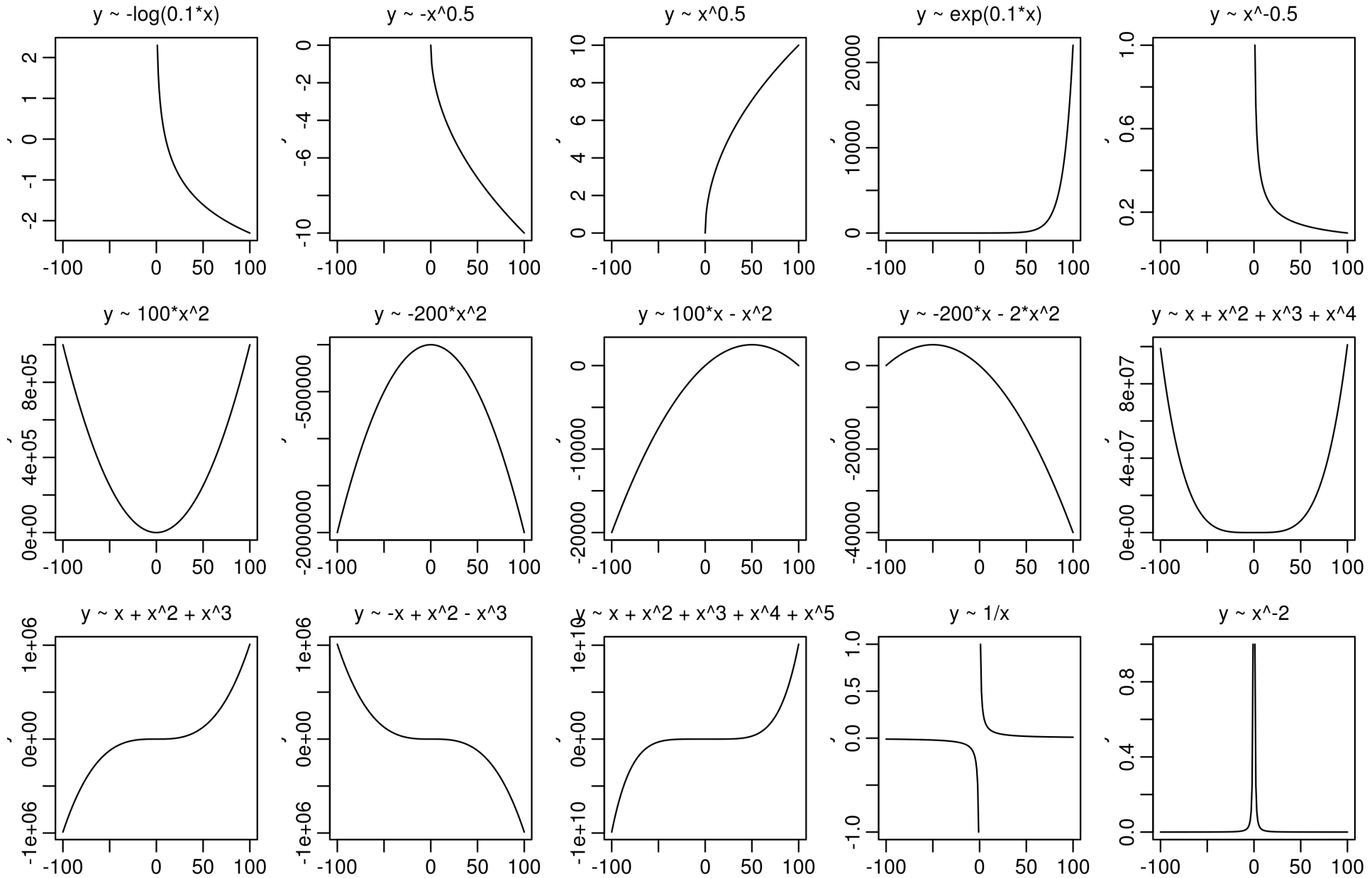
Les inférences sont correctes mais sur un mauvais modèle !!

Cependant les GLM sont très flexibles et permettent de modéliser de nombreuses relations qui n'ont pa sla forme d'une ligne droite

*Un modèle peut en réalité être dit linéaire sans être une droite ou plan

Linéarité - additivité

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



Linéarité - additivité

Diagnostic

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

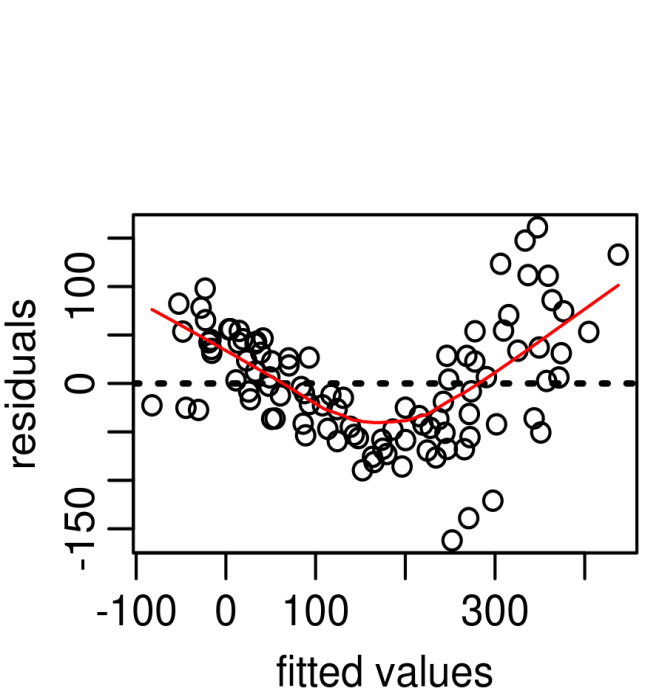
Mais ce n'est pas suffisant car souvent le bruit masque des problèmes de non linéarité.

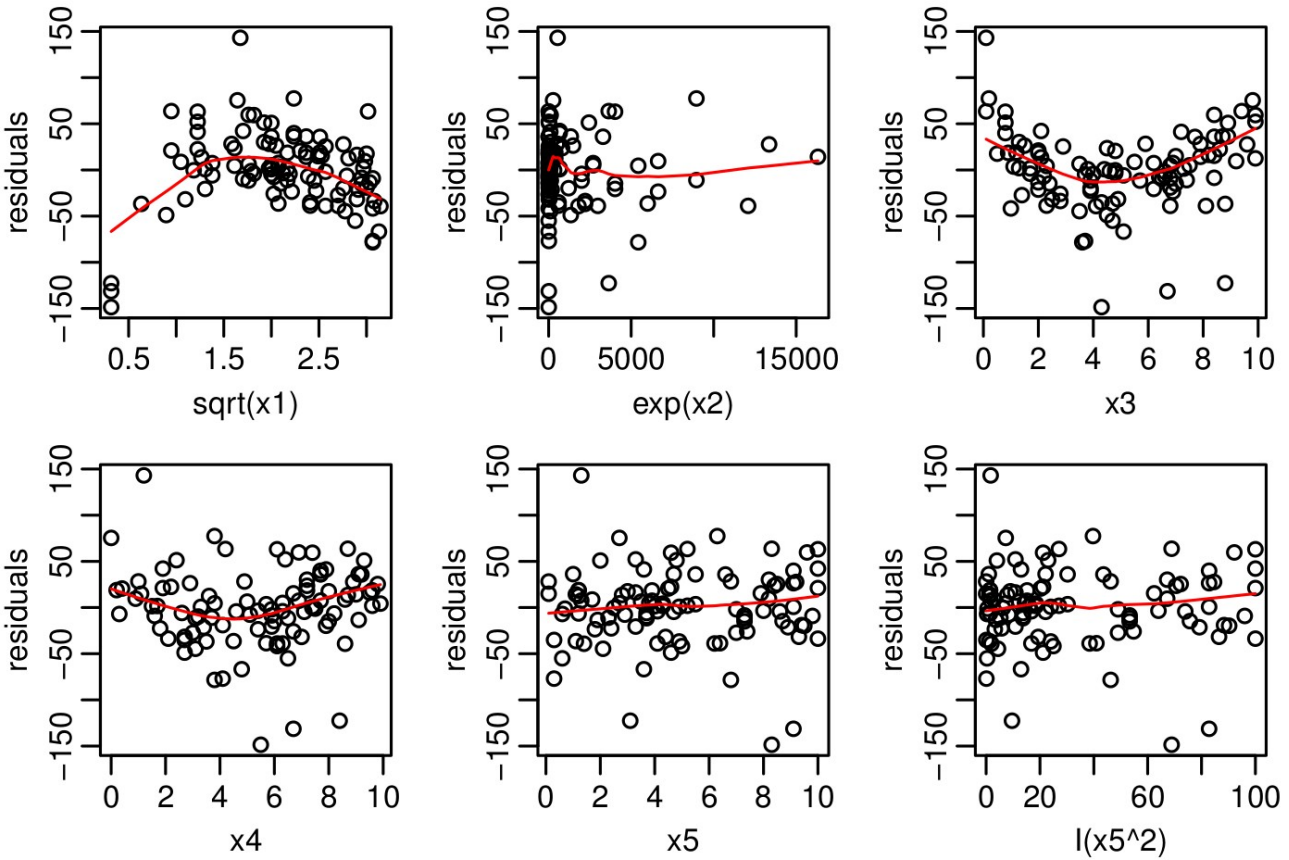
L'idéal est de faire des graphiques des résidus en fonction des valeurs prédites et en fonction de chaque variable explicative
pex avec `diagplot` et `diagplot2`

Si le nuage de point n'est pas uniformément réparti de part et d'autre du 0, le modèle surestime ou sous-estime les données dans une partie de l'espace du modèle.
Ce genre de pattern dénote en général une relation non linéaire ou parfois un problème d'additivité
(absence d'une interaction importante)

Linéarité - additivité

Diagnostic





Linéarité - additivité

Diagnostic

Il est également important de réfléchir sur les relations entre y et les x a priori en fonction de ce qu'on connaît du système étudié.

Par exemple si x représente une gamme environnementale très large, on peut s'attendre à ce que certaines espèces montrent une relation en cloche avec cette variable (optimum écologique) qui appelle une polynomiale d'ordre 2

Linéarité - additivité

Pistes de solutions

On peut soit adapter le GLM ou bien utiliser d'autres méthodes spécifiquement prévues pour des relations (hautement) non linéaires)

Avec des GLM :

- transformer les variables x : log, exp, sqrt, exposants,...
- transformer y (plus rarement)
- régression polynomiale : x + x^2 + x^3 ...
- transformer la variable x continue en variable qualitative
  ea pour effets multimodaux pex effets saisonnaux
- ajouter des interactions

NB : on transforme aussi souvent les variables pour d'autres raisons que la linéarisation (ea homogénéisation des variances, normalisation des résidus, diminuer l'effet de valeurs extrèmes,...)

Linéarité - additivité

Pistes de solutions

Avec des GLM :

NB : certaines transformations peuvent transformer une relation non additive en relation additive Pex :
si y = a*b*c alors log(y) = log(a) + log(b) + log(c)

NB : Au lieu de faire passer une droite à travers le nuage de points, on peut aussi simplement faire passer plusieurs segments de droites dans des zones où la relation est linéaire
(on crée une variable qualitative découpant la variable continue en zones et on ajoute l'interaction)

voir aussi MARS : Multivariate Adaptive Regression Splines

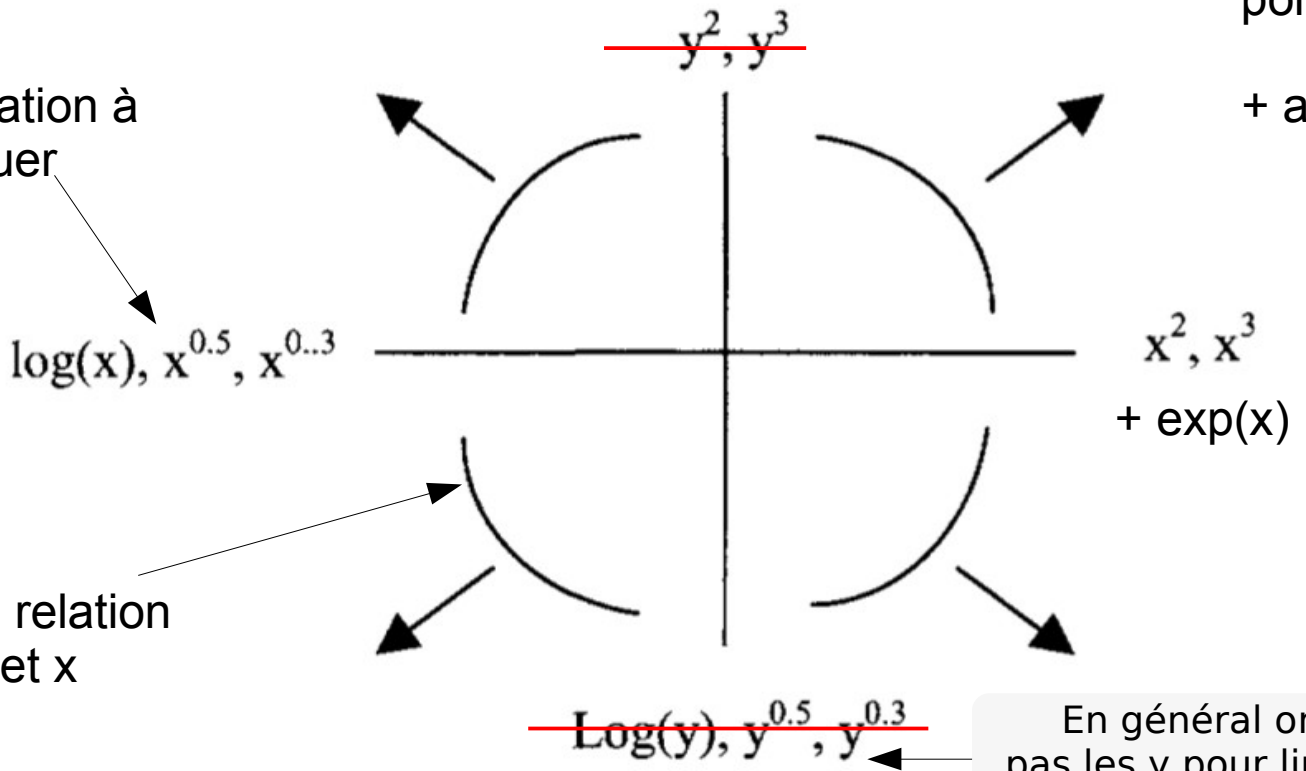
Linéarité - additivité

Comment choisir les transformations ?

"Mosteller and Tukey's bulging rule"
On procède par essais-erreurs en vérifiant l'effet de la transformation sur les graphiques de résidus et sur le "fit" du modèle (R^2, RMSE,...)

+ régressions polynomiales d'ordre 3 ou plus
+ ajout d'interactions

Transformation à appliquer



Forme de la relation entre y et x

En général on ne transforme pas les y pour linéariser la relation

Linéarité - additivité

Comment choisir les transformations ?

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

Si il y a un pattern dans le graphique résidus vs valeurs prédites mais pas dans les graphiques des résidus vs variables explicatives, transformez plutôt la variable dépendante y ou vérifiez qu'il ne manque pas d'interactions.

Si les patterns se trouvent dans les plots résidus vs variables explicatives, transformez plutôt les variables explicatives x

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

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

Linéarité - additivité

Régression polynomiale

Une régression polynomiale a par exemple la forme suivante :

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

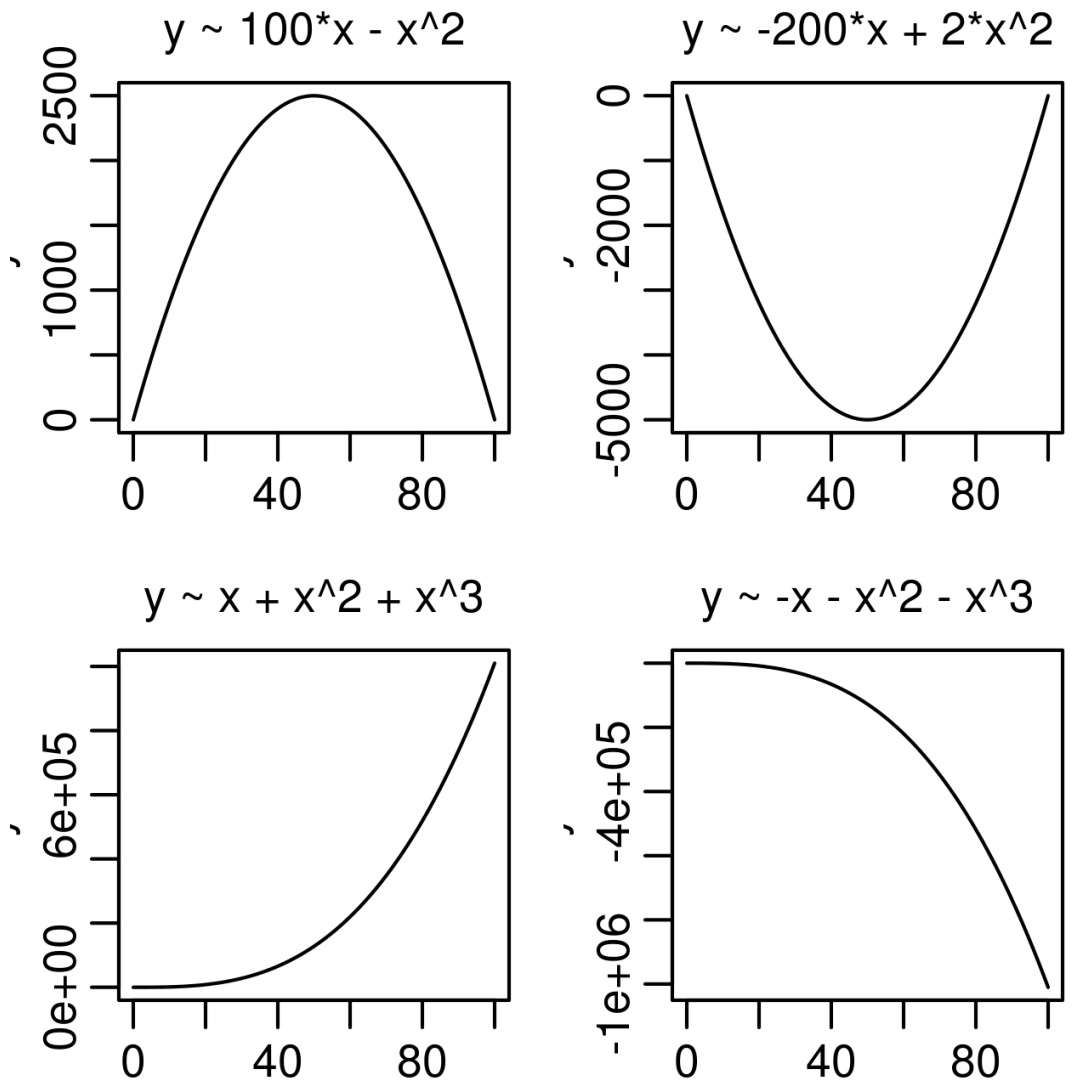
On dit qu'il s'agit d'une polynomiale d'ordre 2 pour x1 et d'ordre 3 pour x2. On appelle parfois x1^2 un "terme quadratique" et x2^3 un "terme cubique".

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

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

Linéarité - additivité

Régression polynomiale



Linéarité - additivité

Exemple Gaussien

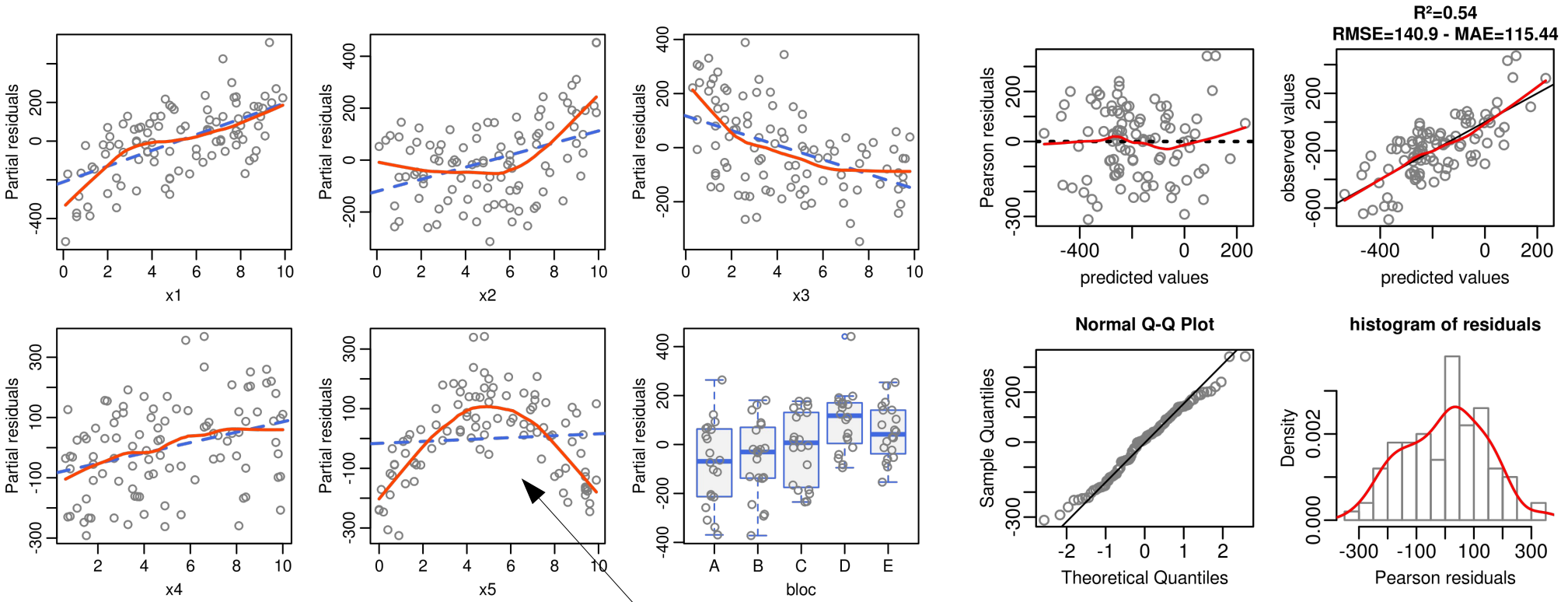
Génération d'un jeu de données gaussien non linéaire...

```
# create a dataset with non linear relationships...
set.seed(1)
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)
bloc = factor(rep(LETTERS[1:5], 20))

y <- 100 * log(x1) + 0.025* exp(x2) - 300*x3^0.25 + 2 * x4^2 + (x5-5) - 15*(x5-5)^2 +
    model.matrix(~bloc) %*% c( 0, 50, 100, 150, 200)   + rnorm(100, 0, 10)
d <- round(data.frame(y, x1, x2, x3, x4, x5),1)
d$bloc <- bloc
```

Linéarité - additivité

Exemple Gaussien

```
m <- lm(y ~ x1 + x2 + x3 + x4 + x5 + bloc  , data=d)
diagplot2(m)
diagplot(m)
```

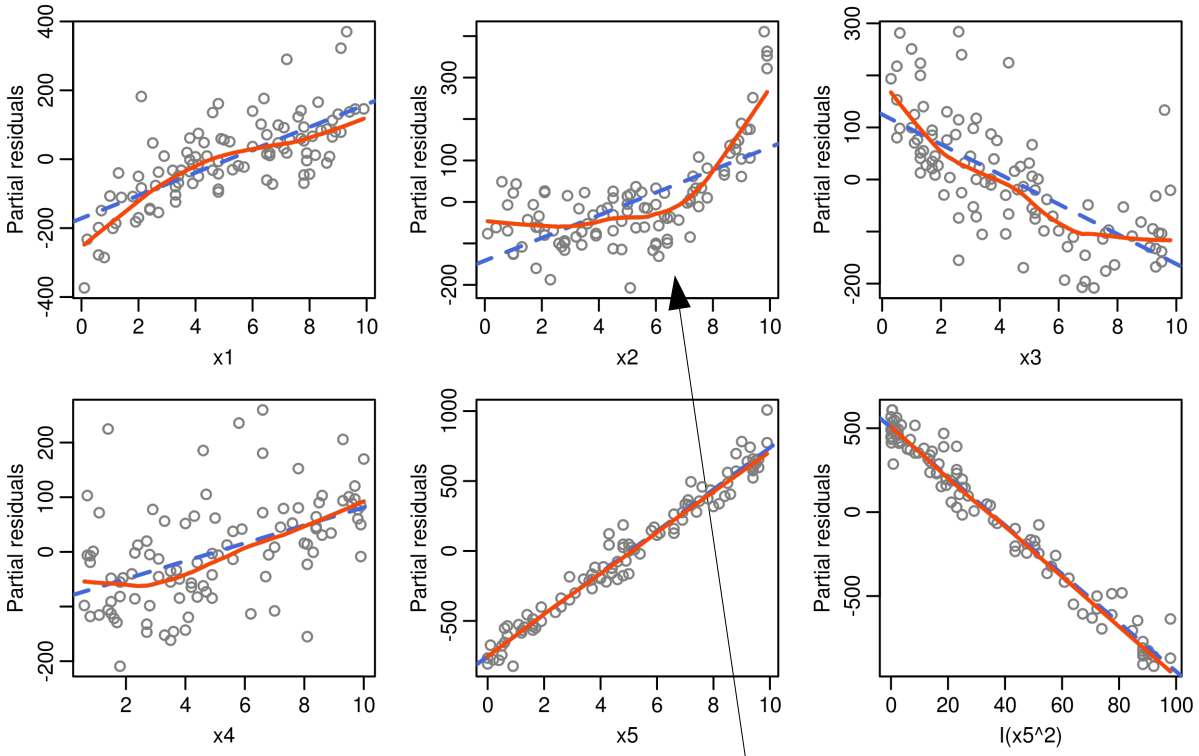


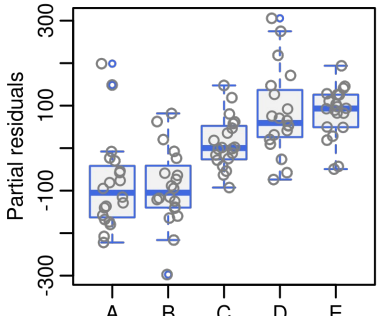
Courbure vers le haut
Pente nulle
--> on essaye une polynomiale sur x5

Linéarité - additivité

Exemple Gaussien

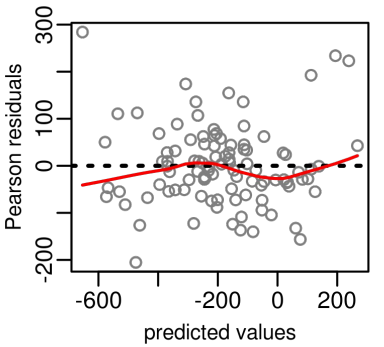
```
m <- lm(y ~ x1 + x2 + x3 + x4 + x5 + I(x5^2) + bloc  , data=d)
diagplot2(m)
diagplot(m)
```

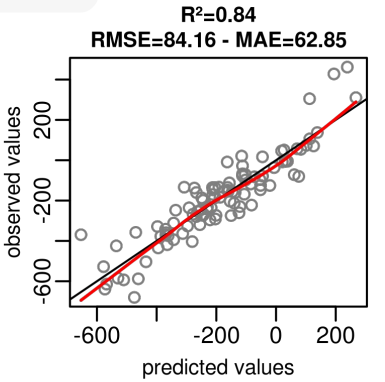


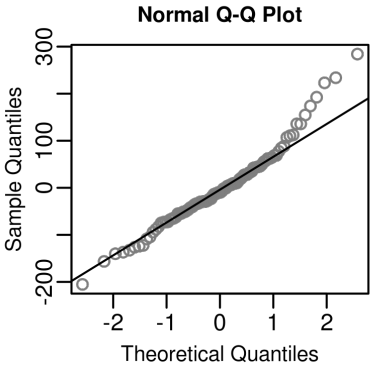


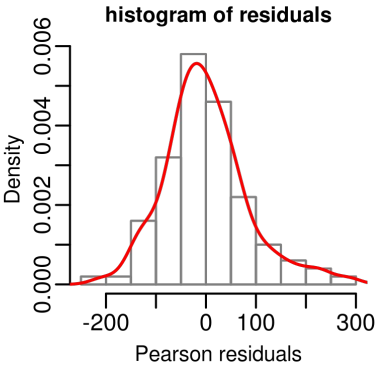
Courbure vers le bas
Pente positive
--> on essaye x2^2

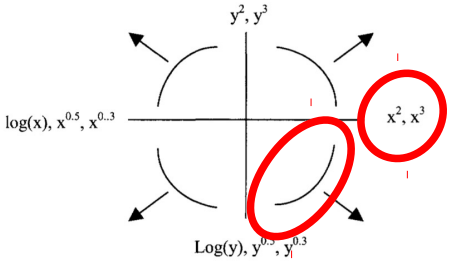
1 paramètre en plus mais amélioration très nette







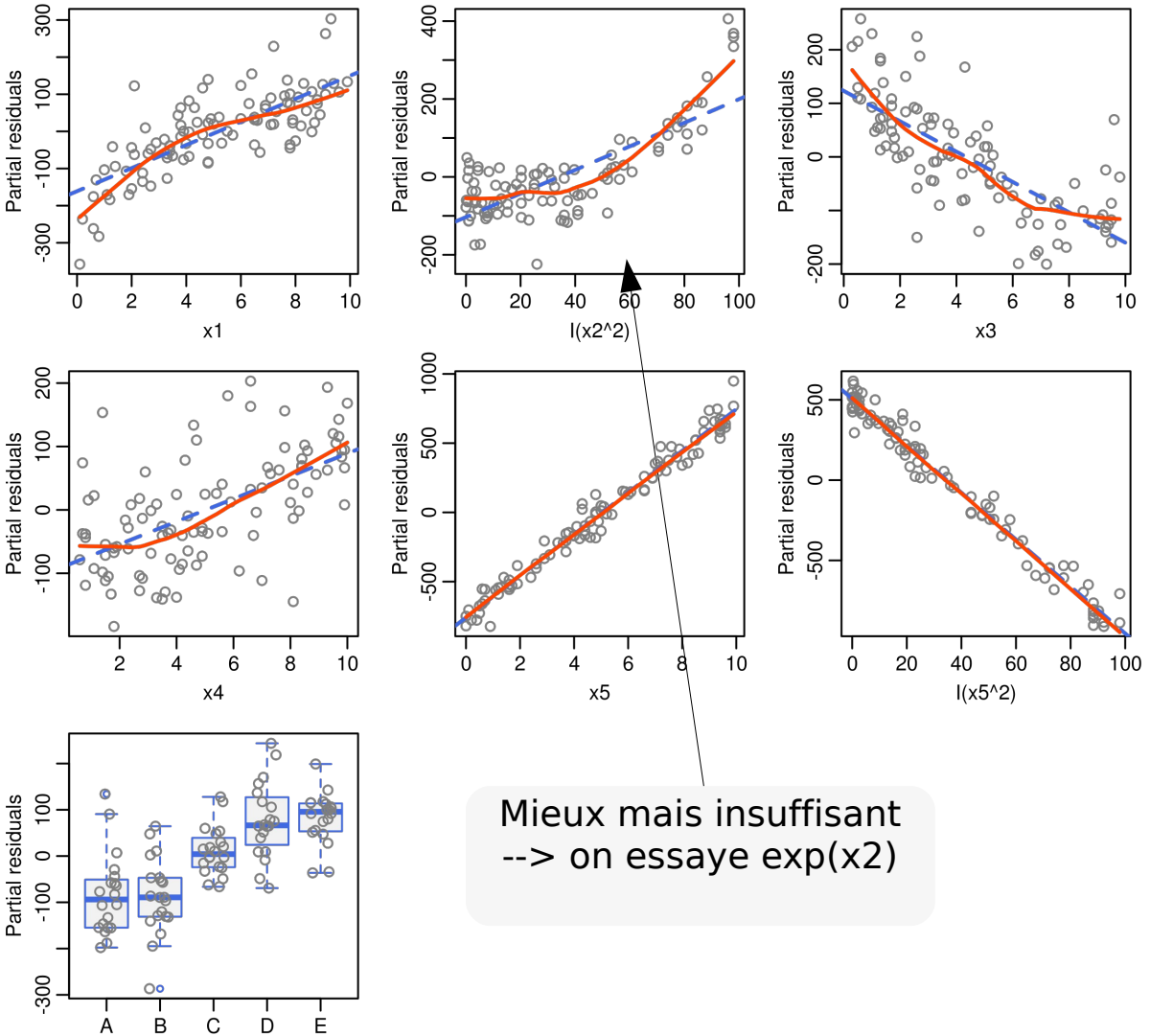




Linéarité - additivité

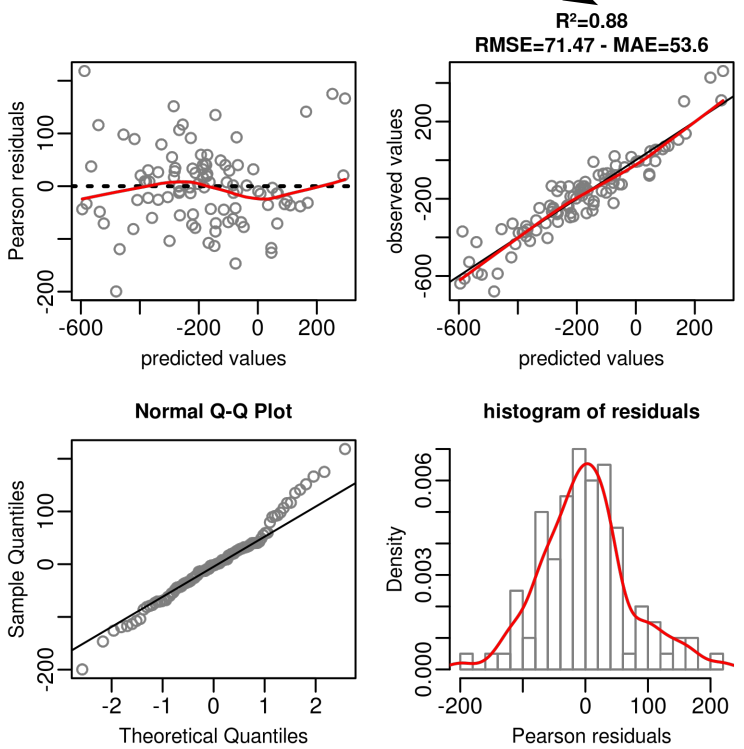
Exemple Gaussien

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



Mieux mais insuffisant
--> on essaye exp(x2)

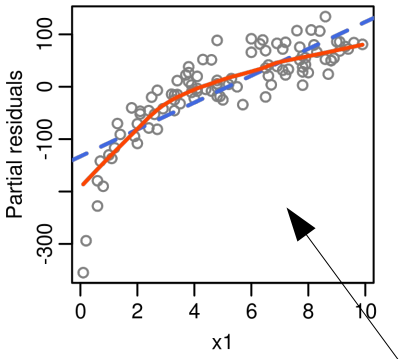
Légère amélioration

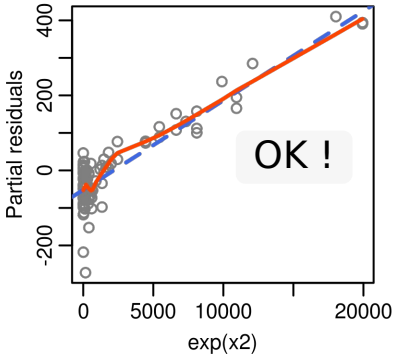


Linéarité - additivité

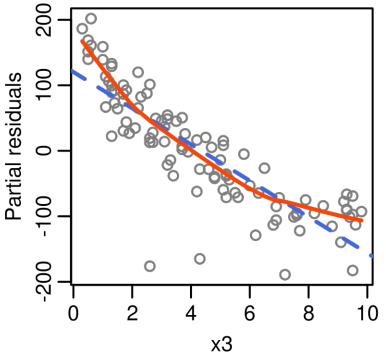
Exemple Gaussien

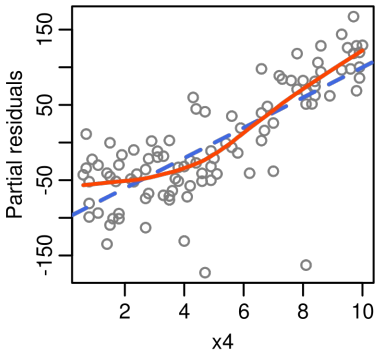
```
m <- lm(y ~ x1 + exp(x2) + x3 + x4 + x5 + I(x5^2) + bloc  , data=d)
diagplot2(m)
diagplot(m)
```

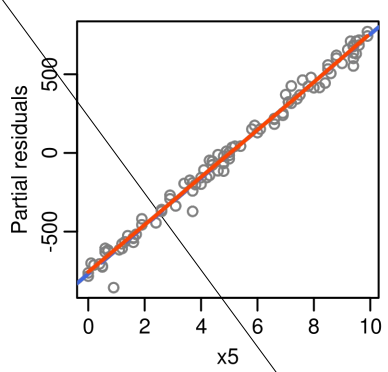


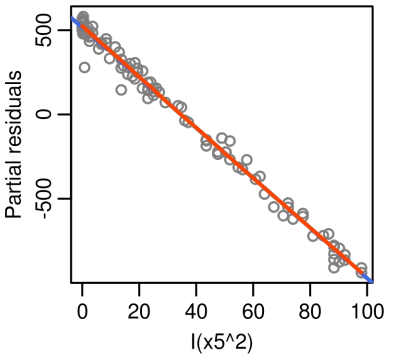


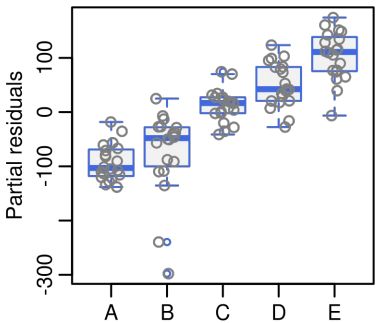
OK !





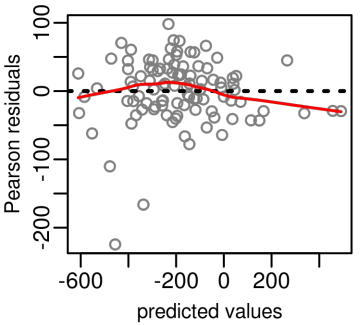


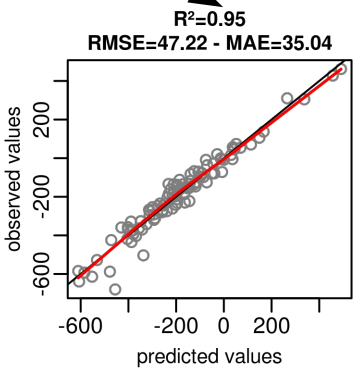


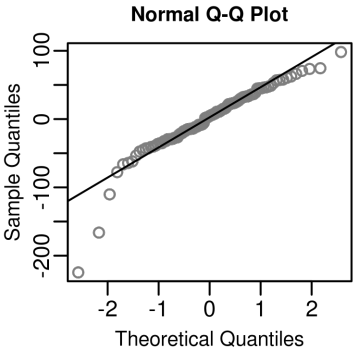


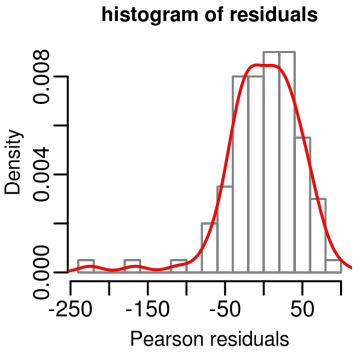
Courbure vers le haut
Pente positive
--> on essaye x1^0.5 ou log(x1),...

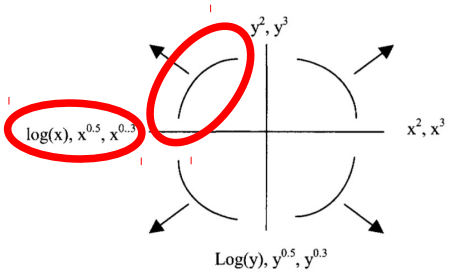
Encore mieux !









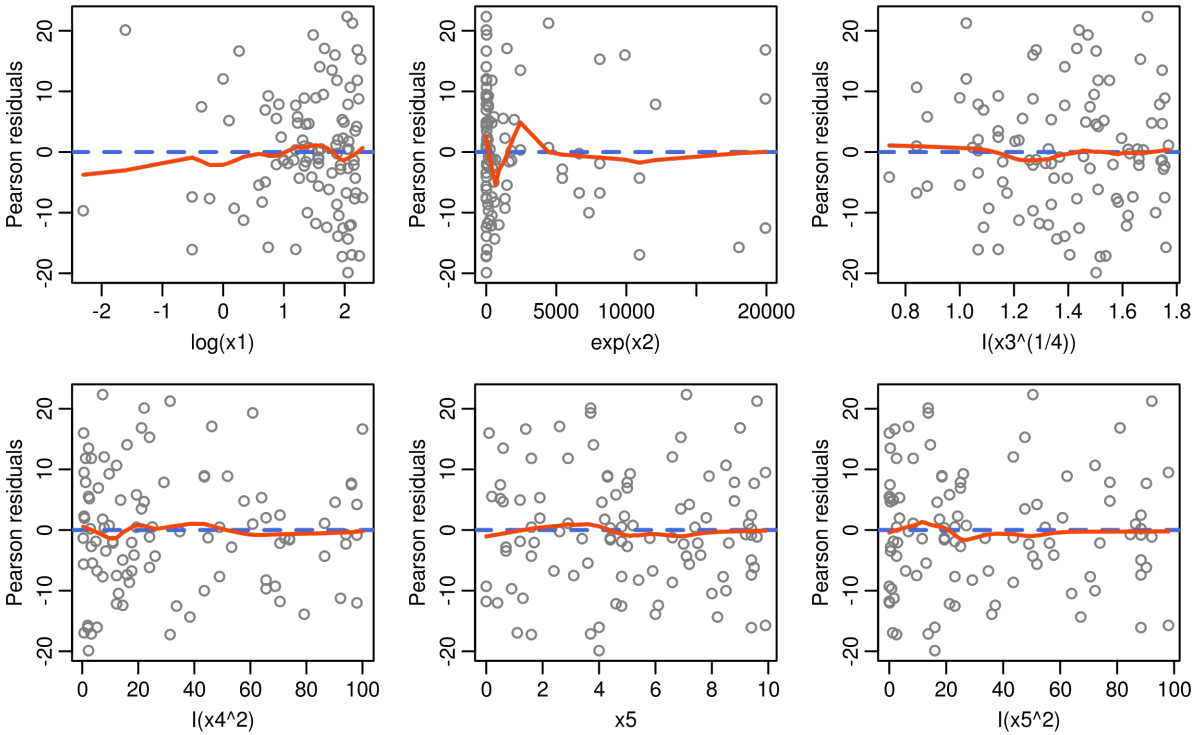


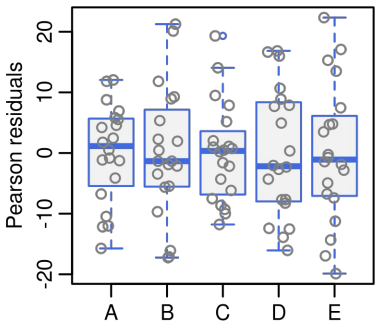
Linéarité - additivité

Exemple Gaussien

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

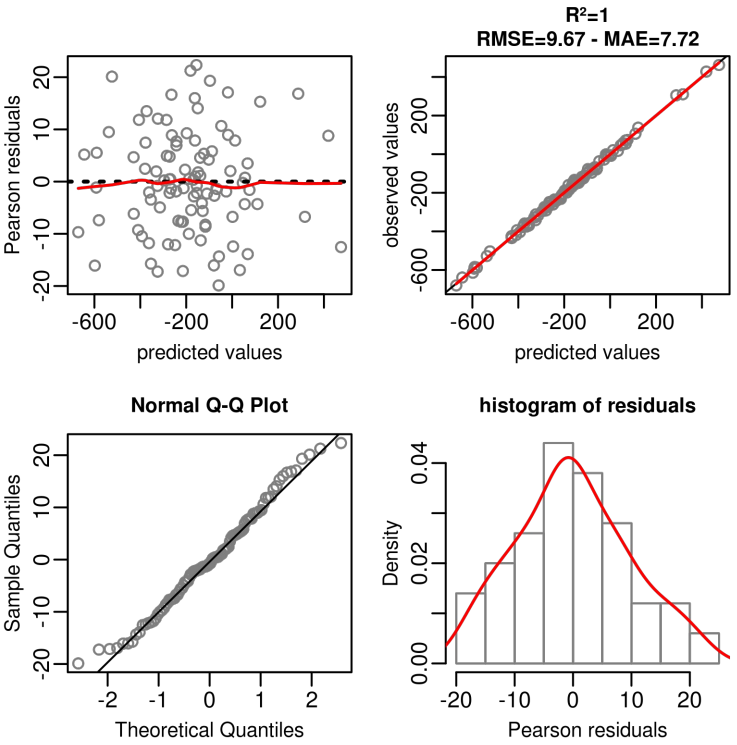
Résultat final après quelques essais/erreurs





On représente ici les résidus et plus les résidus partiels comme aux graphiques précédent pour bien voir qu'il n'y a plus de pattern dans les résidus.

Très nette amélioration par rapport au premier modèle !

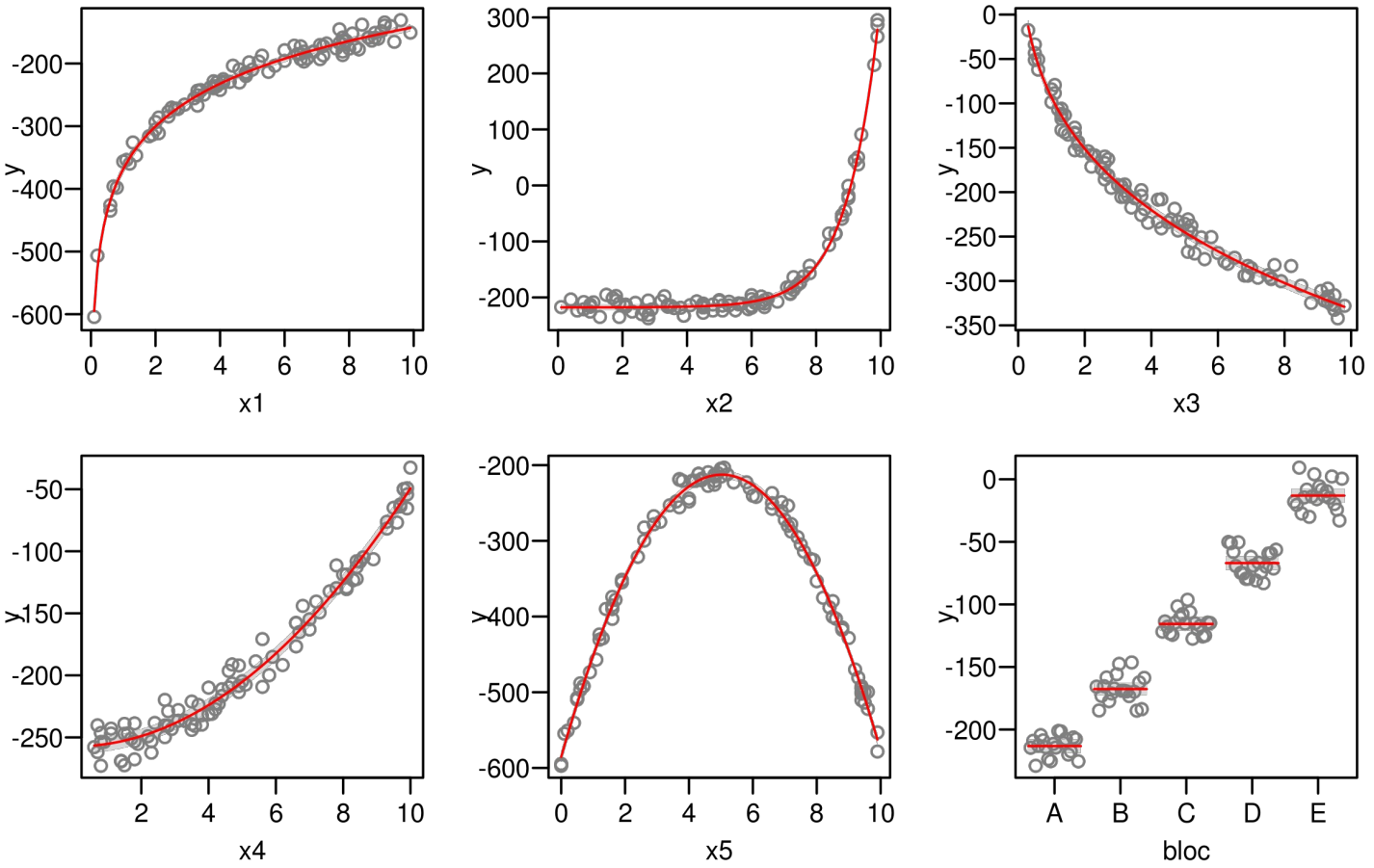


Linéarité - additivité

Exemple Gaussien

```
par(mfrow= c(2,3), mar = c(3,3,1,1), mgp = c(1.8, 0.6, 0))
visreg(m, points.par = list(cex = 1, pch = 1),
       line.par = list(lwd = 1, col = "red"))
```

Visualisation du modèle final avec visreg



Linéarité - additivité

Exemple Binomial

Avec un des données binaires ou de comptage on observe souvent des bandes dans les résidus : c'est normal !

L'interprétation des graphiques de résidus est souvent plus difficile.
Les courbes de lissage sont encore plus importantes ici !
Il peut être utile de rendre ces courbes plus sensibles (argument span plus petit que 1/3)

Exemple simulé sur base de paramètres observés dans un vrai jeu de données

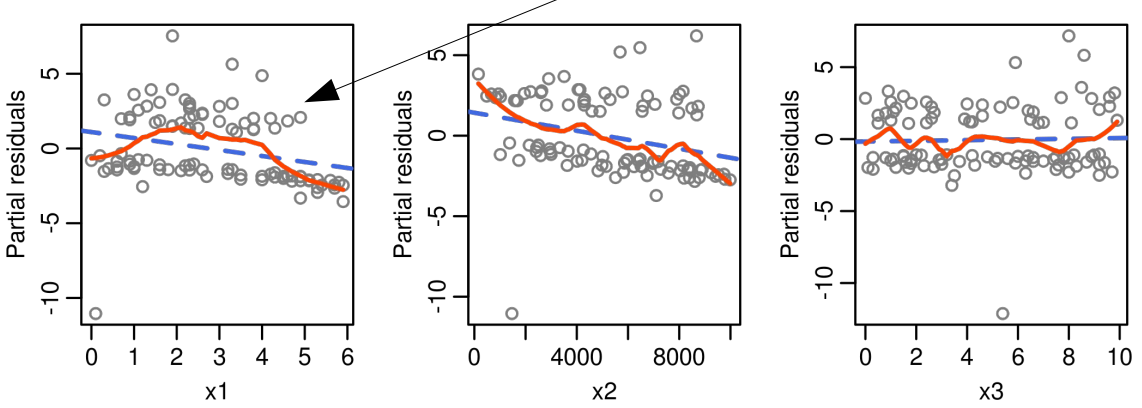
```
set.seed(12)
n <- 100
x1 <- runif(n, 0, 6)
x2 <- runif(n, 0, 10000)
x3 <- runif(n, 0, 10)
x4 <- runif(n, 0, 10)
bloc = factor(rep(LETTERS[1:5], n/5))

linpred <- 10 + 3 * x1 - 0.6 * x1^2 - 1.5 * log(x2)
y <- rbinom(n, p = plogis(linpred), size = 1)
d <- round(data.frame(y, x1, x2, x3, x4),1)
d$bloc <- bloc
```

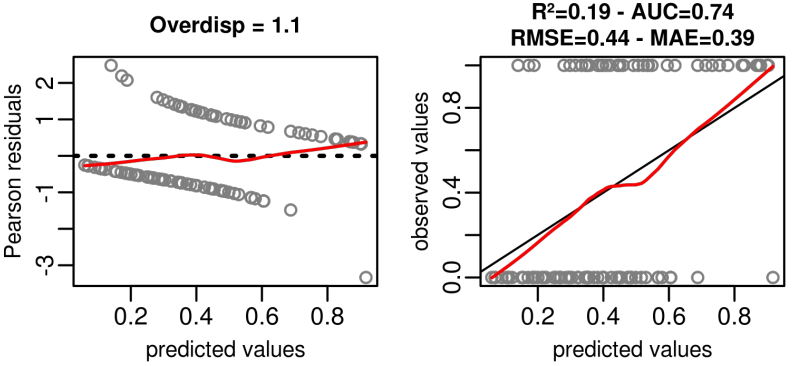
Linéarité - additivité

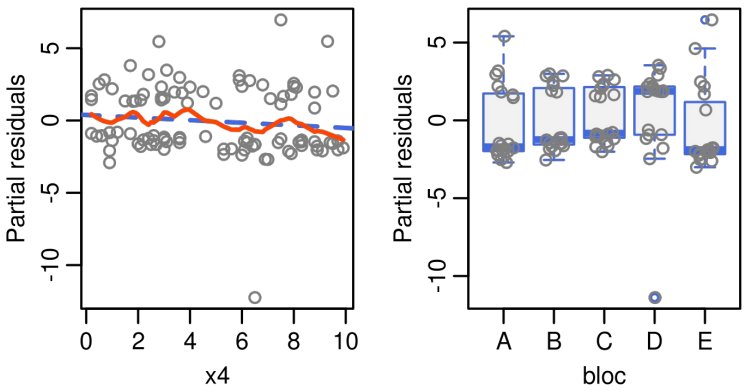
Exemple Binomial

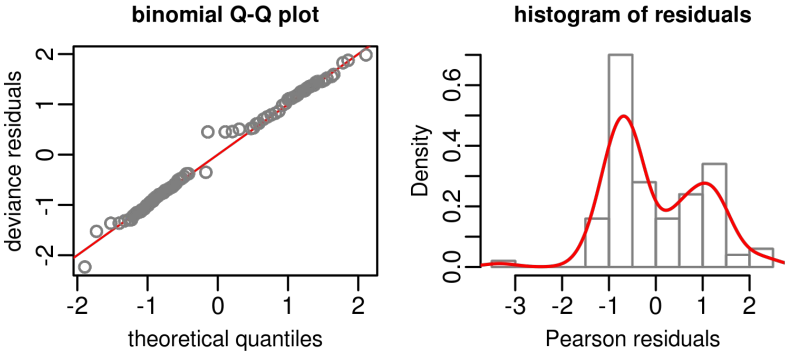
```
m <- glm(y ~ x1 + x2 + x3 + x4 + bloc , data=d, family = binomial)
diagplot2(m, span = 1/5)
diagplot(m)
```

Courbure vers le haut
Légère pente vers le bas
--> I(x1^2) --> mieux mais insuffisant
--> polynomiale x1 + I(x1^2)







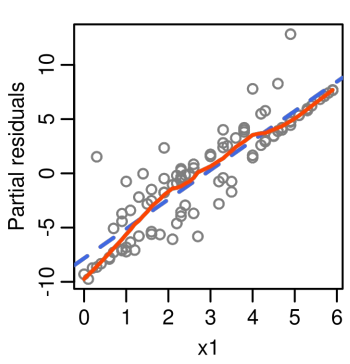


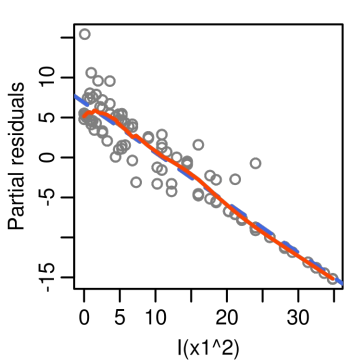
Linéarité - additivité

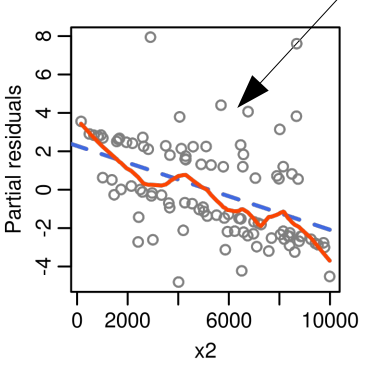
Exemple Binomial

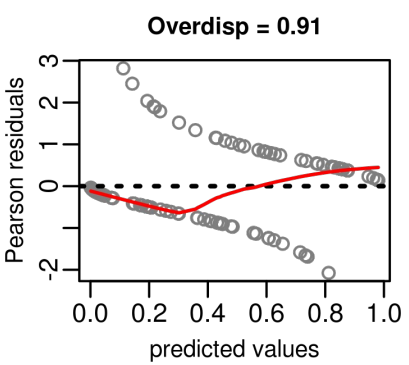
```
m <- glm(y ~ x1 + I(x1^2) + x2 + x3 + x4 + bloc , data=d, family = binomial)
diagplot2(m, span = 1/5)
diagplot(m)
```

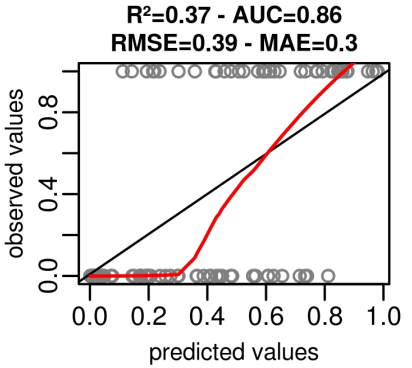
La relation avec x2 est en réalité logarithmique mais ce n'est pas visible ici.
Une relation linéaire est déjà une bonne approximation !!

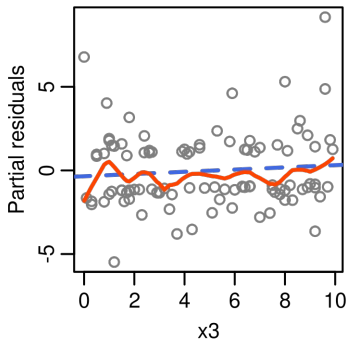


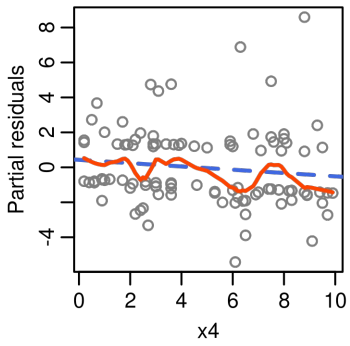


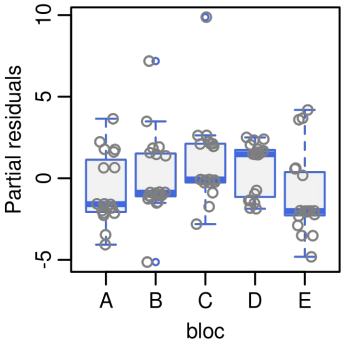


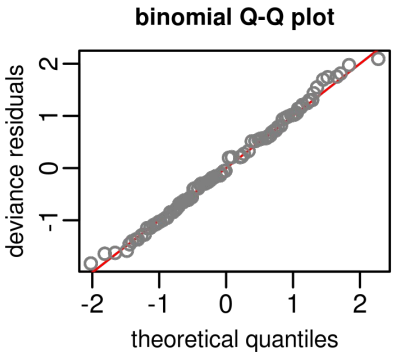


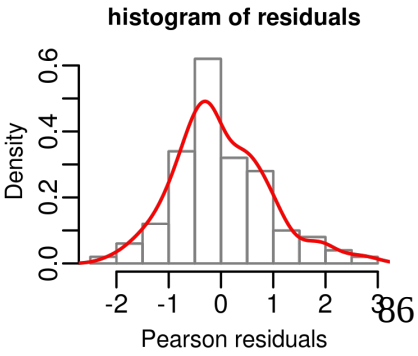












Linéarité - additivité

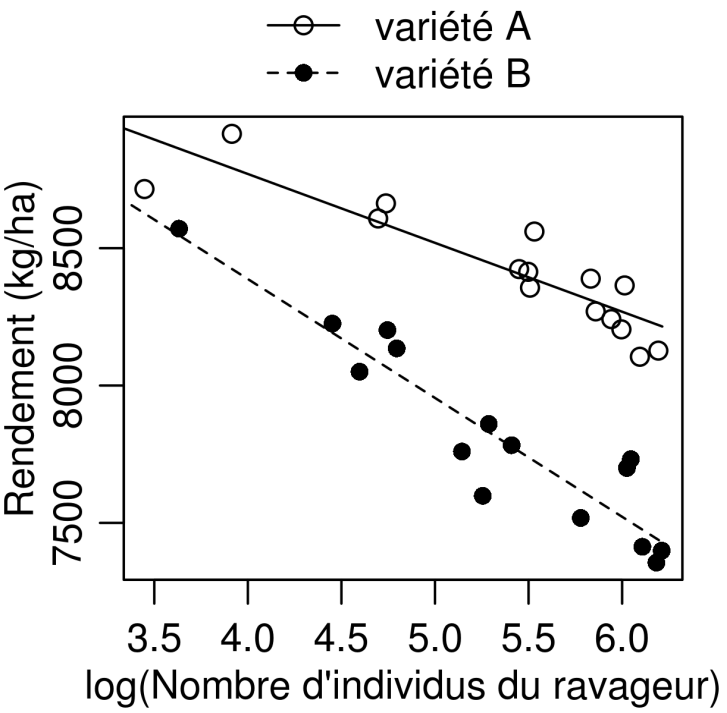
Représentation graphique des relations transformées

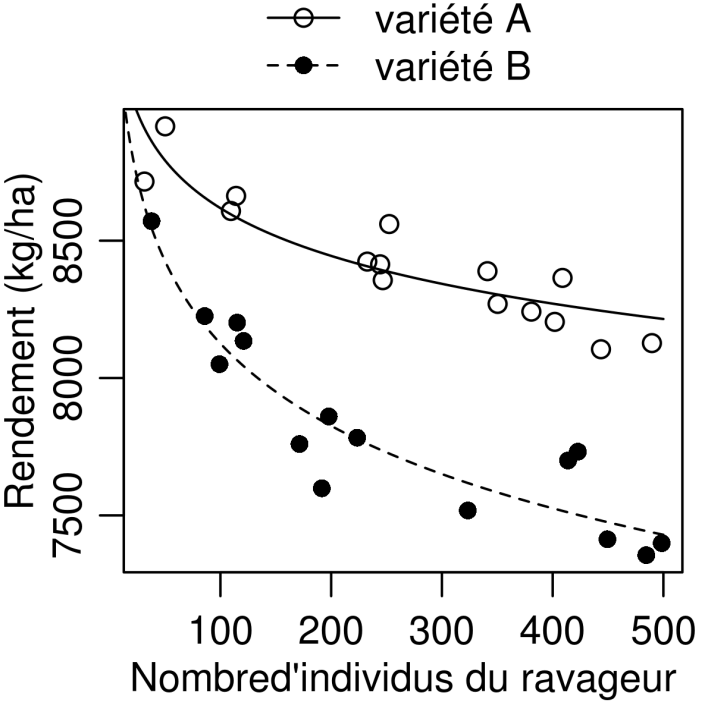
Il faut idéalement représenter les données sur l'échelle d'origine, non transformée pour faciliter l'interprétation.

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

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

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

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

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

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

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

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





Linéarité - additivité

Pistes de solutions

Avec d'autres méthodes statistiques que les GLM :

- Régressions non linéaires
- GAM : Generalized Additive Models
- Arbres de régression/classification

Linéarité - additivité

Régression non linéaire en deux mots

Surtout utiles quand on veut estimer précisément un paramètre de la régression qui a une signification biologique.

Exemple : nombre d'espèces observées au cours du temps d'échantillonnage d'un cadrat --> on veut estimer l'asymptote càd le nombre maximum d'espèce que l'on observerait si on prolongeait le temps indéfiniment.

```
d <- read.table(text='Temps Nbsp Spcaract
5 15 2
10 25 3
15 31 5
20 37 5
25 40 5', header=TRUE)
```

Linéarité - additivité

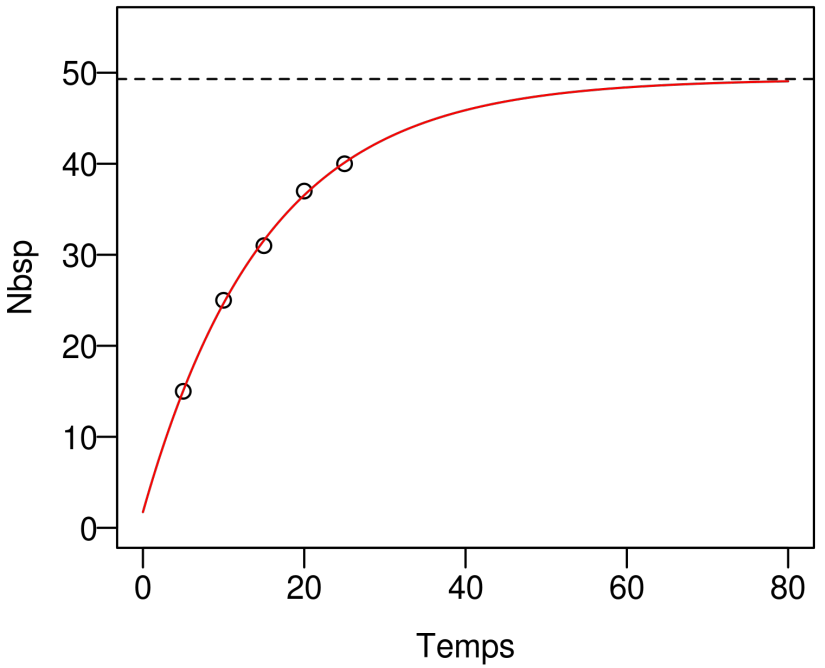
Régression non linéaire en deux mots

```
> mod <- nls(Nbsp ~ SSasymp( Temps, Asym, resp0, lrc), data = d)
> summary(mod)

Formula: Nbsp ~ SSasymp(Temps, Asym, resp0, lrc)

Parameters:
      Estimate Std. Error t value Pr(>|t|)
Asym   49.3102     3.0569  16.131  0.00382 **
resp0   1.7271     1.9125   0.903  0.46182
lrc    -2.7209     0.1602 -16.988  0.00345 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.5776 on 2 degrees of freedom

Number of iterations to convergence: 0
Achieved convergence tolerance: 2.923e-07
```

Linéarité - additivité

Régression non linéaire en deux mots

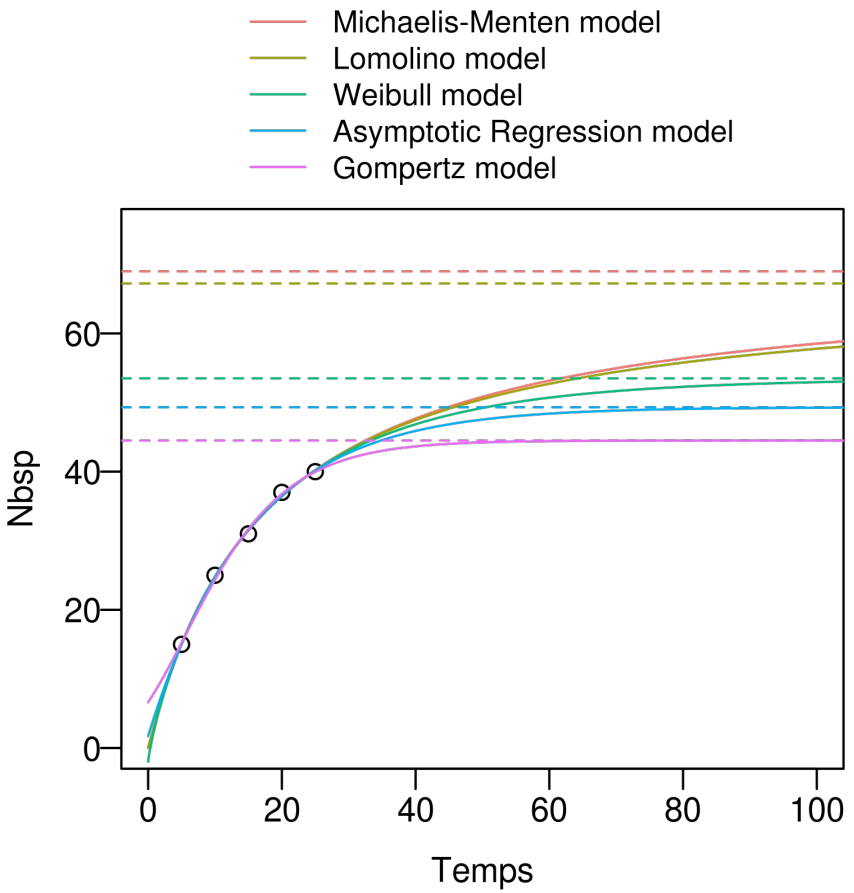
```
Asym <- coef(mod)[1]
resp0 <- coef(mod)[2]
lrc <- coef(mod)[3]

Temps <- seq(0, 80, 0.1)
pred <- Asym + (resp0-Asym) * exp(-exp(lrc) * Temps)

plot(Nbsp ~ Temps, data = d, ylim=c(0,55), xlim = c(0,80))
lines(Temps, pred, col="red")
abline(h = coef(mod)[1], lty = 2)
```



Linéarité - additivité

Régression non linéaire en deux mots

NB dans ce cas différents modèles peuvent donner une estimation de l'asymptote qui peut être très différente à cause de l'extrapolation



Graphique de la dia précédente

```
#' ## Asymptotic Regression model
Temps <- seq(0, 150, 0.1) # valeurs de temps pour lesquelles on veut une prédiction
mod <- nls(Nbsp ~ SSasymp( Temps, Asym, resp0, lrc), data = d) # modèle
Asym <- coef(mod)[1] # on récupère les coefficients du modèle
resp0 <- coef(mod)[2]
lrc <- coef(mod)[3]
pred <- Asym + (resp0-Asym) * exp(-exp(lrc) * Temps) # valeurs prédites
summary(mod) # résumé du modèle

#' ## Michaelis-Menten model
modMM <- nls(Nbsp ~ SSmicmen( Temps, Vm, K), data = d)
Vm <- coef(modMM)[1]
K <- coef(modMM)[2]
predMM <- Vm*Temps/(K+Temps)
summary(modMM)

#' ## Weibull model
modW <- nls(Nbsp ~ SSweibull( Temps, Asym, Drop, lrc, pwr), data = d)
Asym <- coef(modW)[1]
Drop <- coef(modW)[2]
lrc <- coef(modW)[3]
pwr <- coef(modW)[4]
predW <- Asym-Drop*exp(-exp(lrc)*Temps^pwr)
summary(modW)

#' ## Gompertz model
modG <- nls(Nbsp ~ SSgompertz( Temps, Asym, b2, b3), data = d)
Asym <- coef(modG)[1]
b2 <- coef(modG)[2]
b3 <- coef(modG)[3]
predG <-  Asym*exp(-b2*b3^Temps)
summary(modG)
```

Chaque modèle choisi a une forme différente mais ils contiennent tous un paramètre correspondant à l'asymptote

Graphique de la dia précédente (suite)

```
#' ## Lomolino model (dans le package Vegan)
library(vegan)
modLo <- nls(Nbsp ~ SSlomolino( Temps, Asym, xmid, slope), data = d)
Asym <- coef(modLo)[1]
xmid <- coef(modLo)[2]
slope <- coef(modLo)[3]
predLo <- Asym/(1 + slope^log(xmid/Temps))
summary(modLo)

# function to create a palette of n colors with equidistant hues
mypalette <- function(n, alpha = 1, l=65, c=100) {
    hues = seq(15, 375, length=n+1)
    hcl(h=hues, l=l, c=c, alpha = alpha)[1:n]
}
mescouleurs <- mypalette(5)

dev.new(width = 10/2.54, height = 10/2.54)
par(mar = c(3,3,5,1), cex=0.9, las=1, mgp=c(2,0.5,0))
plot(Nbsp ~ Temps, data = d, ylim=c(0,75), xlim = c(0,100))
lines(Temps, predMM, col= mescouleurs[1])
lines(Temps, predLo, col=mescouleurs[2])
lines(Temps, predW, col=mescouleurs[3])
lines(Temps, pred, col=mescouleurs[4])
lines(Temps, predG, col=mescouleurs[5])

abline(h = coef(modMM)[1], lty = 2, col = mescouleurs[1])
abline(h = coef(modLo)[1], lty = 2, col = mescouleurs[2])
abline(h = coef(modW)[1], lty = 2, col = mescouleurs[3])
abline(h = coef(mod)[1], lty = 2, col = mescouleurs[4])
abline(h = coef(modG)[1], lty = 2, col = mescouleurs[5])

legend("top", legend = c( "Michaelis-Menten model", "Lomolino model", "Weibull model",
                          "Asymptotic Regression model", "Gompertz model"),
       col =c(mescouleurs[1],mescouleurs[2], mescouleurs[3], mescouleurs[4], mescouleurs[5]),
       lty=1, bty="n", inset=-0.4, xpd=NA, cex=0.9)
```

Linéarité - additivité

Generalized Additive Models (GAMs) en deux mots

Méthode permettant d'incorporer des courbes de lissage dans des modèles linéaires (généralisés mixtes)

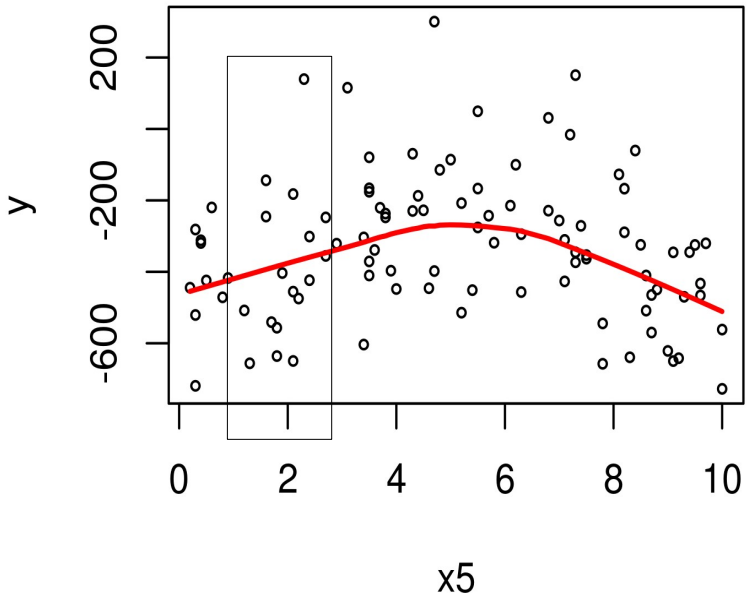
```
set.seed(5)
x1 <- runif(100, 0, 10)
x2 <- runif(100, 0, 10)
x3 <- runif(100, 0, 10)
x4 <- runif(100, 0, 10)
x5 <- runif(100, 0, 10)
x6 <- runif(100, 0, 10)

set.seed(1)
y <- 100 * log(x1) + 0.025* exp(x2) - 300*x3^0.25 + 2 * x4^2 + (x5-5) -
    15*(x5-5)^2 - 10 *x6 + rnorm(100, 0, 50)
d <- round(data.frame(y, x1, x2, x3, x4, x5, x6),1)
```

génération d'un jeu de données avec des relations non linéaires

Linéarité - additivité

Generalized Additive Models en deux mots

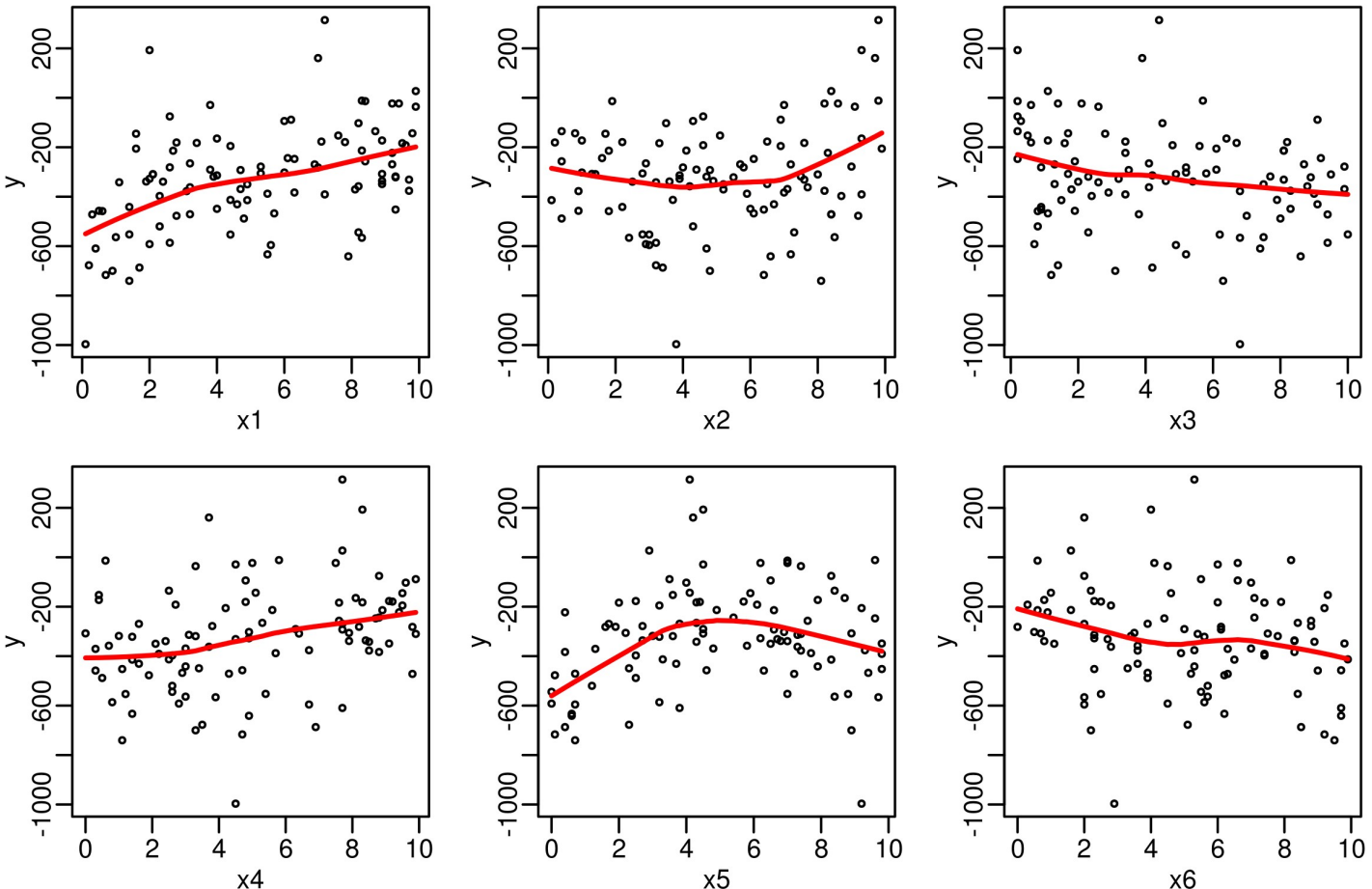
Courbe de lissage ("smoother")



"Fenêtre" autour de la valeur 2 de x5
Pour obtenir une valeur prédite de y, on peut avec les points qui se trouvent dans la fenêtre calculer :
- leur moyenne (mooving average)
- moyenne pondérée négativement quand x s'éloigne de 2 (loess)
- une régression polynomiale (splines)
- une régression polynomiale pondérée (lowess)

--> plus la "fenêtre" est large, plus la courbe est lissée

```
par(mfrow = c(2,3), mar = c(2.5,2.5,1,1), mgp = c(1.5, 0.5,0))
plot(y ~ x1, data=d, cex=0.5)
lines(lowess(d$x1,d$y),col="red", lwd=2)
plot(y ~ x2, data=d, cex=0.5)
lines(lowess(d$x2,d$y),col="red", lwd=2)
plot(y ~ x3, data=d, cex=0.5)
lines(lowess(d$x3,d$y),col="red", lwd=2)
plot(y ~ x4, data=d, cex=0.5)
lines(lowess(d$x4,d$y),col="red", lwd=2)
plot(y ~ x5, data=d, cex=0.5)
lines(lowess(d$x5,d$y),col="red", lwd=2)
plot(y ~ x6, data=d, cex=0.5)
lines(lowess(d$x6,d$y),col="red", lwd=2)
```



Linéarité - additivité

Generalized Additive Models en deux mots

```
> library(mgcv)
> mod <- gam(y ~  s(x1) + s(x2) + s(x3) + s(x4) + s(x5) + s(x6), data=d)
> summary(mod)

Family: gaussian
Link function: identity

Formula:
y ~ s(x1) + s(x2) + s(x3) + s(x4) + s(x5) + s(x6)

Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -324.879      4.517  -71.92   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Approximate significance of smooth terms:
        edf Ref.df     F p-value
s(x1) 7.318  8.276 43.50 < 2e-16 ***
s(x2) 7.391  8.323 52.16 < 2e-16 ***
s(x3) 3.662  4.510 64.67 < 2e-16 ***
s(x4) 4.162  5.073 35.96 < 2e-16 ***
s(x5) 4.165  5.073 82.08 < 2e-16 ***
s(x6) 1.000  1.000 10.01 0.00227 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

R-sq.(adj) =  0.954   Deviance explained = 96.7%
GCV = 2861.7  Scale est. = 2040.5    n = 100
```

Variables avec lissage.
On peut aussi avoir des variables "normales"

edf = estimated degrees of freedom estimés par cross-validation
si edf = 1 : ligne droite

Méthode "non paramétrique" : pas de coefficients pour les variables explicatives avec courbe de lissage s()

Linéarité - additivité

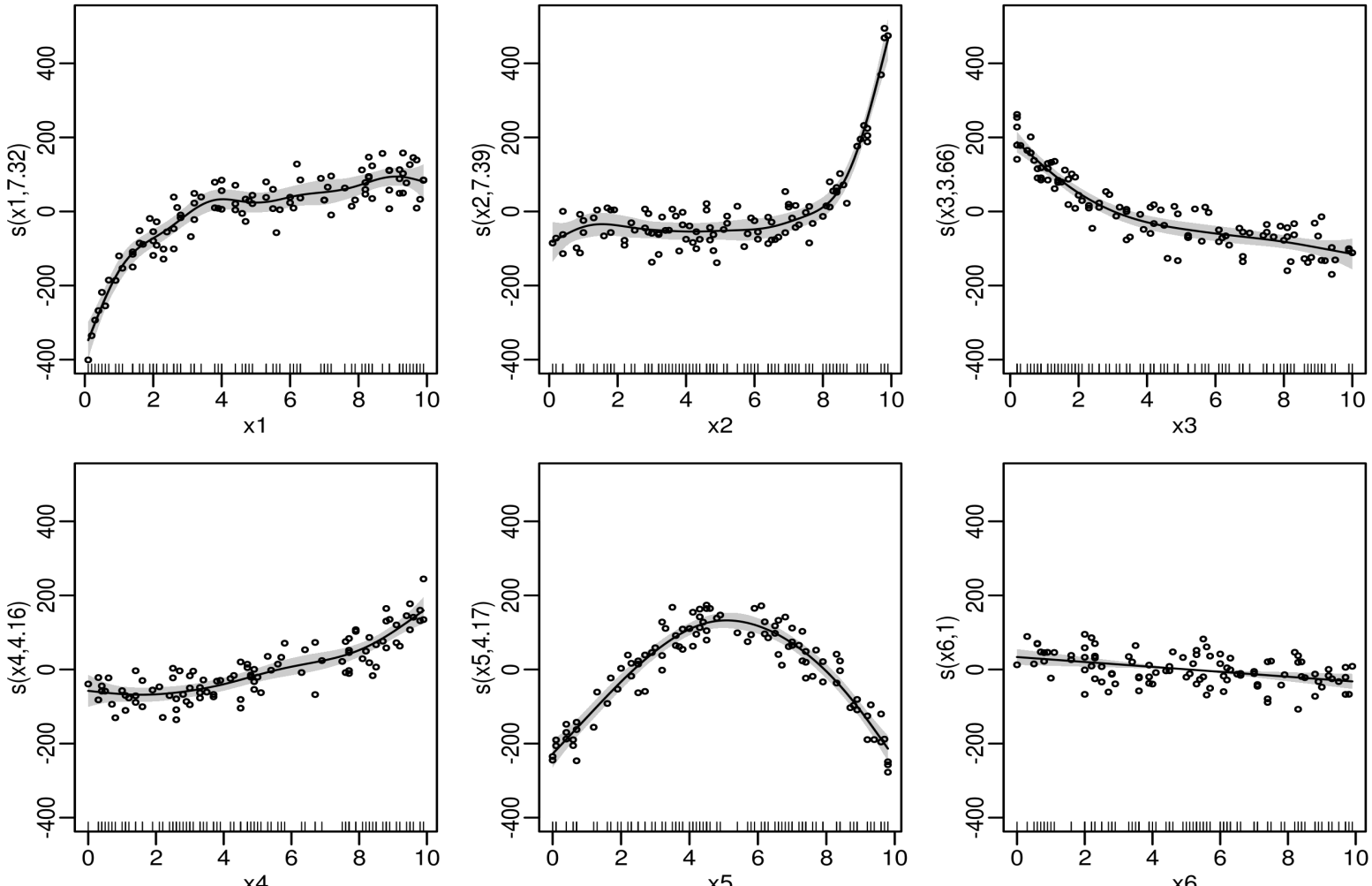
Generalized Additive Models en deux mots

```
par(mfrow = c(2,3), mar = c(2.5,2.5,1,1), mgp = c(1.5, 0.5,0))
plot(mod, residuals = TRUE, pch=1, cex=0.5, shade=TRUE)
```

Permet de visualiser rapidement les relations non linéaires et leur forme après avoir enlevé l'effet des autres variables qui pourraient masquer certaines relations

NB :
Prédictions marginales "à l'intercept" i.e. quand tous les autres x sont = 0

-->
Il peut être utile de centrer les x



Linéarité - additivité

Generalized Additive Models en deux mots

Avantages :

Détection automatique des relation non linéaires

Désavantages :

Non paramétrique :
on ne peut pas résumer le modèle par une simple équation
black-box

Pas d'interactions
(entre les variables lissées)

Tendance à l' overfitting

Linéarité - additivité

Arbres de régression/classification en deux mots

Approche algorithmique complètement différente des GLM.

Permettent de prédire y en la découpant récursivement sur base des variables x.

Le but est de former des groupes de données qui minimisent la variance intra-groupe et maximisent la variance extra-groupe.

Ne nécessite pas de définir de forme précise modèle de départ (pas de problème de non linéarité, pas d'interaction, pas de problèmes de points extrêmes dans les X...).

Le modèle est insensible aux transformations des X
Par contre il est sensible aux transformations de Y
(sur laquelle est calculée la variance)

Arbres de régression/classification en deux mots

```
> library(rpart)
> tree <- rpart(y ~ x1 + x2 + x3 + x4 + x5 + x6, data=d)

> cp <- tree$cptable
> cp <- cp[which(cp[,"xerror"] < cp[nrow(cp), "xerror"] + cp[nrow(cp), "xstd"])[1], "CP"]
> ptree <- prune(tree, cp= cp)

> print(ptree)
n= 100

node), split, n, deviance, yval
      * denotes terminal node

 1) root 100 4430030.0 -324.8790
   2) x1< 1.5 13  358599.7 -594.3385 *
   3) x1>=1.5 87 2986477.0 -284.6149
     6) x5< 1.25 11  177947.6 -524.2182 *
     7) x5>=1.25 76 2085620.0 -249.9355
      14) x2< 9.05 68 1219542.0 -277.2838
        28) x4< 4.2 29  487392.1 -350.2621 *
        29) x4>=4.2 39  462854.4 -223.0179 *
      15) x2>=9.05 8  382915.8  -17.4750 *
```

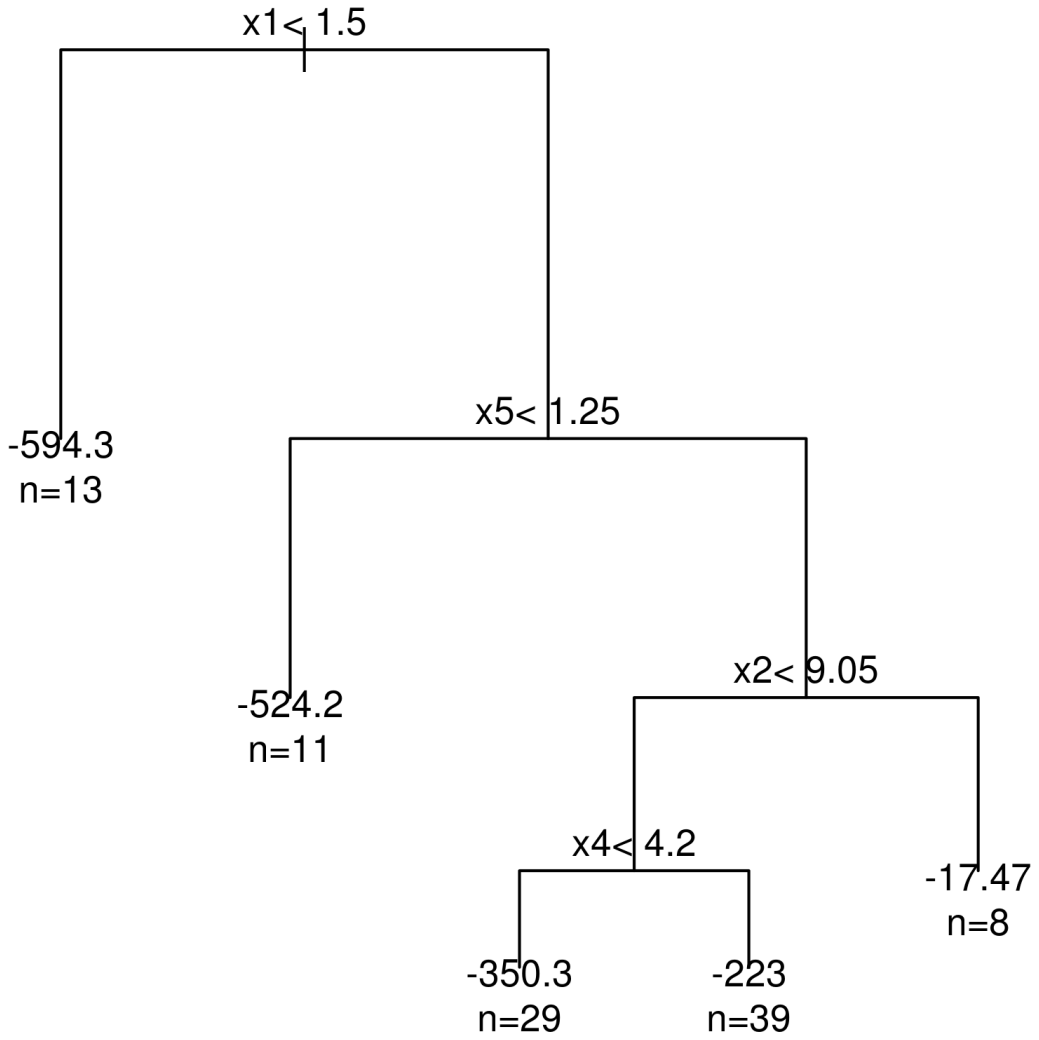
On "élague" (prunning) l'arbre pour limiter l'overfitting en se basant sur une méthode de crossvalidation (one standard error rule) et un coefficient de complexité (cp)

Linéarité - additivité

Arbres de régression/classification en deux mots

Représentation graphique basique

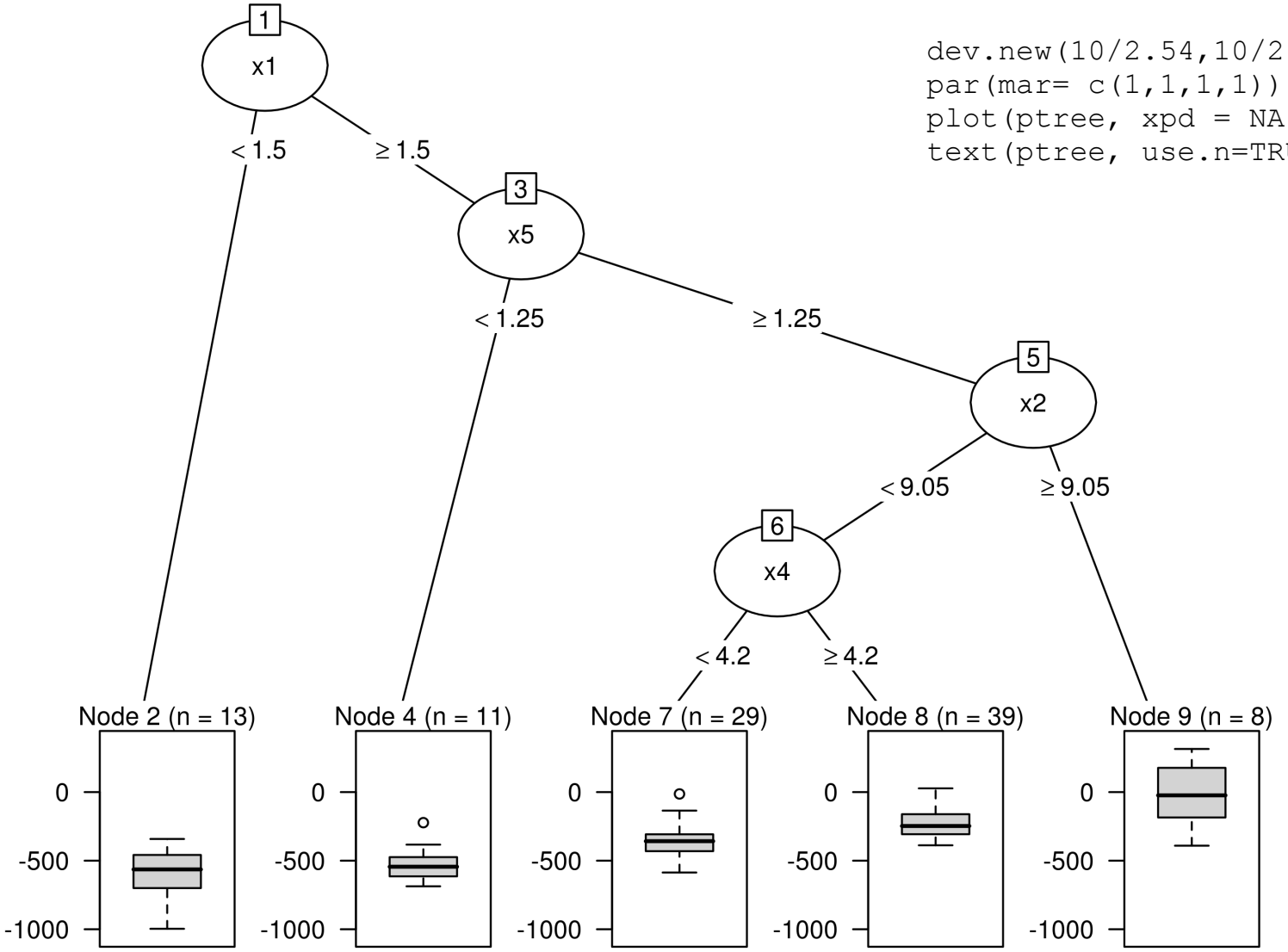
```
dev.new(10/2.54,10/2.54)
par(mar= c(1,1,1,1))
plot(ptree, xpd = NA)
text(ptree, use.n=TRUE, cex = 0.8, xpd
= NA)
```



Linéarité - additivité

Arbres de régression/classification en deux mots

Représentation graphique plus élaborée

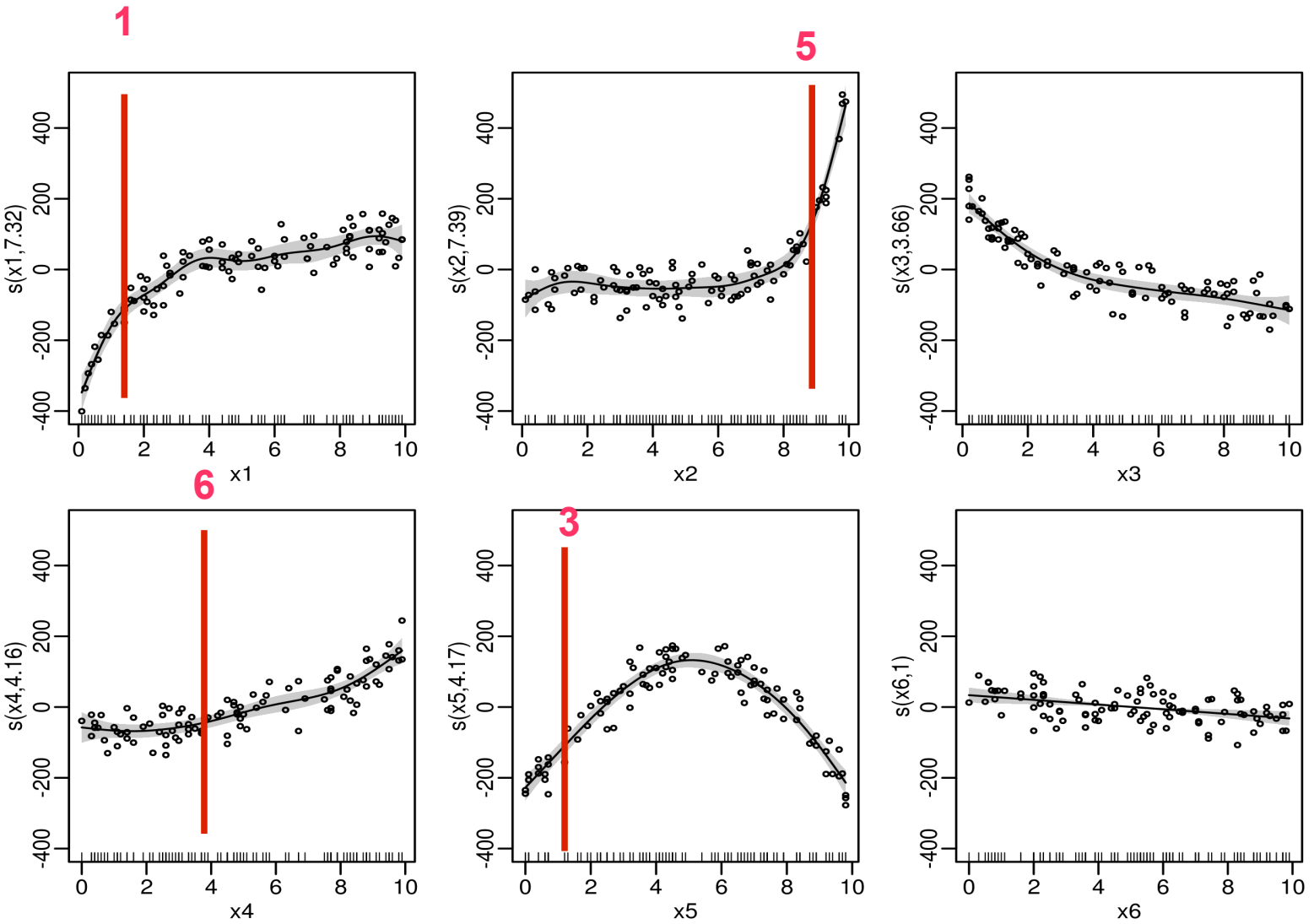



```
dev.new(10/2.54,10/2.54)
par(mar= c(1,1,1,1))
plot(ptree, xpd = NA)
text(ptree, use.n=TRUE, cex = 0.8, xpd = NA)
```

Linéarité - additivité

Arbres de régression/classification en deux mots

GAM pour comparaison



Linéarité - additivité

Arbres de régression/classification en deux mots

Avantages :

Très faciles à interpréter
Pas besoin de spécifier la structure du modèle
Insensible aux valeurs extrêmes en x

Désavantages :

Modèle très simpliste avec prédictions approximatives :
ici 5 valeurs prédites possibles pour tout le jeu de données

Très sensible à de légères modifications du jeu de données

Linéarité - additivité

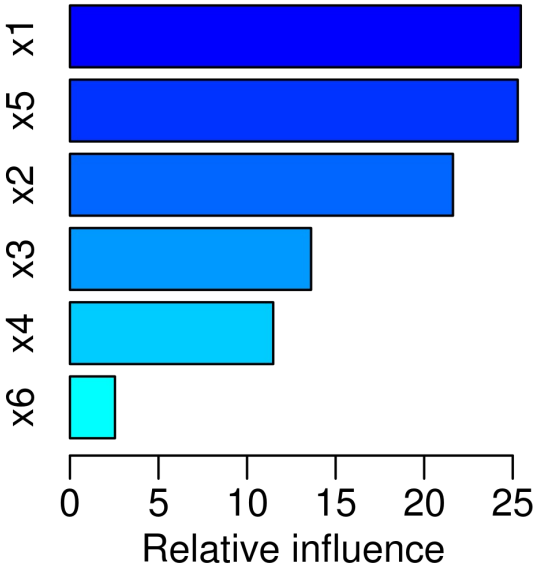
Arbres de régression/classification en deux mots

Nombreux algorithmes plus élaborés pour éviter ces problèmes au prix de l'interprétabilité (bagging, random forests, boosting,...)

Ex : Boosted Regression Tree (BRT)

Paramètres optimaux déterminés par cross-validation

```
> library(gbm)
> brt <- gbm(y ~ x1 + x2 + x3 + x4 + x5 + x6, data=d,
+            n.trees = 7000, n.minobsinnode = 3,
+            interaction.depth = 1, shrinkage = 0.0075, bag.fraction = 0.5)
Distribution not specified, assuming gaussian ...
> summary(brt)
   var   rel.inf
x1  x1 25.456656
x5  x5 25.283247
x2  x2 21.625935
x3  x3 13.615106
x4  x4 11.477168
x6  x6  2.541888
```



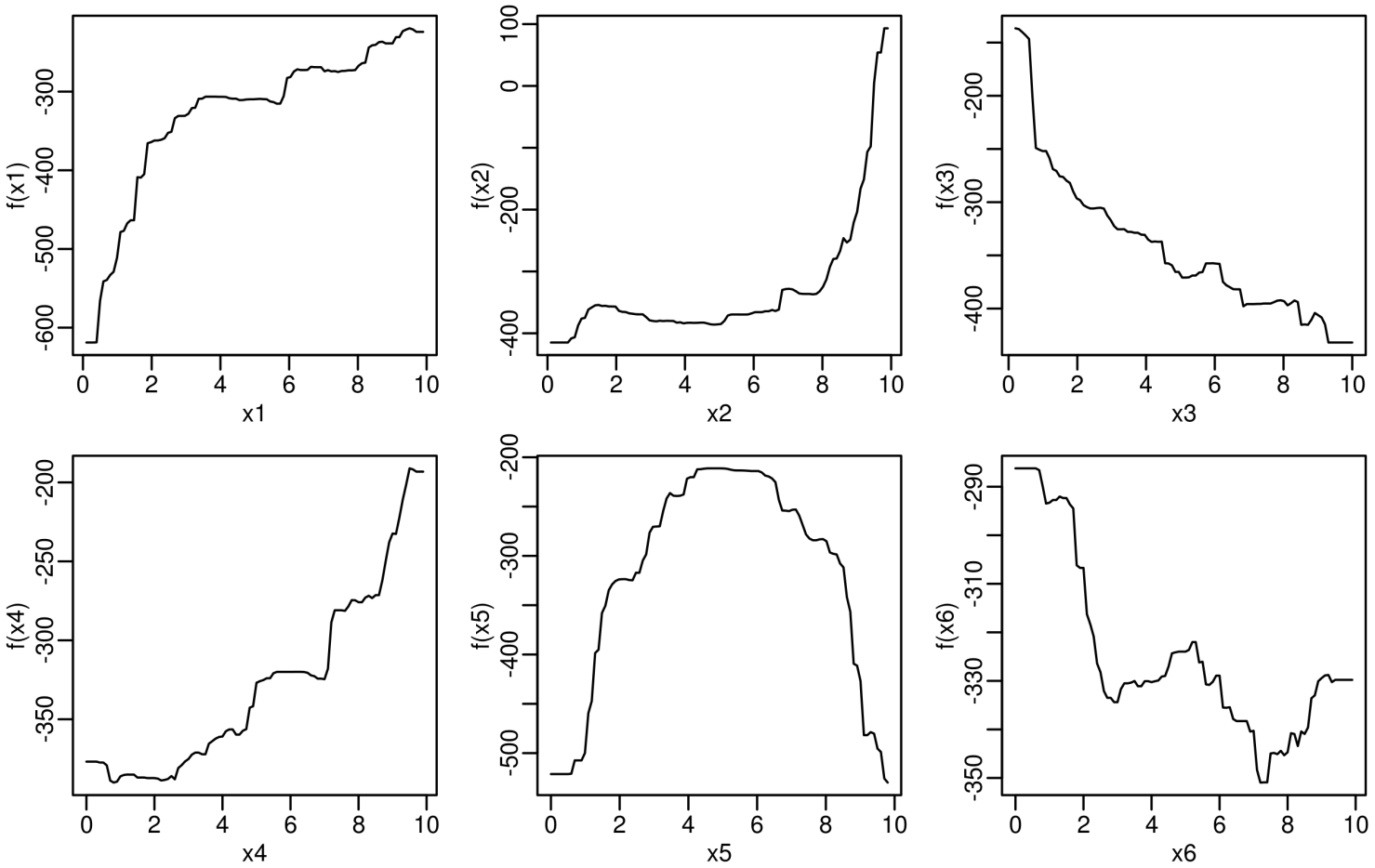
Donne une mesure de l'importance relative des différentes variables...

Linéarité - additivité

Arbres de régression/classification en deux mots

Ex : Boosted Regression Tree (BRT)

```
dev.new(18/2.54, 12/2.54)
par(mfrow = c(2,3), mar = c(2.5,2.5,1,1), mgp = c(1.5, 0.5,0))
plot(brt, i = 1)
plot(brt, i = 2)
plot(brt, i = 3)
plot(brt, i = 4)
plot(brt, i = 5)
plot(brt, i = 6)
```



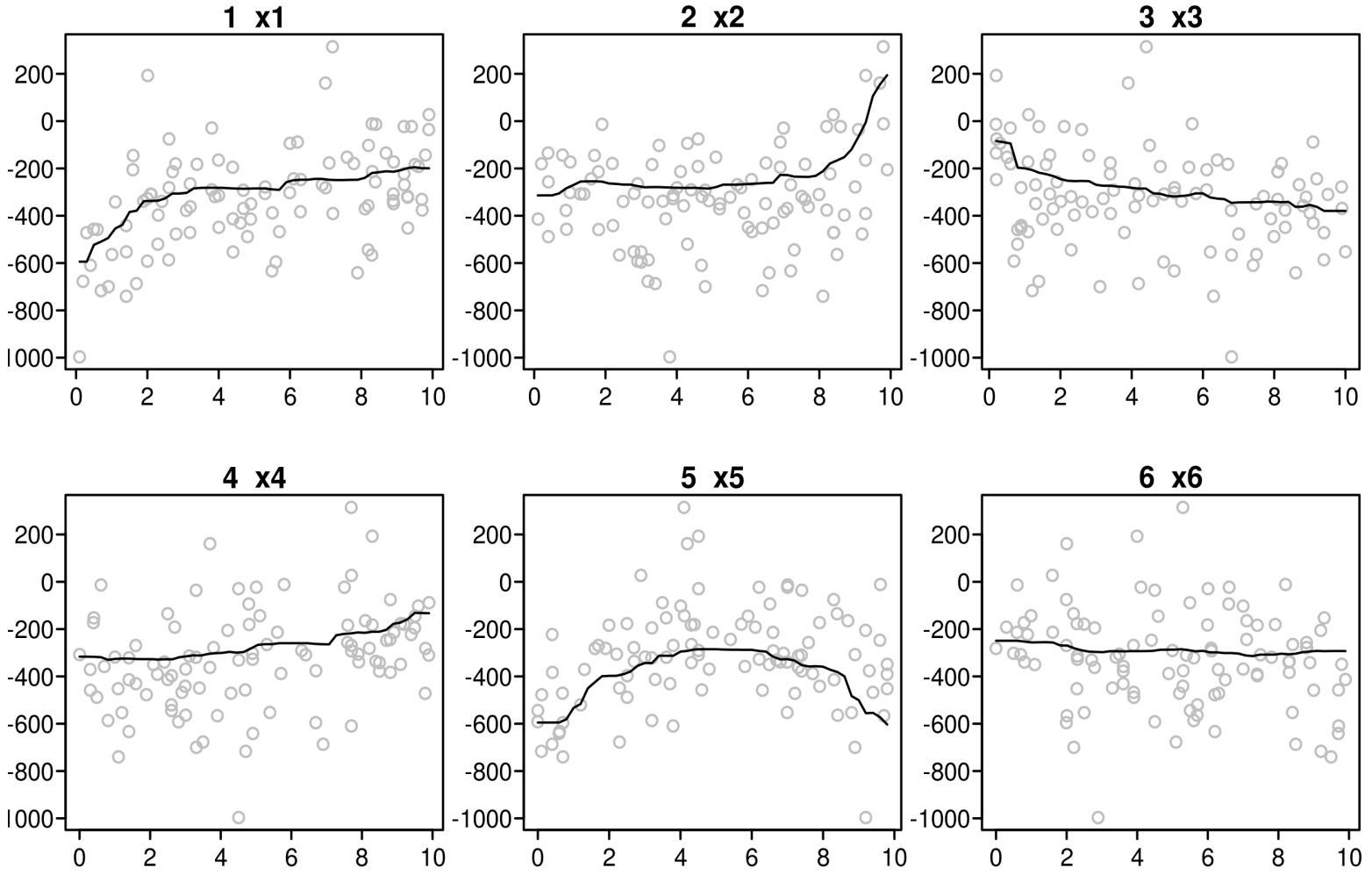
Linéarité - additivité

Arbres de régression/classification en deux mots

Ex : Boosted Regression Tree (BRT)

```
dev.new(width = 16/2.54, height = 8/2.54)
library(plotmo)
plotmo(brt, degree2 = FALSE, trace = -1, caption = "", mfrow = c(2,3), las = 1,
 pt.col= "grey75", pt.pch = 1)
```

Prédiction pour toutes les autres x fixés à leur valeur moyenne sur une même échelle + valeurs observées

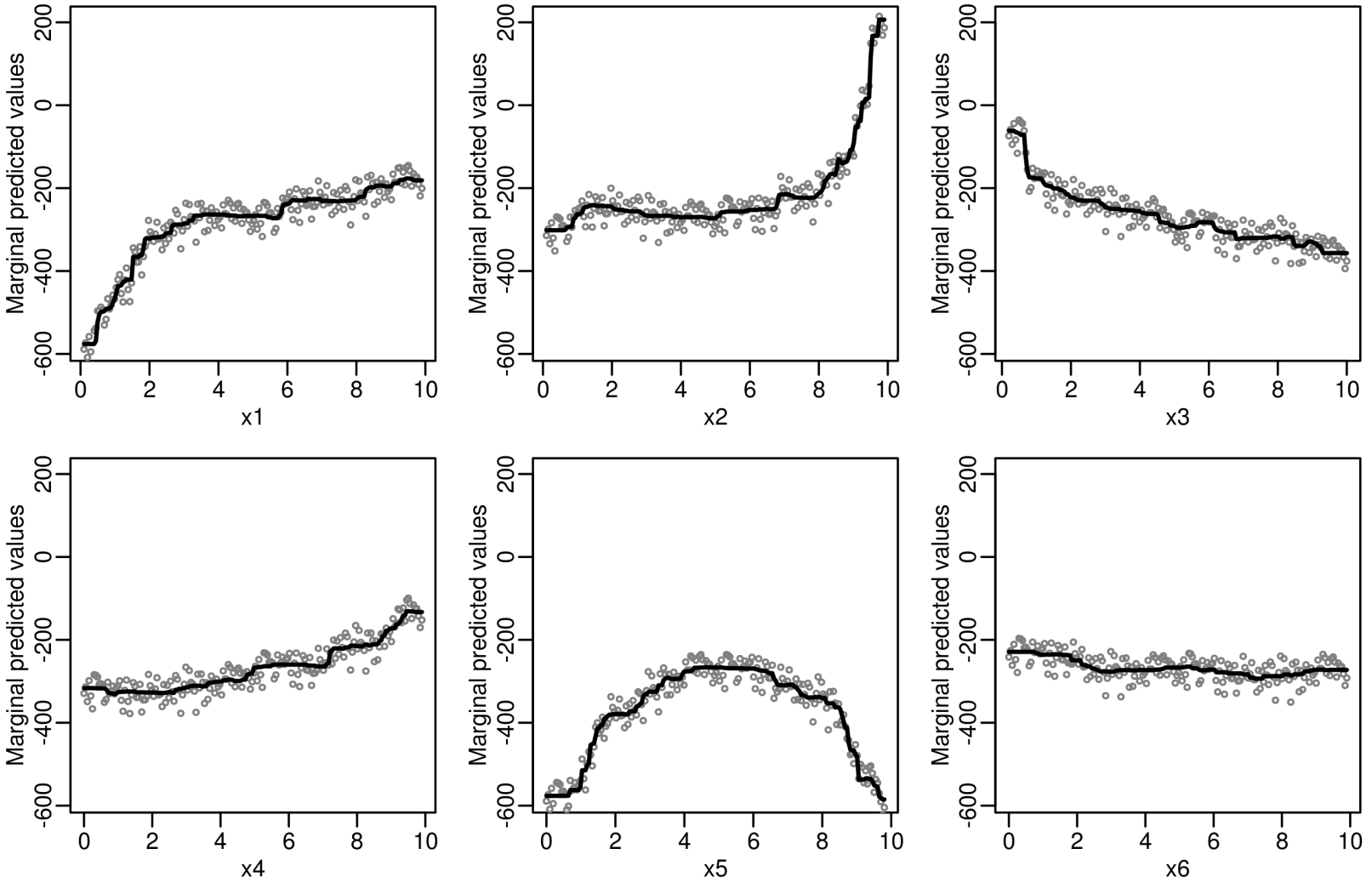


Linéarité - additivité

Arbres de régression/classification en deux mots

Ex : Boosted Regression Tree (BRT)

Prédiction pour toutes les autres x fixés à leur valeur moyenne sur une même échelle + "partial residuals"

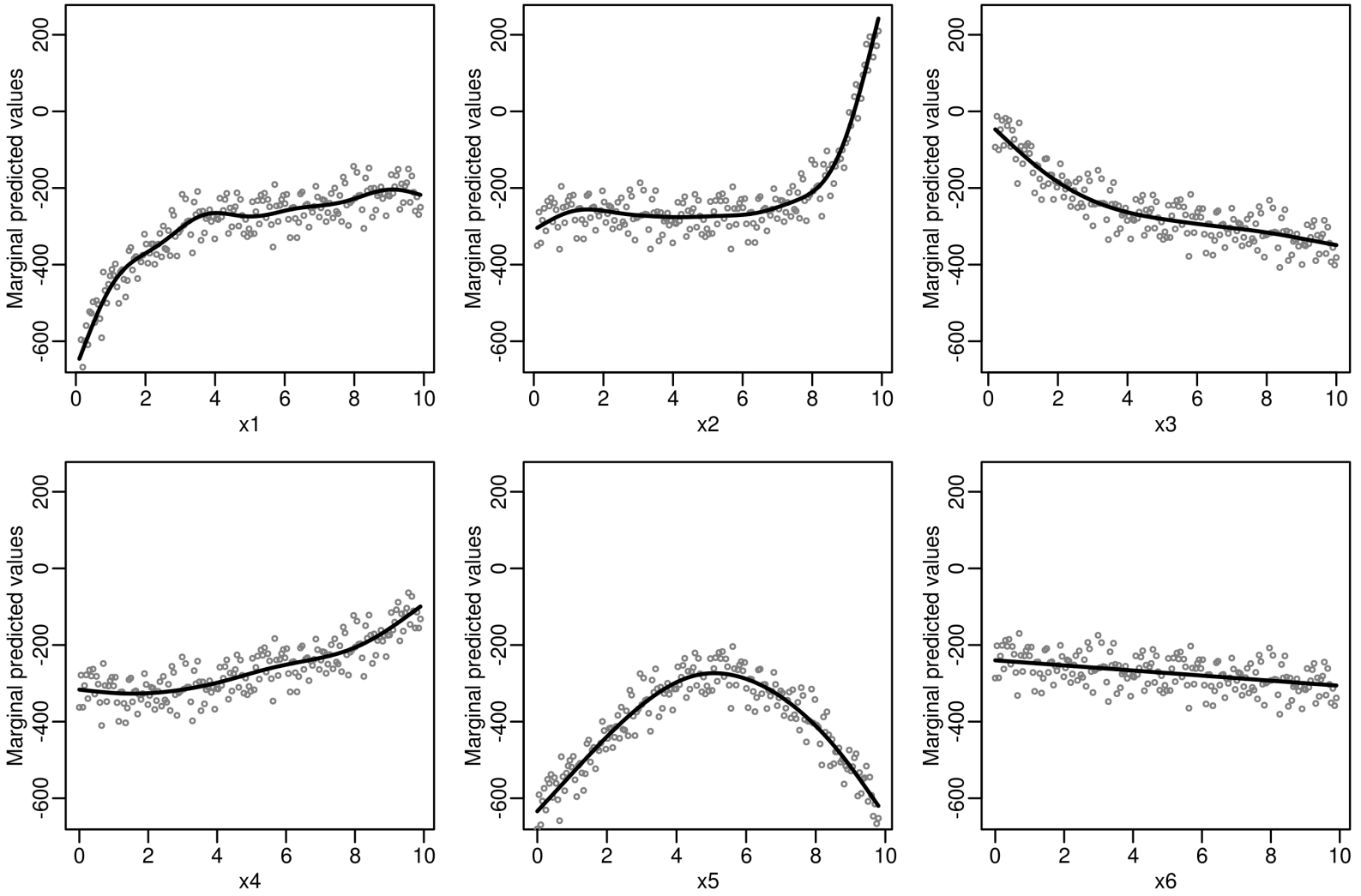


Linéarité - additivité

Arbres de régression/classification en deux mots

Ex : Graphiques similaires avec GAM pour comparaison

Prédiction pour toutes les autres x fixés à leur valeur moyenne sur une même échelle + "partial residuals"



Graphiques des deux slides précédents :

```
# Graphiques BRT

# Calcul des résidus
brt_pred <- predict(brt, newdata= x,  n.trees = 7000, n.minobsinnode = 3,
                    interaction.depth = 1, shrinkage = 0.0075, bag.fraction = 0.5)
residbrt <- d[,1] - brt_pred

n <- 200
x <- d[,-1]
pr_seq <- apply(x, 2, function(x) seq(from = min(x), to= max(x),
                                      length.out = n))
pr_mean <- apply(x, 2, function(x) rep(mean(x), n))
newdata <- vector(mode = "list", length = ncol(pr_seq))
pred <- as.data.frame(matrix(nrow = n, ncol = ncol(x)))
names(newdata) <- colnames(pred) <- colnames(x)

for(i in 1:length(newdata)) {
    pr <- pr_mean
    pr[,i] <- pr_seq[,i]
    newdata[[i]] <- as.data.frame(pr)
    pred[,i] <- predict(brt, newdata=newdata[[i]],  n.trees = 7000, n.minobsinnode = 3,
                        interaction.depth = 1, shrinkage = 0.0075, bag.fraction = 0.5)
}

lim <- c(min(pred), max(pred))

dev.new(18/2.54, 12/2.54)
par(mfrow = c(2,3), mar = c(2.5,2.5,1,1), mgp = c(1.5, 0.5,0))
for(i in 1:length(newdata)) {
    plot(pred[,i] ~ newdata[[i]][,i], type ="n", ylim = lim,
         xlab = colnames(pred)[i], ylab = "Marginal predicted values")
    points(y = pred[,i] + residbrt, x = newdata[[i]][,i],
           pch = 1, cex = 0.5, col = "grey50")
    lines(pred[,i] ~ newdata[[i]][,i], lwd = 2)
}
```

Graphiques des deux slides précédents :

```
# Graphiques GAM


for(i in 1:length(newdata)) {
    pr <- pr_mean
    pr[,i] <- pr_seq[,i]
    newdata[[i]] <- as.data.frame(pr)
    pred[,i] <- predict(mod, newdata=newdata[[i]])
}

lim <- c(min(pred), max(pred))

dev.new(18/2.54, 12/2.54)
par(mfrow = c(2,3), mar = c(2.5,2.5,1,1), mgp = c(1.5, 0.5,0))
for(i in 1:length(newdata)) {
    plot(pred[,i] ~ newdata[[i]][,i], type ="n", ylim = lim,
         xlab = colnames(pred)[i], ylab = "Marginal predicted values")
    points(y = pred[,i] + resid(mod), x = newdata[[i]][,i],
           pch = 1, cex = 0.5, col = "grey50")
    lines(pred[,i] ~ newdata[[i]][,i], lwd = 2)

}
```

Hypothèses sur la variance des résidus