Hypothèses sur la variance des résidus

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,... ?

Hypothèses sur la variance des résidus

Le problème est différent pour les modèles gaussiens d'une part et les autres GLMs d'autre part et sera traité à part

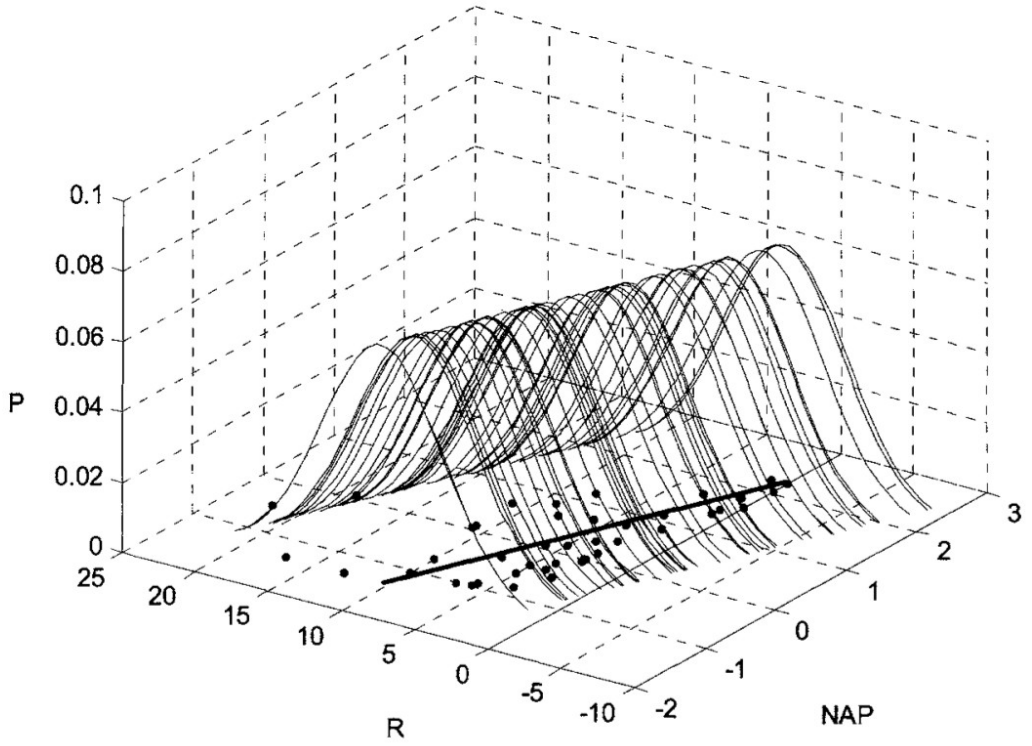
Pour les modèles Gaussiens : **Homogénéité des variances**
Pour chaque valeur de chaque variable explicative, la distribution des résidus devrait approximativement suivre une distribution normale de variance constante

Pour tous les autres GLM : Surdispersion
(y compris Negative Binomial etc...)
La variance des résidus doit suivre une fonction de la moyenne. Lorsque la variation est plus grande que prévue, il y a "surdispersion" ce qui provoque une sous estimation des erreurs standard et une sur -estimation de la vraisemblance

Modèles Gaussiens : Homogénéité des variances

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

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



Modèles Gaussiens : Homogénéité des variances

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

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

Comme pour la normalité, on ne peut en général pas vérifier directement ce présupposé car on ne dispose en général pas de suffisamment de répétitions pour chaque valeur de x.

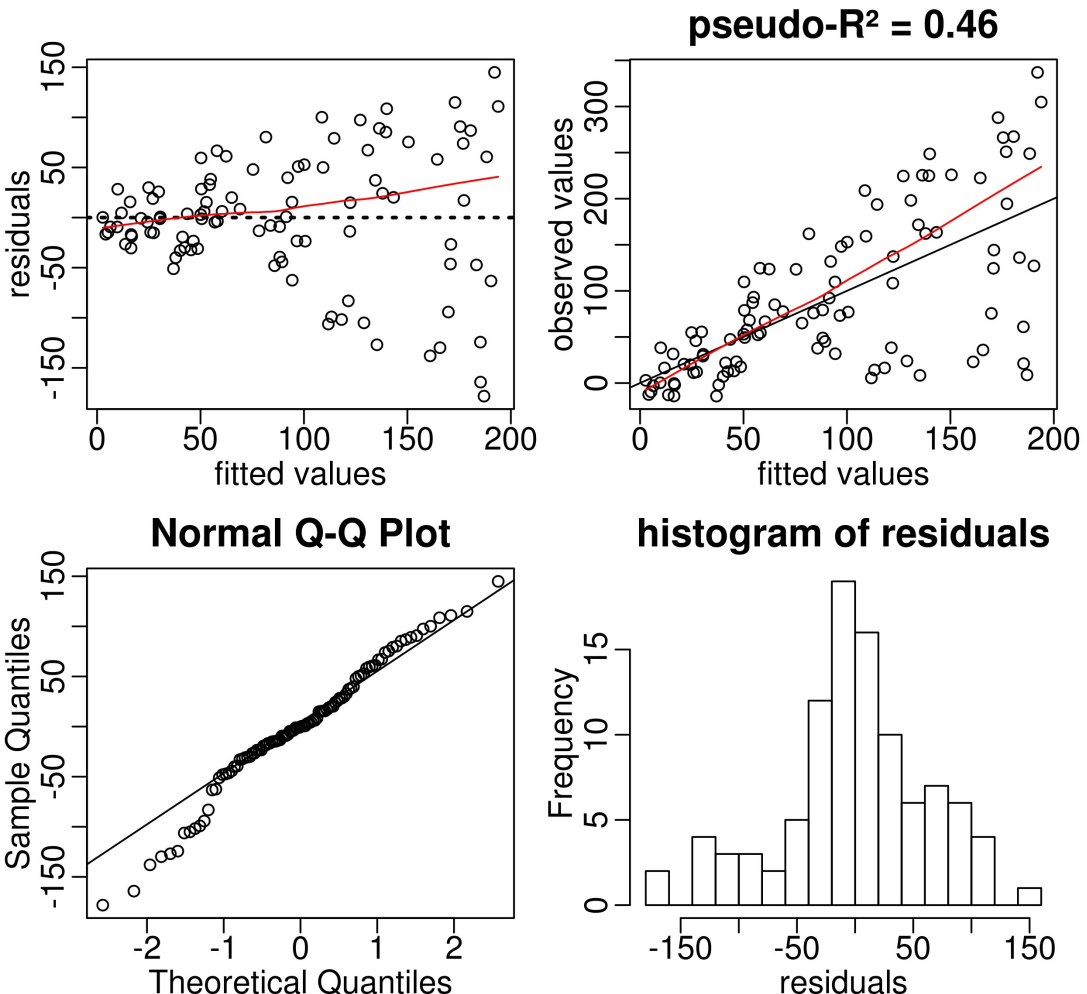
En général on trace un graphique des résidus en fonction des valeurs prédites et/ou de chaque variable explicative.
Il faut essayer d'éviter les cas où la variance résiduelle augmente avec la moyenne (forme triangulaire du nuage de points)

Les fonctions `diagplot()` et `diagplot2()` dans mytoolbox.R

Modèles Gaussiens : Homogénéité des variances

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

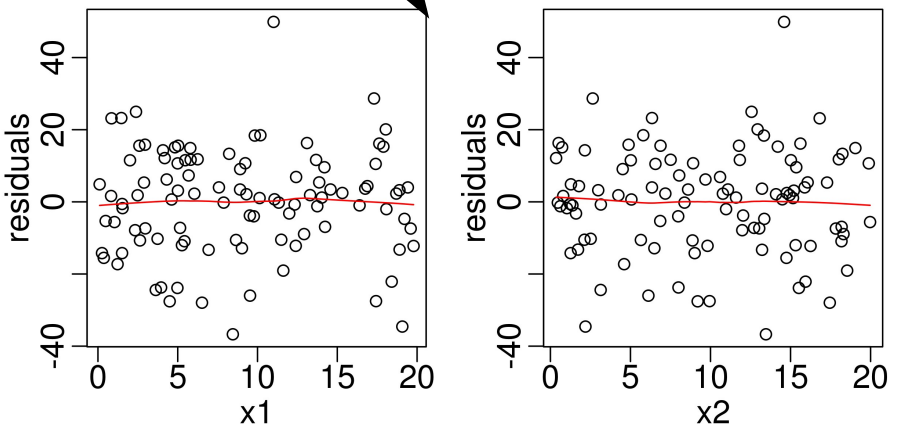
```
n <- 100
x1 <- runif(n, 0, 20)
x2 <- runif(n, 0, 20)
y <- x1 * x2 +  rnorm(n, 0, 15)

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

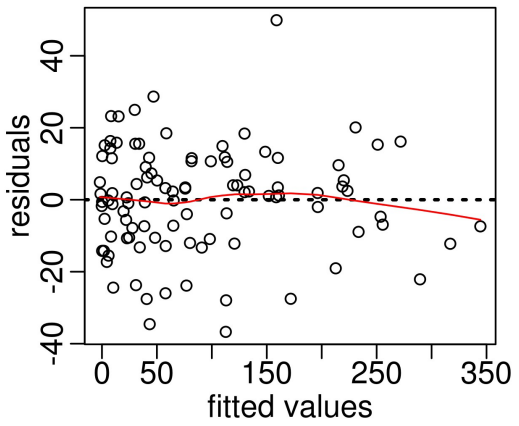


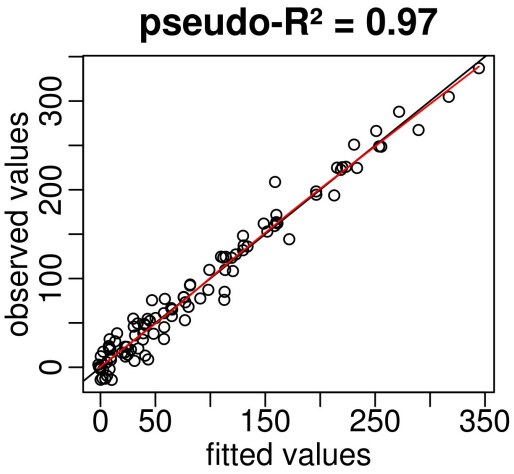
Modèles Gaussiens : Homogénéité des variances

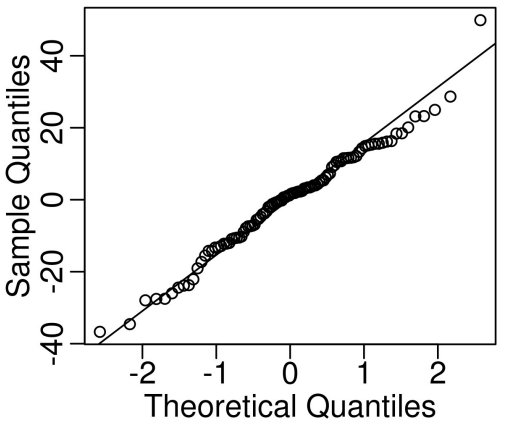
Dans ce cas précis, le problème est totalement résolu en ajoutant une deuxième variable explicative et l'interaction

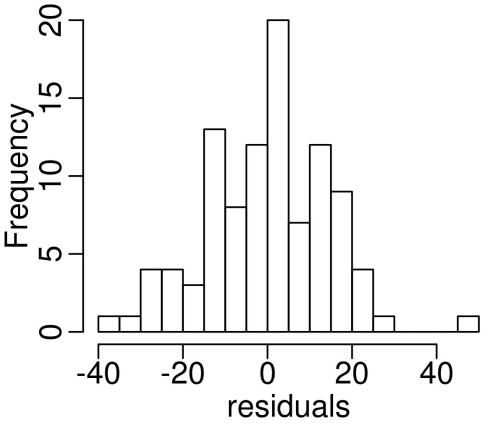
```
mod <- lm(y ~ x1*x2 )
diagplot(mod)
diagplot2(mod)
```











Modèles Gaussiens : Homogénéité des variances

Hétérogénéité des variances : solutions ?

- utiliser une distribution plus appropriée qui suppose une relation différente entre la moyenne et la variance (pex Poisson, Gamma)

- ajouter des interactions ou certaines variables explicatives importantes

- transformer y
(typiquement log pour donner moins d'importance aux valeurs extrêmes ou racine carrée pour un effet moins fort)

- modifier la matrice de variance-covariance des résidus pour estimer plusieurs variances résiduelles au lieu d'une seule
pex via package `nlme (gls)`

Surdispersion

Une caractéristique importante à vérifier dans les GLM est de savoir si la relation entre la moyenne et la variance respecte bien celle du modèle choisi (Poisson ou Binomial)

Pour la loi de Poisson :

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

Pour la loi Binomiale :

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

Le paramètre φ est appelé "coefficient de surdispersion".
Lorsqu'il est > 1, on dit qu'il y a surdispersion

Surdispersion

La surdispersion a des conséquences sur l'inférence

1) il faut utiliser les erreurs standard multipliées par la racine carrée du coefficient de surdispersion
(pour les intervalles de confiance, tests de Wald etc...)

2) il faut utiliser la log-vraissemblance divisée par le coefficient de surdispersion
pour les comparaisons de modèles (avec une distribution de F au lieu de Chi²), pour les AIC (=QAIC)

En pratique donc les inférences ne sont pas assez prudentes
(les p valeurs, intervalles de confiance etc... sont trop petits).

Par contre les coefficients sont non biaisés.

Surdispersion

Diagnostique

Il faut calculer systématiquement le coefficient de surdispersion.
Il y a deux méthodes principales

Le rapport entre la déviance résiduelle et le nombre de degrés de liberté ("residual df") données par le summary donne une idée approximative de la surdispersion.

```
> mod <- glm(prop ~ sex + dose, weights = n, data=d, family = binomial)
> mod2 <- glm(cbind(dead, n-dead) ~ sex + dose, data=d, family = binomial)
> summary(mod)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -6.6042     1.0208  -6.469 9.83e-11 ***
sexM          3.5689     0.7931   4.500 6.81e-06 ***
dose          0.8682     0.1236   7.025 2.15e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 236.7523  on 11  degrees of freedom
Residual deviance:   6.7239  on  9  degrees of freedom
AIC: 28.658
```

Surdispersion :
6.7239/9 = 0.7471

Surdispersion

Diagnostique

On obtient une estimation plus précise sur base des "Pearson residuals"
Les Pearson residuals sont des résidus standardisés par l'écart type théorique des valeurs prédites.

$$z_i = \frac{y_i - \hat{y}_i}{sd\left(\hat{y}_i\right)}$$

$$overdispersion = \frac{1}{n-k}\sum_{i=1}^{n} z_i^2$$

NB : Pour les plots de résidus, on utilise les pearson residuals qui doivent donc avoir une variance homogène comme un graphique de résidu classique/
Ces résidus sont fournis par `residuals(mymodel, type : "pearson")`

Surdispersion

Diagnostique

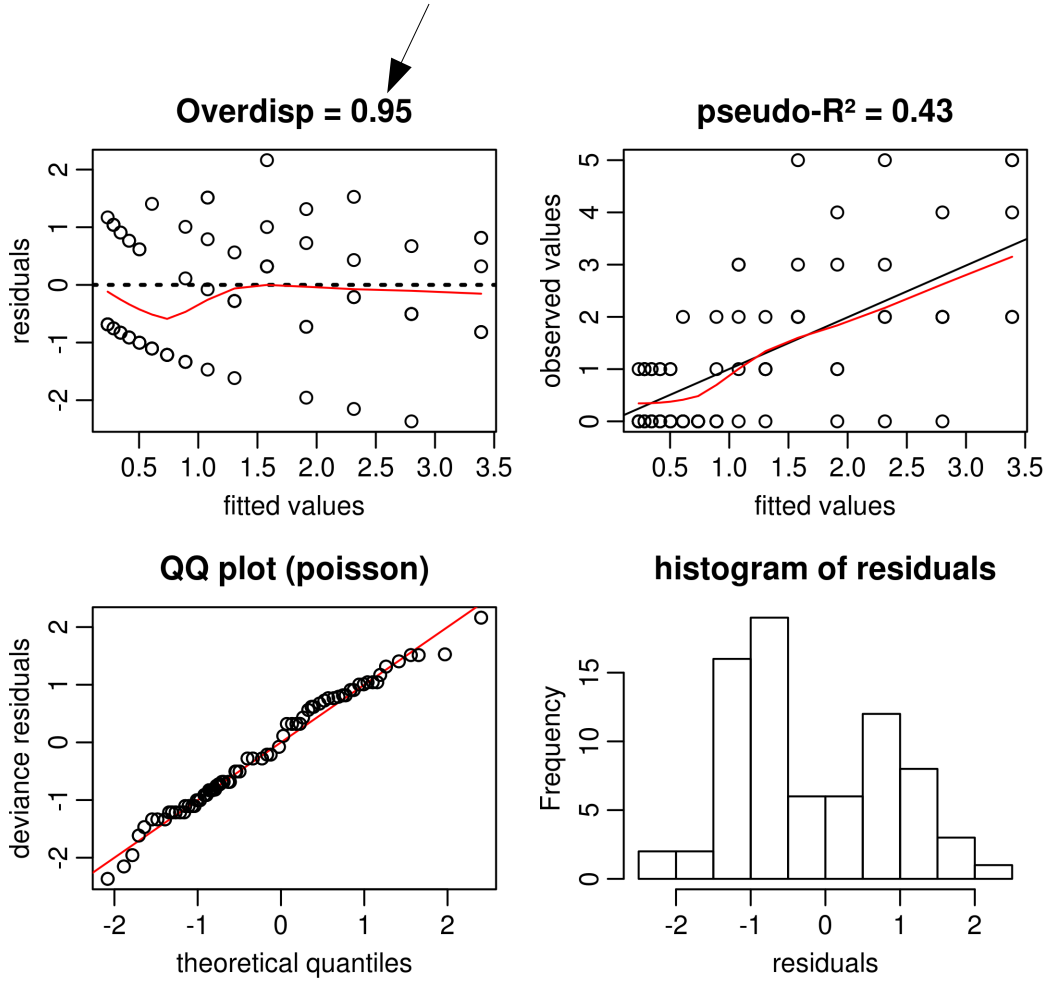
La fonction `overdisp` (`mytoolbox.R`) d'estimer la surdispersion sur base de la déviance et sur base des "Pearson residuals"

```
overdisp <- function(mod) {

    k <- attr(logLik(mod),"df")
    n <- length(fitted(mod))

    pearsonresid <- (1/(n-k)) * sum(resid(mod,"pearson")^2)
    dev <- deviance(mod)/(n-k)

    result <- c(pearsonresid,dev)
    names(result) <- c("pearsonresid","deviance")
    return(result)
    }
```

Surdispersion

Diagnostique

NB : la fonction diagplot (mytoolbox.R) vous donne le coefficient de surdispersion (sur base des résidus de Pearson)



Surdispersion

Surdispersion : solutions ?

En général il est conseillé dans un premier temps d'améliorer le modèle pour limiter au maximum la surdispersion

- ajouter des interactions ou certaines variables explicatives importantes

- linéariser la relation en transformant les x ou avec une fonction de lien différente

- ajouter des facteurs aléatoires permettant de prendre en compte la non indépendance

- "traiter" les valeurs extrêmes

- essayer une autre fonction de lien

- si des données de comptage ne suivent de manière évidente pas une distribution de Poisson, utiliser une distribution gaussienne après transformation des y (log en général)

Surdispersion

Surdispersion : solutions ?

Une source fréquente de surdispersion est un **excès de 0** en particulier pour les distributions de Poisson.

En général cela survient quand on mesure deux phénomènes conjoints : la probabilité de présence puis si il y a présence, l'abondance.

Il existe des modèles spécifiques pour ce genre de données qui utilisent une mixture de distribution binomiale et de Poisson : Zero Inflated Poisson Models, Zero Inflated Negative Binomial models,...

Mais on peut aussi :

- transformer les données de comptages en présence/absence

- agrandir les unités d'échantillonnage (afin d'avoir moins de 0) ou rassembler des unités plus petites

pex : si on a mesuré l'abondance d'une espèce dans 10 quadrats de 1 m², on sommera ces valeurs pour obtenir une seule estimation du nombre d'individus par 10 m²

Surdispersion

Surdispersion : solutions ?

Si la surdispersion est raisonable on peut en tenir compte dans les inférences avec des méthodes adaptées :

- quasilikelihood
- utilisation de QAICc
- test de F au lieu du LRT
- correction des erreurs standard
-utiliser d'autres distributions comprenant un paramètre de variance

Negative Binomiale (pour une distribution de Poisson),

Beta Binomiale (pour une distribution binomiale)

On peut aussi ajouter dans un modèle mixte une variable aléatoire correspondant à un identifiant unique pour chaque observation.

Si la surdispersion est vraiment très élevée, il est vraisemblable que le modèle n'est pas adapté et il vaut sans doute mieux renoncer à l'utiliser tel quel même avec des "corrections".

Surdispersion

Exemple

Simulation de données pour un modèle de Poisson surdispersé.
Deux méthodes donnant des résultats légèrement différents
(surdispersion additive vs multiplicative)
pour une surdispersion identique

1) simulation au moyen d'une distribution négative binomiale qui permet de fixer exactement le paramètre de surdispersion (multiplicative) à 4

```
overdisp <- 4
n <-5
x <- rep(0:14, each = n)
set.seed(12)
lambda <- 1 - 0.15 * x
set.seed(1)
y2 <- rnbinom(n*15, size = exp(lambda)/(overdisp-1), mu = exp(lambda))
```

Surdispersion

Exemple

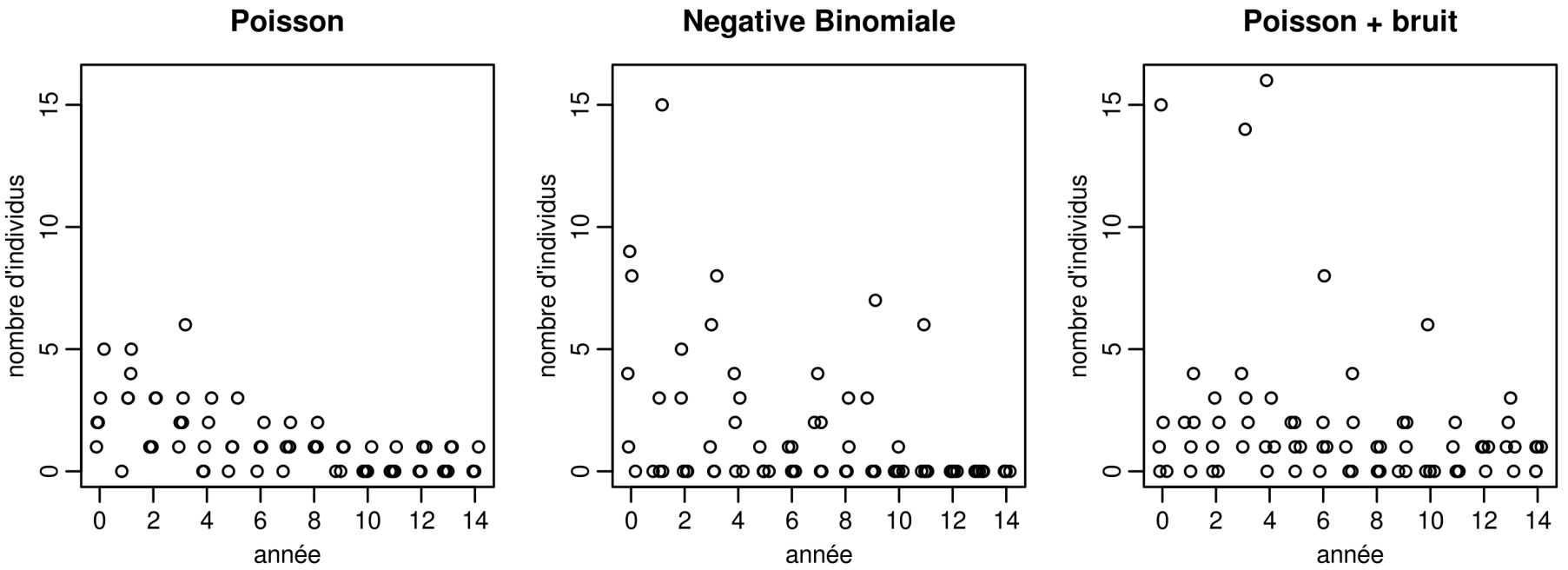
2) simulation en ajoutant du bruit (surdispersion additive) au "linear predictor". On a choisi un écart type de façon à obtenir une surdispersion de 4 également

```
n <-5
x <- rep(0:14, each = n)
set.seed(12)
lambda <- 1 - 0.15 * x + rnorm(n*15, 0, 1.15)
set.seed(1)
y3 <- rpois(n*15, exp(lambda))
```

Surdispersion

Exemple

Simulation de données pour modèle de poisson surdispersé



Surdispersion

Tests de Wald corrigés :

```
> mod <- glm(y2 ~ x, family = poisson)
> summary(mod)
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   1.3840     0.1423   9.723  < 2e-16 ***
x            -0.2015     0.0278  -7.250 4.18e-13 ***

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 280.22  on 74  degrees of freedom
Residual deviance: 215.94  on 73  degrees of freedom
AIC: 297.63

> overdisp(mod)
pearsonresid     deviance
    4.069687     2.958097
> summary(mod, dispersion = overdisp(mod)[1])

            Estimate Std. Error z value Pr(>|z|)
(Intercept)  1.38397    0.28715   4.820 1.44e-06 ***
x           -0.20154    0.05608  -3.594 0.000326 ***

(Dispersion parameter for poisson family taken to be 4.069687)

    Null deviance: 280.22  on 74  degrees of freedom
Residual deviance: 215.94  on 73  degrees of freedom
AIC: 297.63
```

Analyse classique non corrigée

Tests de Wald corrigés. NB : les coefficients sont les mêmes mais leur erreur standard a été multipliée par sqrt(4.07)

Attention l'AIC n'est pas corrigé et n'est pas correct !!

Surdispersion

Comparaison de modèles

On utilise une statistique de test adaptée du test de rapport de vraisemblance qui suit approximativement une distribution de F

```
> overdisp(mod)
pearsonresid     deviance
    4.069687     2.958097

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

Model:
y2 ~ x
       Df Deviance    AIC F value    Pr(>F)
<none>      215.94 297.63
x       1   280.22 359.91  21.731 1.381e-05 ***
---
Warning message:
In drop1.glm(mod, test = "F") :
  le test F implique une famille 'quasipoisson'

> Anova(mod, test = "F")

Error estimate based on Pearson residuals

               SS Df      F    Pr(>F)
x          64.283  1 15.796 0.0001641 ***
Residuals 297.087 73
---
```

$$F = \frac{\left(D_{small} - D_{large}\right) / \left(df_{small} - df_{large}\right)}{\varphi}$$

NB : drop1 utilise le coefficient de surdispersion basé sur la déviance alors que Anova utilise l'estimation basée sur les résidus de Pearson

Surdispersion

Quasilikelihood

L'argument 'family" permet de spécifier "quasipoisson" et "quasibinomial". Dans ce cas le coefficent de surdispersion est automatiquement estimé et les erreurs standard ajustées.

```
> mod <- glm(y2 ~ x, family = poisson)
> summary(mod)
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   1.3840     0.1423   9.723  < 2e-16 ***
x            -0.2015     0.0278  -7.250 4.18e-13 ***

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 280.22  on 74  degrees of freedom
Residual deviance: 215.94  on 73  degrees of freedom
AIC: 297.63

> mod <- glm(y2 ~ x, family = quasipoisson)
> summary(mod)
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.38397    0.28716   4.820 7.61e-06 ***
x           -0.20154    0.05608  -3.594 0.000589 ***

(Dispersion parameter for quasipoisson family taken to be 4.069724)

    Null deviance: 280.22  on 74  degrees of freedom
Residual deviance: 215.94  on 73  degrees of freedom
AIC: NA
```

L'AIC n'est plus disponible

Surdispersion

Quasilikelihood
Comparaison de modèles

```
> mod <- glm(y2 ~ x, family = quasipoisson)
> drop1(mod, test = "F")
Single term deletions

Model:
y2 ~ x
       Df Deviance F value    Pr(>F)
<none>      215.94
x       1   280.22  21.731 1.381e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

> Anova(mod, test = "F")
Analysis of Deviance Table (Type II tests)

Response: y2
Error estimate based on Pearson residuals

               SS Df      F    Pr(>F)
x          64.283  1 15.796 0.0001641 ***
Residuals 297.087 73
```

curieusement, drop1 n'utilise pas l'estimation du coefficient de surdispersion estimé par le modèle `Anova(mod, test = "F")` par contre l'utilise

Surdispersion

Negative Binomial model

On peut utiliser la fonction `nb.glm` du package MASS qui est construite sur `glm`

```
> mod <- glm(y2 ~ x, family = quasipoisson)
> summary(mod)
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.38397    0.28716   4.820 7.61e-06 ***
x           -0.20154    0.05608  -3.594 0.000589 ***

(Dispersion parameter for quasipoisson family taken to be 4.069724)

    Null deviance: 280.22  on 74  degrees of freedom
Residual deviance: 215.94  on 73  degrees of freedom
AIC: NA
```

Pour les modèles binomiaux, on utilisera un modèle Beta Binomial disponible dans le package aod

L'interprétation est exactement identique. On peut choisir les mêmes fonctions de lien.

```
> library(MASS)
> mod <- glm.nb(y2 ~ x)
> summary(mod)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   1.4410     0.4198   3.433 0.000598 ***
x            -0.2106     0.0577  -3.650 0.000262 ***

(Dispersion parameter for Negative Binomial(0.3306) family taken to be 1)

    Null deviance: 70.272  on 74  degrees of freedom
Residual deviance: 56.584  on 73  degrees of freedom
AIC: 210.29
```

paramètre theta qui est une combinaison de paramètres

Attention, le modèle négatif binomial peut lui-même être surdispersé !

On dispose ici d'un AIC valide

Distribution des résidus

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

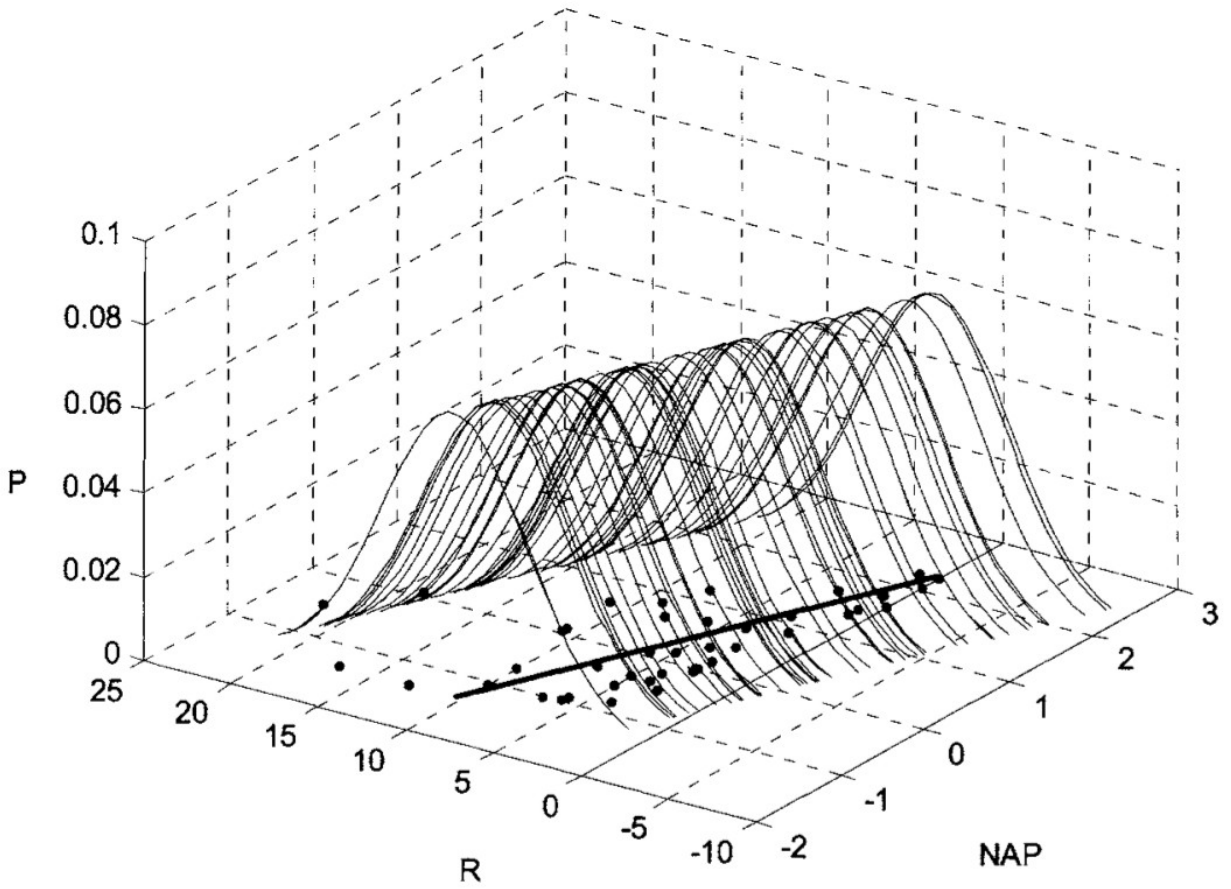
+ l'erreur de mesure des X doit être négligeable

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,... ?

Distribution des résidus

Pour chaque valeur de chaque variable explicative, la distribution des résidus devrait approximativement suivre une distribution normale ou Poisson ou Binomiale etc... selon le type de modèle choisi.

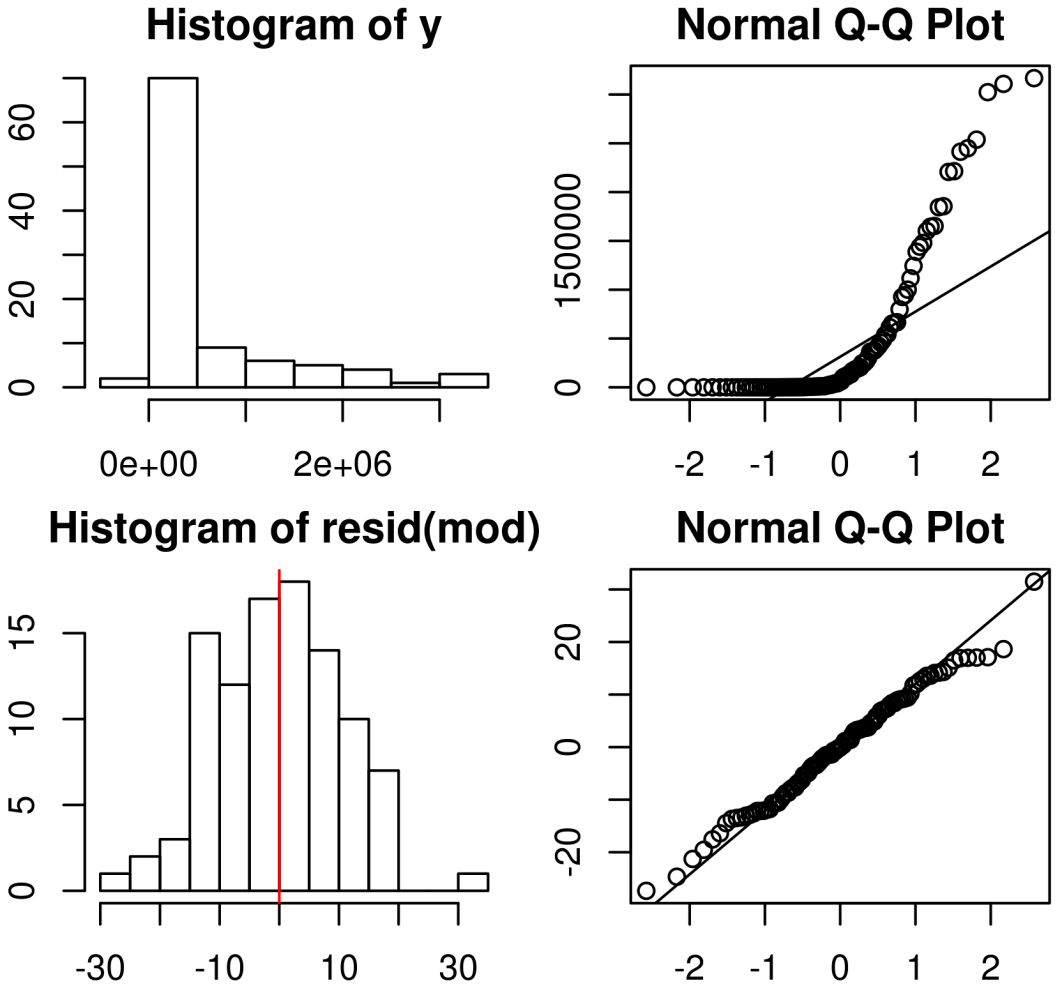


Distribution des résidus

Diagnostique avec des qq plots et des histogramme des résidus

Rappel : même si en théorie la variable mesurée doit suivre la distribution choisie (pour chaque valeur de x) c'est bien les résidus qu'il faut examiner pour les diagnostiques

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

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

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



Distribution des résidus

Solutions ?

- Changer de distribution !!

- Améliorer le modèle : linéarité, interactions, valeurs extrêmes,...

- transformer Y (pour les distributions gaussiennes)

- utiliser des tests par permutation ou du bootstrap (en particulier pour les modèles estimés avec lm)

Rappel : de toutes les conditions d'applications c'est sans doute la moins importante en particulier pour les modèles estimés par la méthode des moindres carrés

Homogénéité de la variance des résidus

A propos des tests de normalité et d'homoscedasticité

Ces tests sont en général inutiles

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

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

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

Homogénéité de la variance des résidus

A propos des tests de normalité et d'homoscedasticité

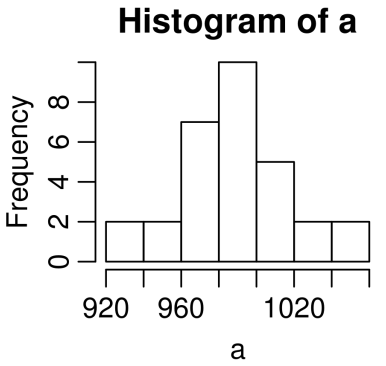
```
set.seed(11)
a <- rpois(30, 1000)
a10 <- rep(a,10)

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

> shapiro.test(a)

     Shapiro-Wilk normality test

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

> shapiro.test(a10)

     Shapiro-Wilk normality test

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

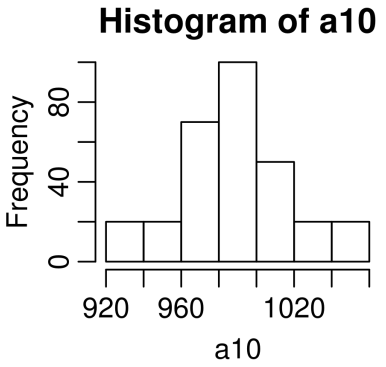
a = 30 obs avec une distribution de poisson de moyenne = 1000

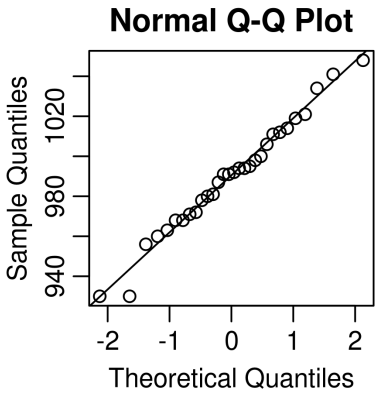
a10 = a répété 10 fois --> la distribution est exactement la même !

a ne diffère pas significativement d'une distribution normale

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







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

Indépendance des résidus

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,... ?

Indépendance des résidus

Hypothèse

Chaque observation apporte une information indépendante des autres.
Cette condition est surtout importante pour obtenir des inférences correctes, moins pour l'estimation des paramètres

Normalement l'échantillonnage aléatoire et la randomisation des traitements garantissent en partie l'indépendance.

Typiquement la non indépendance survient de deux manières :

1) par design : parce qu'on a choisi de ne pas faire un échantillonnage aléatoire mais plutôt des groupes d'échantillons ou de sous-échantillonner ou de répéter des mesures
2) à cause de corrélation spatiale ou temporelle entre les observations

Indépendance des résidus

Non Indépendance par design

Il faut bien réfléchir au design expérimental avant de faire l'expérience et toujours favoriser les vrais réplicats (du traitement) plutôt que des pseudo-réplicats quand c'est possible.

En cas de pseudoréplication, il faut incorporer au modèle les variables décrivant la relation entre points d'échantillonnage

Typiquement on ajoute les variables décrivant les groupes d'observations comme variables aléatoires, souvent hiérarchisées (pex blocs, sites, vaches dans des fermes dans des régions).

On peut aussi résumer les mesures répétées en une seule mesure par exemple en prenant la moyenne par groupe.

Indépendance des résidus

Non Indépendance par design

Il n'y a en général pas de diagnostic pour ce genre de problèmes. C'est la manière de récolter les données qui guide la modélisation.

Dans certains cas on peut cependant ajouter une variable comme effet aléatoire de groupe et calculer le coefficient de corrélation intra-classe (variance divisée par la variance totale de la partie aléatoire) pour estimer la corrélation entre les observations.

Indépendance des résidus

Corrélation spatiale et temporelle

Ce type de corrélation a typiquement deux origines :

1) des variables explicatives qui influencent Y sont elles-mêmes réparties de manière non uniforme dans le temps et dans l'espace.

Pex : si il y a un gradient d'humidité Nord Sud et que la variable que l'on mesure est influencée par l'humidité, des points plus proches vont en moyenne avoir des valeurs plus proches de Y.

Indépendance des résidus

Corrélation spatiale et temporelle

Ce type de corrélation a typiquement deux origines :

2) la variable Y a une influence directe sur la la valeur de Y en un point proche dans l'espace ou le temps
(c'est l'autocorrélation au sens strict).

Pex : L'abondance d'une espèce l'année A aura sans doute une influence sur son abondance l'année A+1 ou la présence d'une espèce sur un site peut avoir un effet négatif (répulsion) ou positif (aggrégation) sur la probabilité de présence dans un point d'échantillonnage voisin.

Indépendance des résidus

Corrélation spatiale et temporelle

Les corrélations spatiales et temporelles sont des phénomènes statistiquement très similaires et les méthodes pour les traiter sont souvent proches voire identiques.

La corrélation temporelle :
dépend d'une distance dans le temps
est unidimensionnelle
les données sont souvent prélevées à intervalles réguliers mais pas toujours

La corrélation spatiale :
d'une distance dans l'espace
est bidimensionnelle ou unidimensionnelle (transects, rivières,...)
les données sont souvent prélevées de manière irrégulière mais pas toujours

Indépendance des résidus

Corrélation spatiale et temporelle

On peut distinguer 3 grandes raisons de s'intéresser à ces corrélations :

on va s'attarder sur ce point ici

1) comme une nuisance

elle augmente le risque d'erreur de type I à cause de la pseudoréplication et parfois brouille les tendances principales qui sont l'intérêt premier. L'objectif est donc de s'en débarrasser..

2) comme une information utile que l'on utilise dans un but prédictif en particulier pour l'interpolation (pex krigeage)

3) comme un sujet d'étude en tant que tel

ie on veut mettre en évidence des structures cycliques, on veut savoir à quelle échelle spatiale se produit tel phénomène, on veut distinguer l'effet spatial environnemental de l'effet purement auto-corrélatif, etc...

Indépendance des résidus

Corrélation spatiale et temporelle

Diagnostique

Typiquement, une fois que l'on a un modèle global qui nous paraît satisfaisant (linéarité, variance et distribution des résidus etc...) on calcule la corrélation entre résidus situés à plusieurs distances spatiales ou temporelles.

Les mesures d'auto-corrélation ne sont valables que sur des séries "stationnaires" c'est à dire ne montrant pas de tendance globale dans le temps ou dans l'espace.

Avec les modèles temporels, le temps est en général inclus dans le modèle et les résidus ne comprennent donc pas de tendance temporelle.

Pour les données spatiales, il faut en général faire un modèle des résidus en fonction des coordonnées géographiques des points avant de calculer les corrélations. Si il y a une tendance il faut en général incorporer les coordonnées géographiques au modèle original et utiliser les nouveaux résidus

Indépendance des résidus

Auto-corrélation temporelle

Les données temporelles sont souvent prises à des intervalles réguliers et leur corrélation se mesure alors avec un coefficient d'auto-corrélation proche du coefficient de corrélation de Pearson

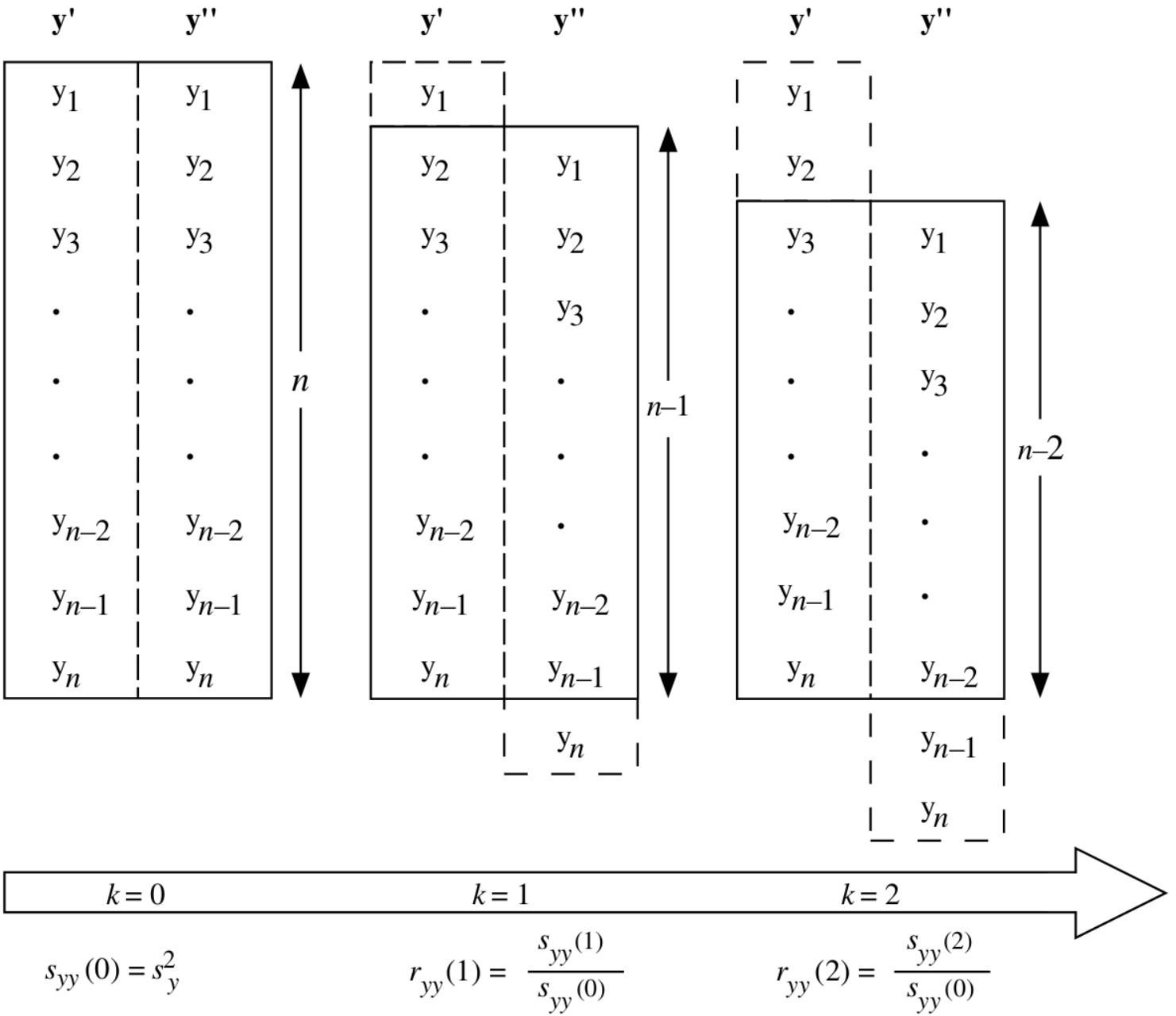
Auto-covariance = covariance de y avec elle-même après un décalage ("lag") de k unités temporelles

$$r_{yy}(k) = \frac{S_{yy}(k)}{S_{yy}(0)} = \frac{\frac{1}{n}\sum_{i=1}^{n-k}(y_{i+k}-\bar{y})(y_i-\bar{y})}{\frac{1}{n}\sum_{i=1}^{n}(y_i-\bar{y})^2}$$

Variance de y = autocovariance pour un décalage de k=0

NB : le dénominateur reste identique quelle que soit la valeur de k
--> différence avec une corrélation de Pearson

Indépendance des résidus



Legendre & Legendre (2012)

Indépendance des résidus

Auto-corrélation temporelle

L'autocorrélation se mesure pour plusieurs distances (lags) --> fonction d'auto-corrélation (afc) représentée en général graphiquement en fonction de la distance.
Plus k augmente, plus le nombre de données pour estimer $r_{yy}(k)$ est petit entraînant une précision plus faible

Dans R fonction `acf()` avec une méthode pour `plot()`.
Ou fonction `ACF` pour plusieurs séries (modèles mixtes) dans le package `nlme`

Indépendance des résidus

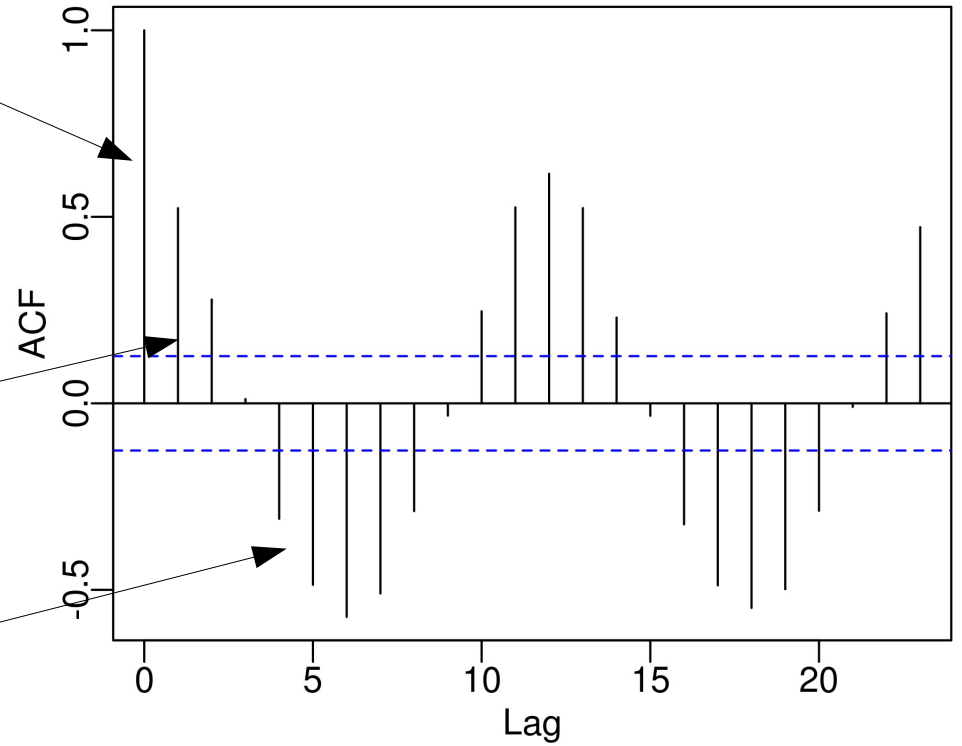
Auto-corrélation temporelle

```
> mod <- lm(y ~ time , data=d2)
> acf(resid(mod))
```

Corrélation = 1 pour k=0

Corrélation positive pour k = 1 ou 2

Corrélation négative pour k = 4 à 8



Pattern cyclique de longeur = 12

Indépendance des résidus

Auto-corrélation spatiale

L'***indice I de Moran*** est une généralisation du coefficient d'autocorrélation pour des distances (spatiales ou temporelles) irrégulières. On mesure la covariance entre points situés dans une gamme de distances au lieu de distances fixes.

prend la valeur 1 quand les points h et i se trouvent dans la classe de distance d, 0 dans le cas contraire

$$I(d) = \frac{\frac{1}{W}\sum_{h=1}^{n}\sum_{i=1}^{n} w_{hi}(y_h-\bar{y})(y_i-\bar{y})}{\frac{1}{n}\sum_{i=1}^{n}(y_i-\bar{y})^2} \qquad pour\ h \neq i$$

I a une valeur comprise en général entre -1 (corrélation négative) et + 1(corrélation positive) peut être parfois légèrement hors de ces limites

Indépendance des résidus

Auto-corrélation spatiale

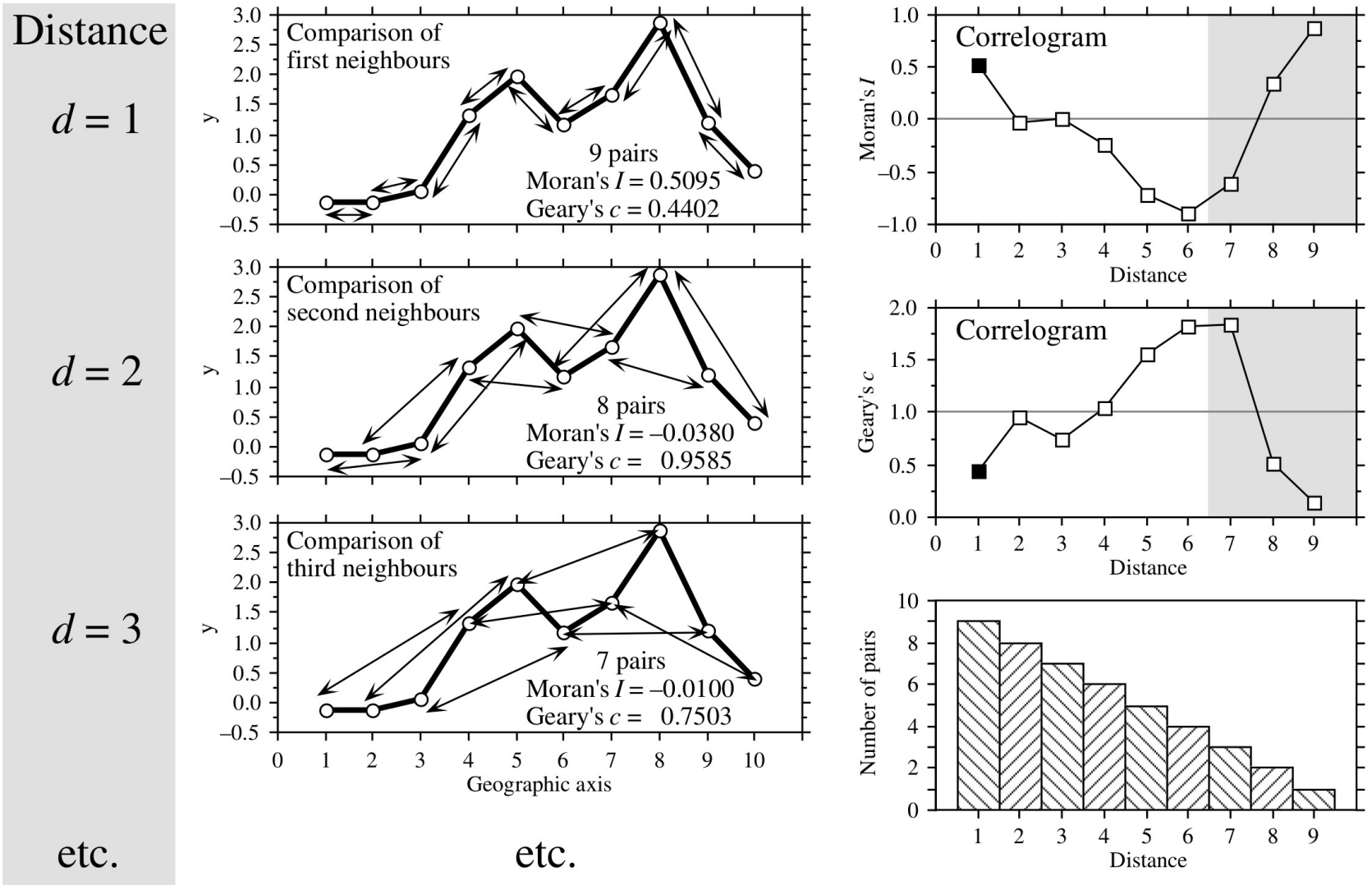
Pour les corrélations spatiales, on utilise aussi parfois l'***indice c de Geary***.
La corrélation est positive quand c est entre 0 et 1 et négative quand c > 1.

$$c(d) = \frac{\frac{1}{2W}\sum_{h=1}^{n}\sum_{i=1}^{n} w_{hi}(y_h - y_i)^2}{\frac{1}{n-1}\sum_{i=1}^{n}(y_i - \bar{y})^2} \qquad pour\ h \neq i$$

Le graphique représentant I ou c en fonction de la classe de distance est appelé **corrélogramme** (spatial)

Indépendance des résidus

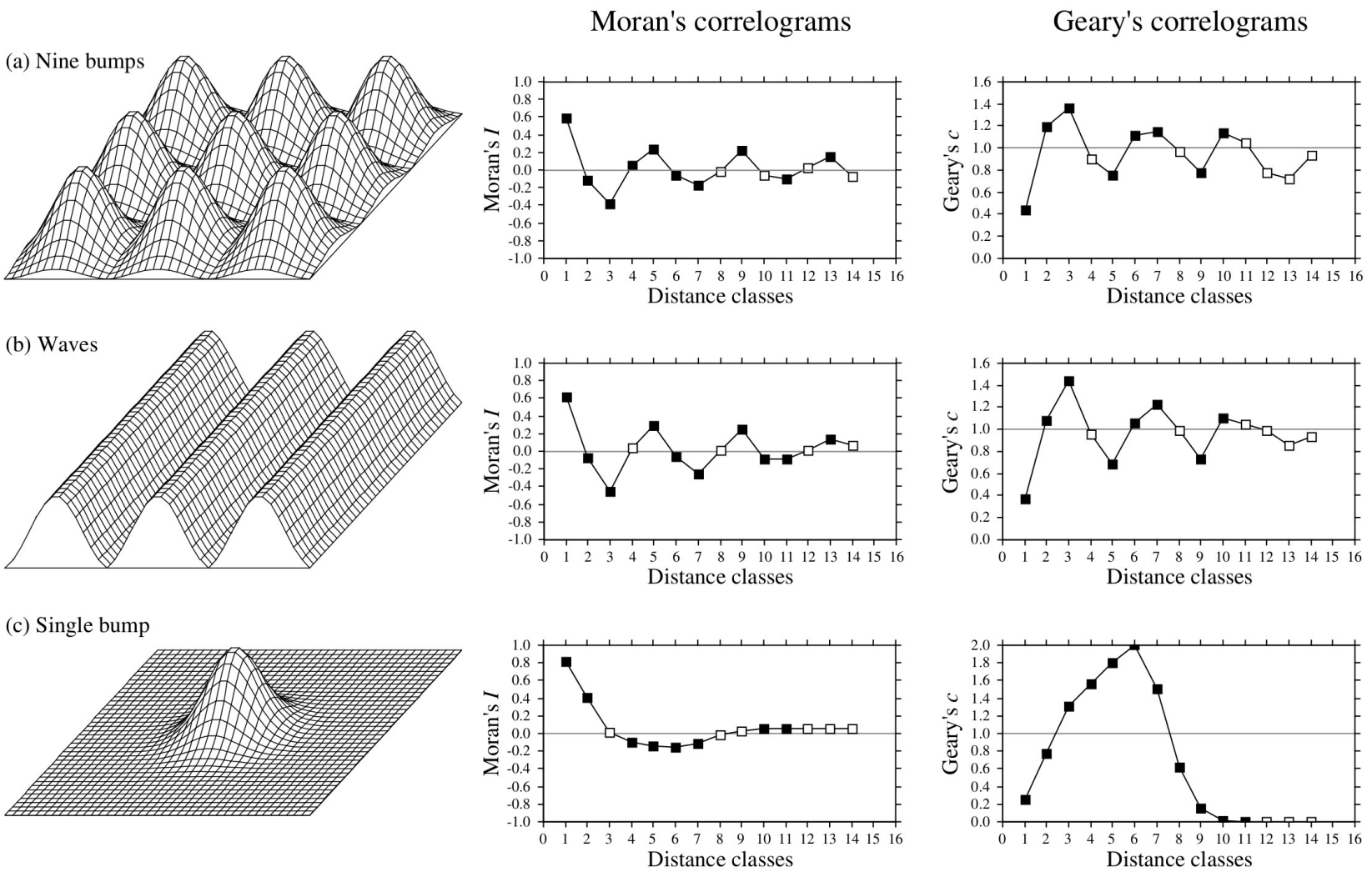
Auto-corrélation spatiale



Legendre & Legendre (2012)

Indépendance des résidus

Auto-corrélation spatiale



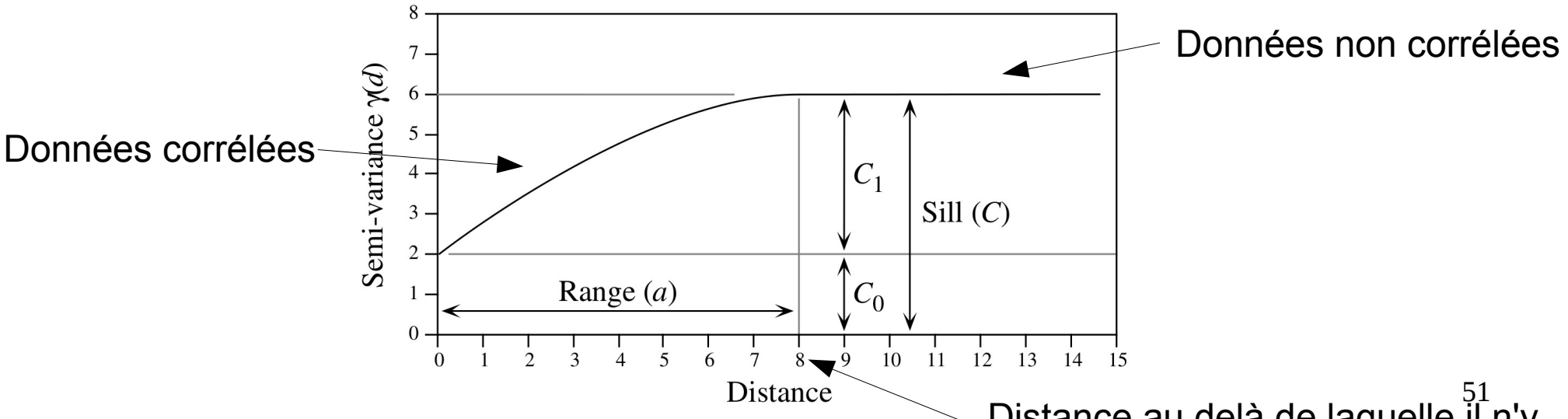
Legendre & Legendre (2012)

Indépendance des résidus

Auto-corrélation spatiale

On représente aussi parfois un **variogramme** ou semi-variogramme qui représente graphiquement le numérateur du c de Geary appelé semi-variance.

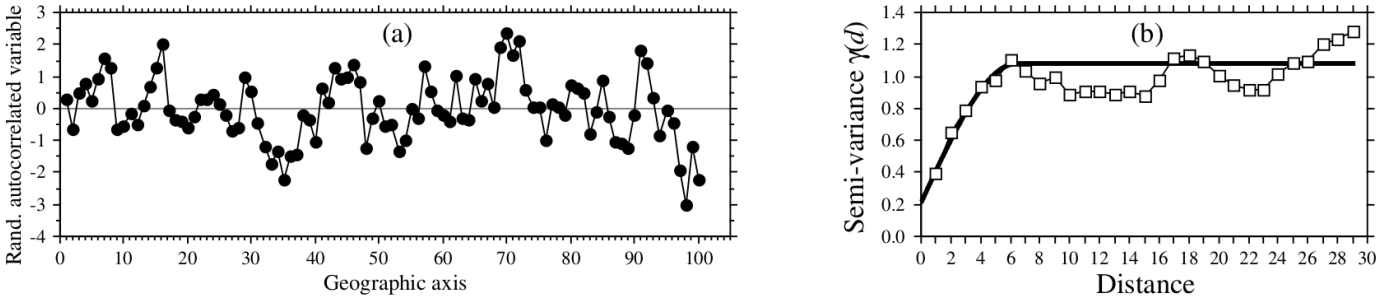
En général on ne réalise pas de test sur le variogramme contrairement aux corrélogrammes mais on essaye de représenter la semi-variance par un modèle mathématique (non-linéaire en général) qui peut être utilisé ensuite pour de la prédiction/interpollation (pex krigeage) ou pour introduire une structure de corrélation dans les résidus d'un modèle

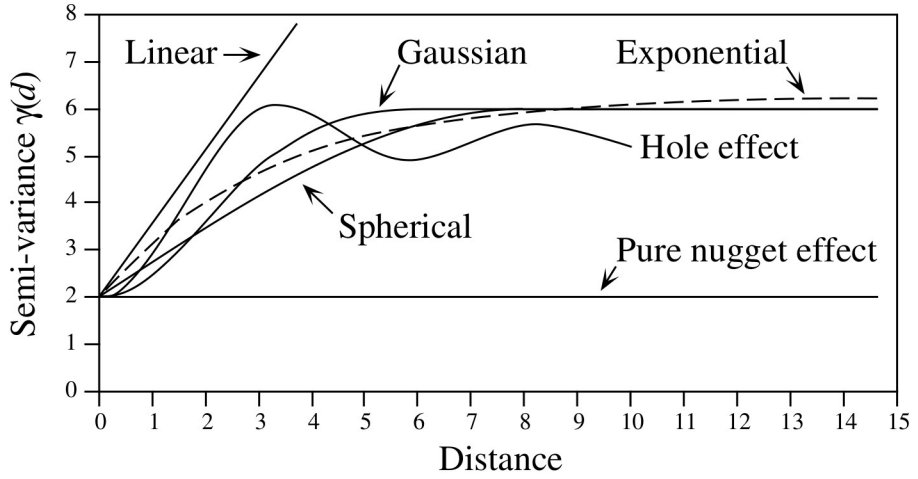


Legendre & Legendre (2012)

Indépendance des résidus

Auto-corrélation spatiale





Legendre & Legendre (2012)

Indépendance des résidus

Auto-corrélation spatiale

Dans R on peut calculer et représenter les corrélogrammes par exemple avec les fonctions `sp.correlogram` et `dnearneigh` du package `spdep`

La difficulté est de trouver la bonne gamme de distance pour éviter d'avoir trop peu de valeurs par classe de distance, et d'avoir un nombre ni trop petit ni trop grand de classes.

```
> library(vegan)
> data(mite)
> data(mite.xy)
> data(mite.env)
> library(spdep)
> xy <- as.matrix(mite.xy[, c("x", "y")])
```

Exemple modélisation de l'abondance d'un acarien en fonction de l'humidité du sol d'après Borcard et al 2011

Indépendance des résidus

Auto-corrélation spatiale

```
> mod <- lm(mite$SUCT ~ mite.env[,2])
> summary(mod)
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   38.60473    4.32564   8.925 4.62e-13 ***
mite.env[, 2] -0.05272    0.00996  -5.293 1.39e-06 ***

> mod <- lm(resid(mod) ~ xy)
> summary(mod)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   7.9602     3.5296   2.255   0.0274 *
xyx          -3.5057     1.7444  -2.010   0.0485 *
xyy          -0.7239     0.5030  -1.439   0.1548
```

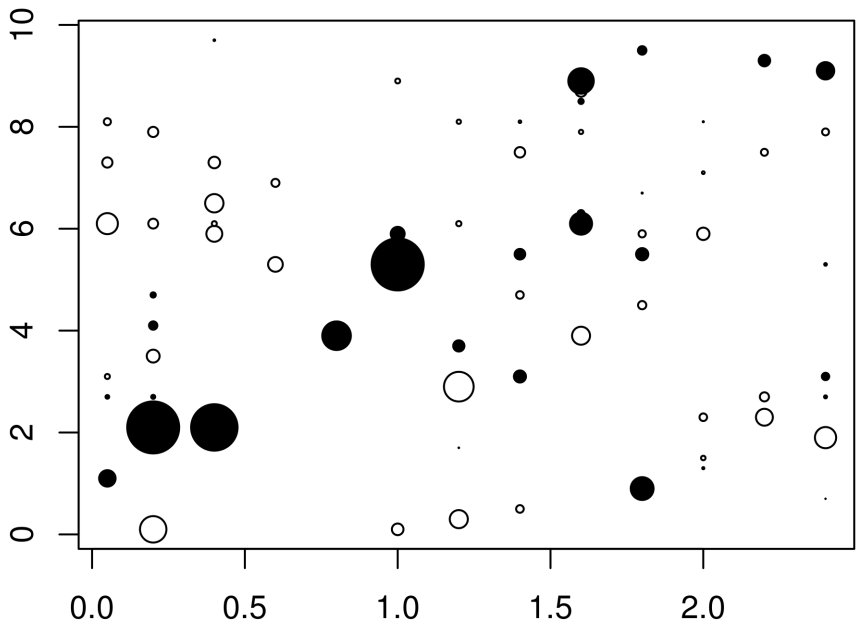
Il y a une légère tendance spatiale (non stationnarité)
--> va travailler sur les résidus de ce deuxième modèle

Indépendance des résidus

Auto-corrélation spatiale

On peut représenter les résidus du modèle sur leurs coordonnées géographiques

```
mycol <- c("white", "black")[ifelse(resid(mod)<0, 1, 2)]
plot(xy[,"x"], xy[,"y"], cex = abs(resid(mod))/10, bg = mycol, pch = 21)
```



Points proportionnels à la taille du résidu.
Noir : résidu positif, blanc : résidu négatif.

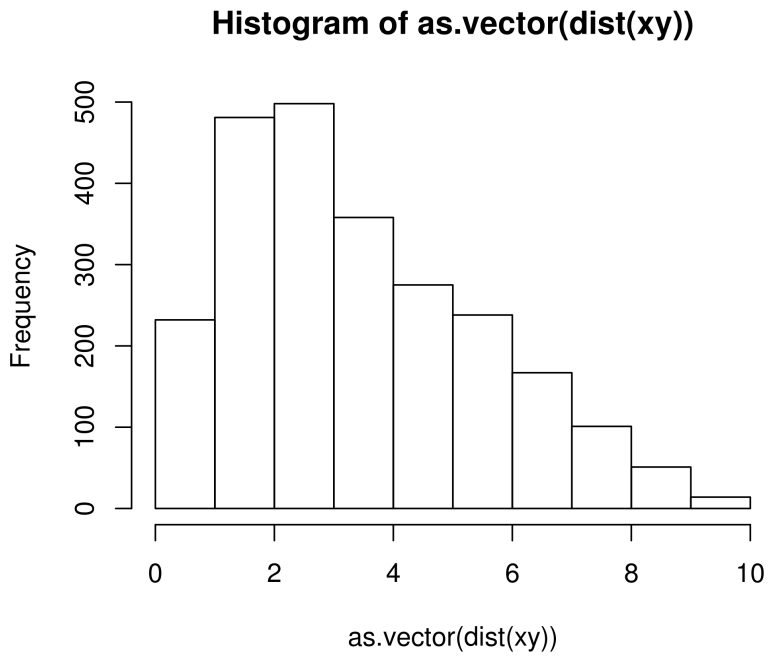
Lorsqu'il y a de fortes corrélations spatiales, il y a des groupes de points noirs ou blancs.

Indépendance des résidus

Auto-corrélation spatiale

On explore ensuite en général les distances entre les points pour choisir les classes de distances

```
> a <- as.matrix(dist(xy, upper=TRUE))
> is.na(diag(a)) <- TRUE
> max(apply(a, 1, min, na.rm = TRUE))
[1] 1.011187

> hist(as.vector(dist(xy)))
```



La distance au plus proche voisin la plus grande est de 1.011 mètres.
Si on choisis des classes < que 1.01 m certains points se retrouveront seuls dans la première classe.
Cela peut arriver mais il faut éviter d'avoir trop de points isolés

La gamme de distances va de 0 à 10 mètres

On aurait donc 10 mesures en choisissant un pas de 1m et 20 mesures en choisissant 0.5 m (mais avec des valeurs isolées au premier pas)

Indépendance des résidus

Auto-corrélation spatiale

`sp.correlogram` utilise un objet de classe `nb` créé par la fonction `dnearneigh` contenant une liste de points situés à une distance déterminée les uns des autres

```
> summary(dnearneigh(xy, 0, 1))
Neighbour list object:
Number of regions: 70
Number of nonzero links: 454
Percentage nonzero weights: 9.265306
Average number of links: 6.485714
1 region with no links:
7
Link number distribution:

 0  1  3  4  5  6  7  8  9 10 11 14
 1  2  8  6 11  8  9  6  9  7  1  2
(...)

> summary(dnearneigh(xy, 0, 0.5))
(...)
9 regions with no links:
1 5 7 10 28 31 35 65 70
Link number distribution:
 0  1  2  3  4
 9 22 21 10  8
(...)
```

Avec un pas de 1 m : on a 1 point isolé et 2 points avec seuelement 1 voisin à une distance de 1m

Avec un pas de 0.5 m on a 9 points isolés et 22 points avec seulement 1 voisin --> distance trop courte

Indépendance des résidus

Auto-corrélation spatiale

Calcul du I de Moran pour des classes de distance de 1 m

```
> nb <- dnearneigh(xy, 0, 1)
> correlog <- sp.correlogram(nb, resid(mod),  method="I", order=10, zero.policy = TRUE)
> print(correlog, p.adj.method="holm")
Spatial correlogram for resid(mod)
method: Moran's I
          estimate expectation   variance standard deviate Pr(I) two sided
1 (69)   0.1996281  -0.0147059  0.0046862           3.1310         0.01742 *
2 (69)  -0.0456190  -0.0147059  0.0033106          -0.5373         1.00000
3 (69)  -0.0355419  -0.0147059  0.0027905          -0.3944         1.00000
4 (69)  -0.0696779  -0.0147059  0.0027237          -1.0533         1.00000
5 (69)   0.0069478  -0.0147059  0.0031446           0.3861         1.00000
6 (69)  -0.0146220  -0.0147059  0.0039847           0.0013         1.00000
7 (69)   0.0213127  -0.0147059  0.0054146           0.4895         1.00000
8 (68)  -0.1827645  -0.0149254  0.0068323          -2.0305         0.33842
9 (59)  -0.0670648  -0.0172414  0.0087341          -0.5331         1.00000
10 (47)  0.2185780  -0.0217391  0.0122774           2.1689         0.27084
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
> plot(correlog)
```

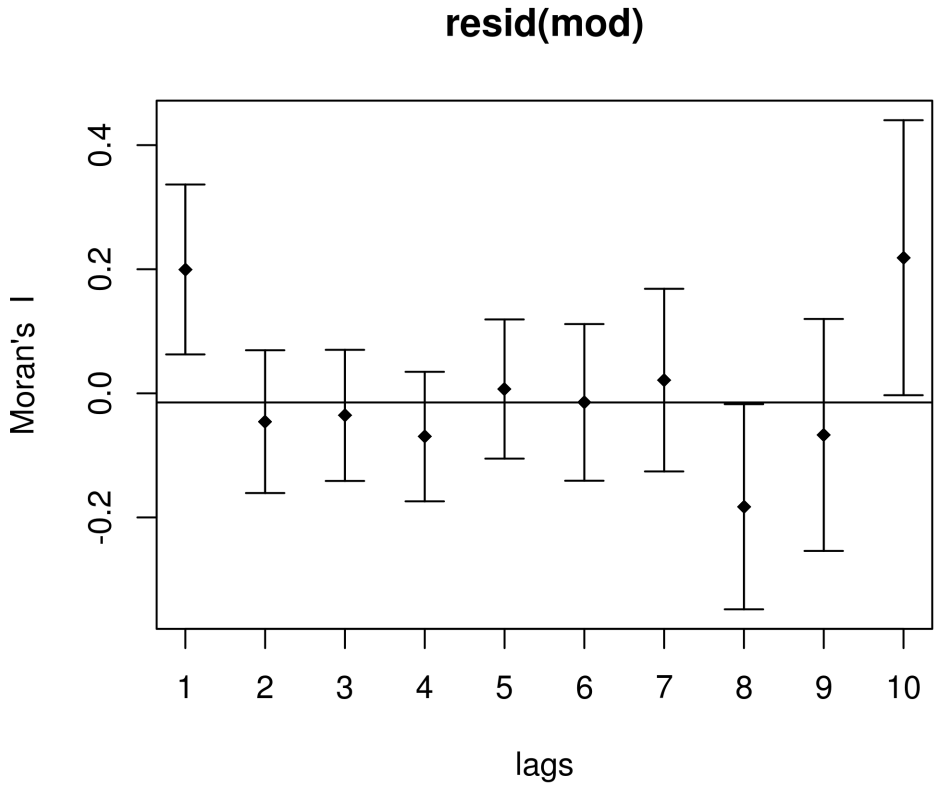
p.valeur ajustée pour le nombre de comparaisons -->
la correction dépend du nombre de classes

Indépendance des résidus

Auto-corrélation spatiale

Calcul du I de Moran pour des classes de distance de 1 m

```
> plot(correlog)
```



Il y a une légère corrélation positive entre les sites situés entre 0 et 1 m les uns des autres

Indépendance des résidus

Auto-corrélation spatiale

Calcul du c de Geary pour des classes de distance de 1 m

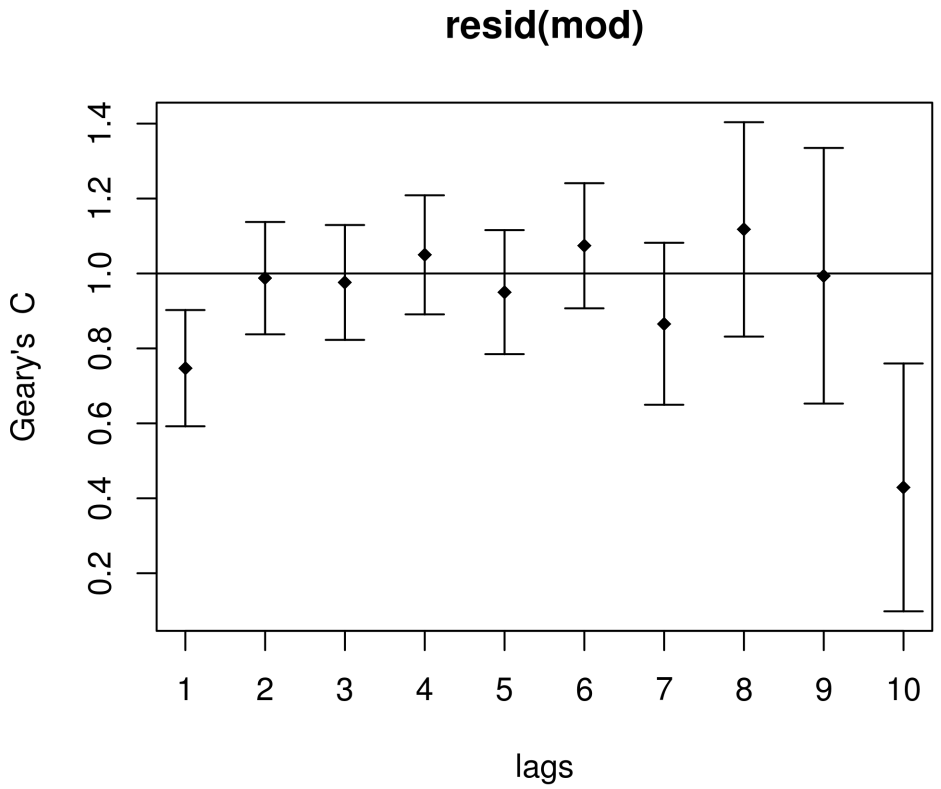
```
> correlog <- sp.correlogram(nb,  resid(mod),  method="C", order=10, zero.policy = TRUE)
> print(correlog, p.adj.method="holm")
Spatial correlogram for resid(mod)
method: Geary's C
         estimate expectation  variance standard deviate Pr(I) two sided
1 (69)  0.7473110   1.0000000 0.0060145          -3.2583        0.010089 *
2 (69)  0.9875097   1.0000000 0.0056192          -0.1666        1.000000
3 (69)  0.9760586   1.0000000 0.0058737          -0.3124        1.000000
4 (69)  1.0497704   1.0000000 0.0063124           0.6264        1.000000
5 (69)  0.9502693   1.0000000 0.0068477          -0.6010        1.000000
6 (69)  1.0739732   1.0000000 0.0069693           0.8861        1.000000
7 (69)  0.8657610   1.0000000 0.0116795          -1.2421        1.000000
8 (68)  1.1176518   1.0000000 0.0204611           0.8225        1.000000
9 (59)  0.9938951   1.0000000 0.0290892          -0.0358        1.000000
10 (47) 0.4290974   1.0000000 0.0273438          -3.4525        0.005554 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
> plot(correlog)
```

Indépendance des résidus

Auto-corrélation spatiale

Calcul du c de Geary pour des classes de distance de 1 m

```
> plot(correlog)
```

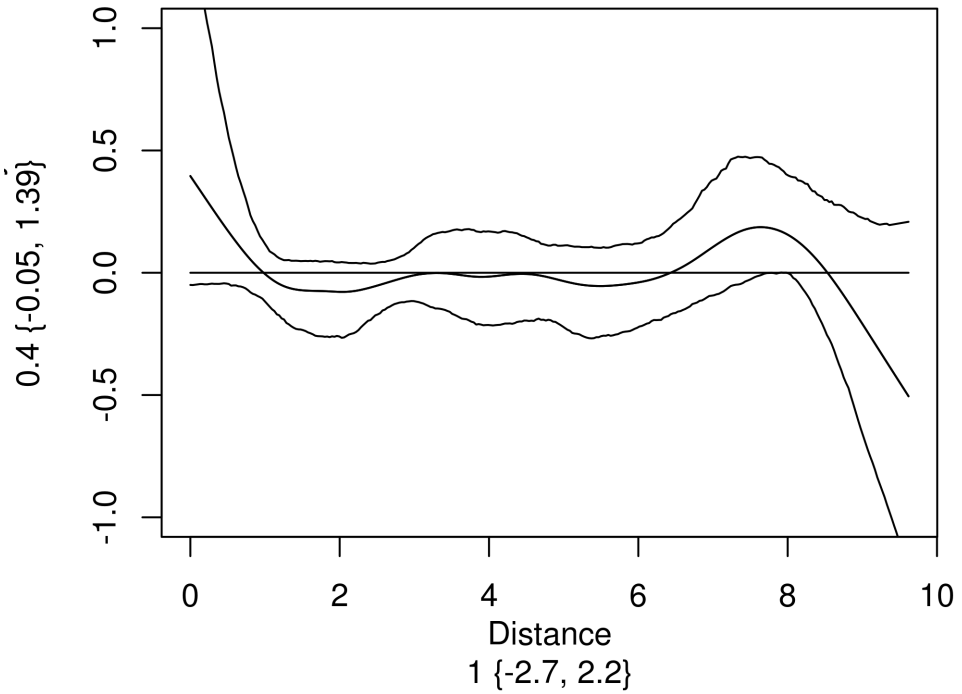


Indépendance des résidus

Auto-corrélation spatiale

Le package `ncf` fourni aussi une fonction de corrélogramme avec courbe de lissage et intervalle de confiance par bootstrap

```
library(ncf)
correlog <- spline.correlog(x = xy[,"x"], y = xy[,"y"], z = resid(mod))
plot(correlog)
```



Indépendance des résidus

Que faire quand on constate qu'il y a de la corrélation spatiale ou temporelle dans les résidus ?

1) Ajouter des variables environnementales elles-même corrélées

2) réduire/synthétiser les données
(pex prendre la moyenne par an plutôt que toutes les données détaillées)

3) Incorporer une structure de corrélation dans la matrice variance-covariance des résidus du modèle

4) ajouter des variables non-environnementales au modèle :

- les valeurs de y à des points d'échantillonnage proches dans le temps ou l'espace, souvent avec une pondération et/ou une mesure de la corrélation (auto-regressive models)

- les coordonnées x et y des termes polynomiaux et leurs interactions

- des vecteurs propres spatiaux (ie Moran Eigen Maps)

- etc...

Ensuite vérifier sur le corrélograme que la corrélation a bien disparu

Indépendance des résidus

Exemple : corrélation temporelle

Mesures de température mensuelles sur 15 sites pendant 20 ans.
On veut savoir si il y a une tendance linéaire dans les températures

```
# simulation du jeu de données
nsites <- 16
ny <- 20
n <- nsites * ny * 12

# création des variables site et year
site <- paste("site", rep(sprintf("%02.0f", 1:nsites), each = ny*12), sep = "_")
year <- rep(rep(1:ny, times = nsites), each = 12)
month <- as.factor(rep(1:12, times = nsites*ny))
time <- rep(c(1:(12*ny)), times = nsites)

d <- data.frame(site, year, month, time)

# moyenne et variance des pentes et variance résiduelle
int.mean <- 0
int.sd <- 3
slope.mean <- 0.15
slope.sd <- 0.05
sigma <- 5
monthbeta <- c(1, 3, 9, 13, 14, 18, 22, 18, 13, 8, 3)

# Génération des pentes et des intercepts pour chaque groupe
set.seed(1)
int <- rnorm(n = nsites, mean = int.mean, sd = int.sd )
set.seed(2)
slope <- rnorm(n = nsites, mean = slope.mean, sd = slope.sd )
beta <- c(int, monthbeta , slope)

X <- model.matrix (~ site + site : year - 1 + month)
lin.pred <- X %*% beta
set.seed(3)
y <- rnorm(n = n, mean = lin.pred, sd = sigma)

d <- data.frame(y, site, year, month, time)
```

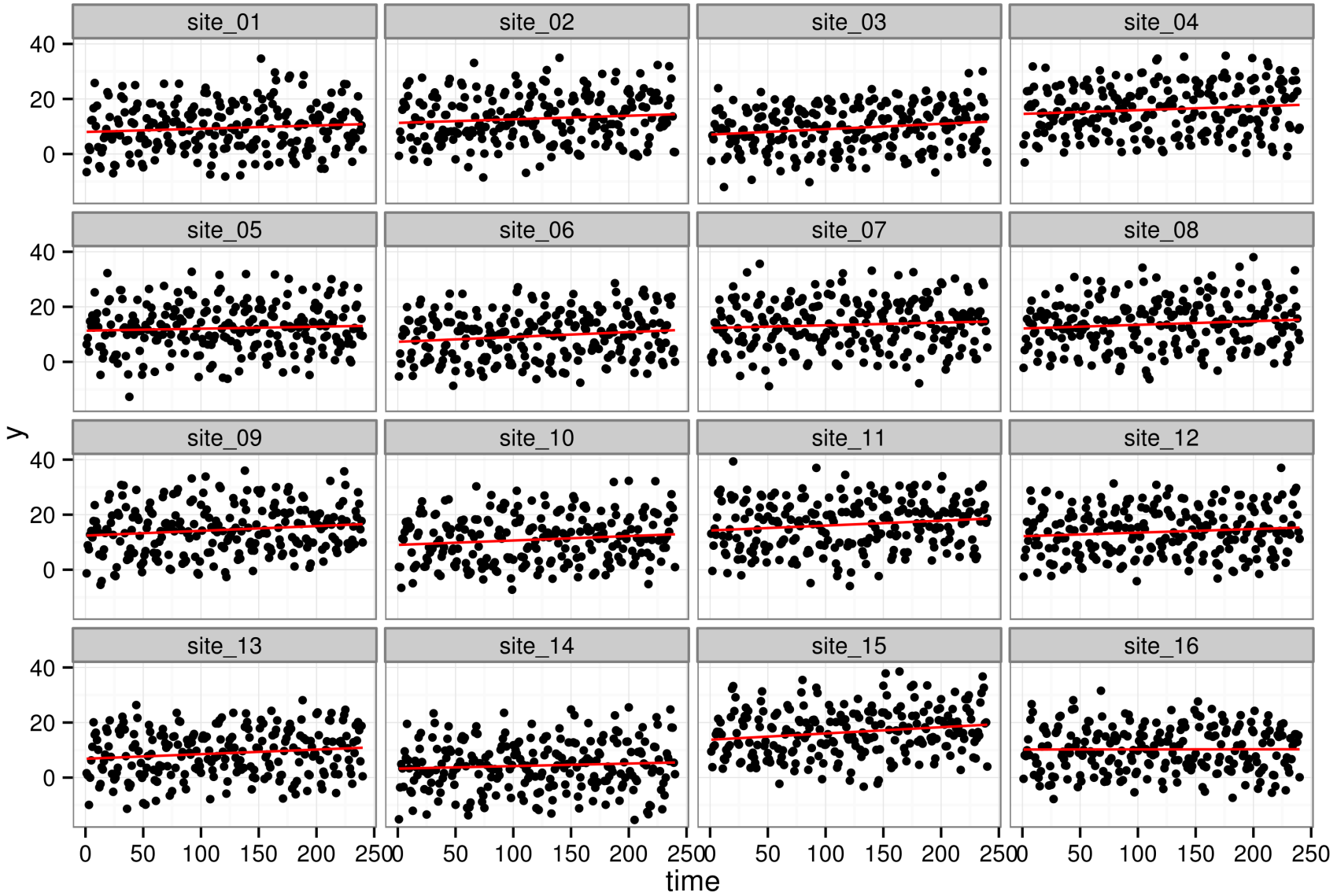
Indépendance des résidus

Exemple : corrélation temporelle

```
library(ggplot2)

ggplot(data=d, aes(y = y, x = time)) +
    geom_point(shape = 20) +
    stat_smooth(method = "lm", se = FALSE, color = "red") +
    facet_wrap(~site) + theme_bw()

ggplot(data=d, aes(y = y, x = time)) +
    geom_line() +
    stat_smooth(method = "lm", se = FALSE, color = "red") +
    facet_wrap(~site) + theme_bw()
```

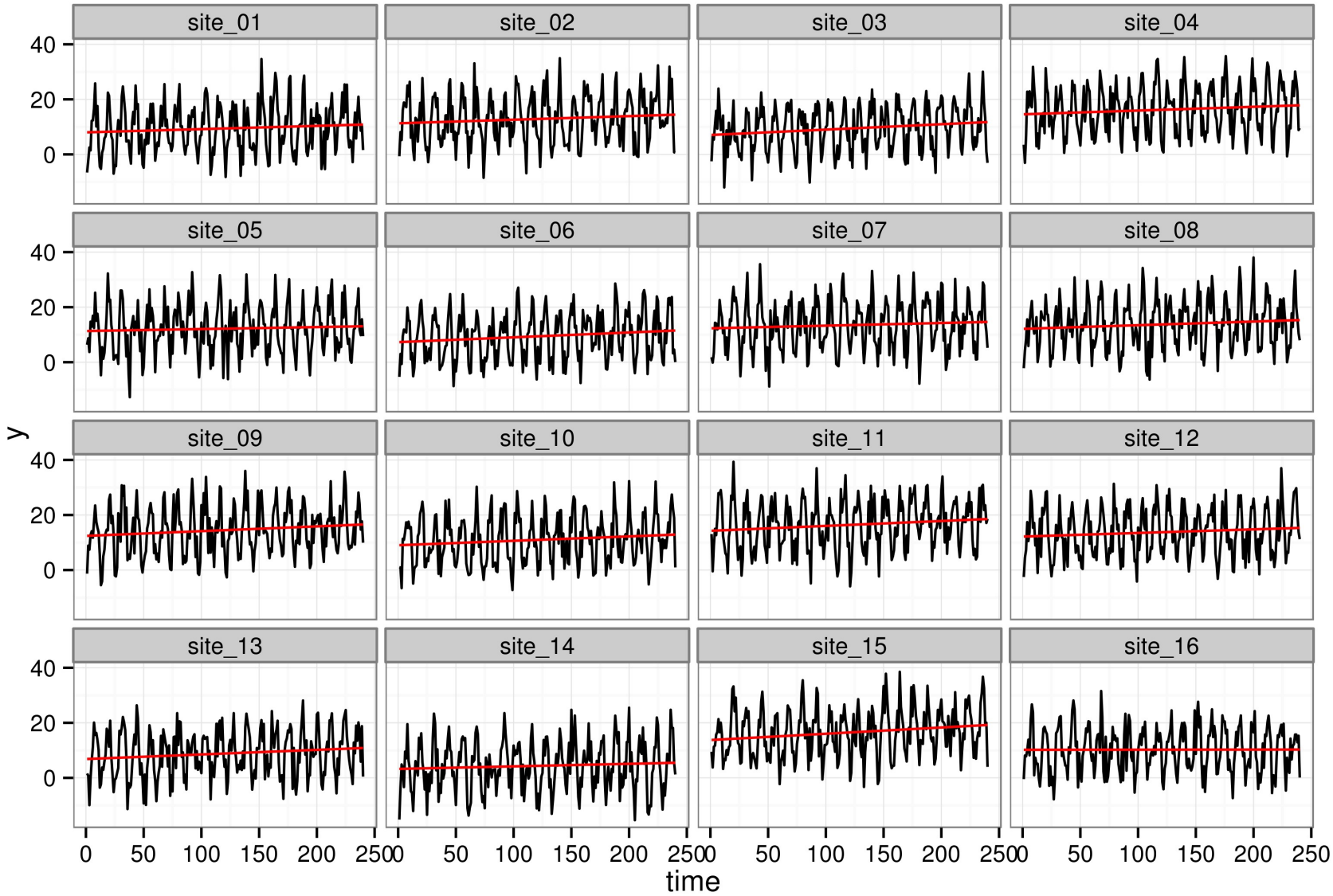
Indépendance des résidus

Exemple : corrélation temporelle



Indépendance des résidus

Exemple : corrélation temporelle



Indépendance des résidus

Exemple : corrélation temporelle

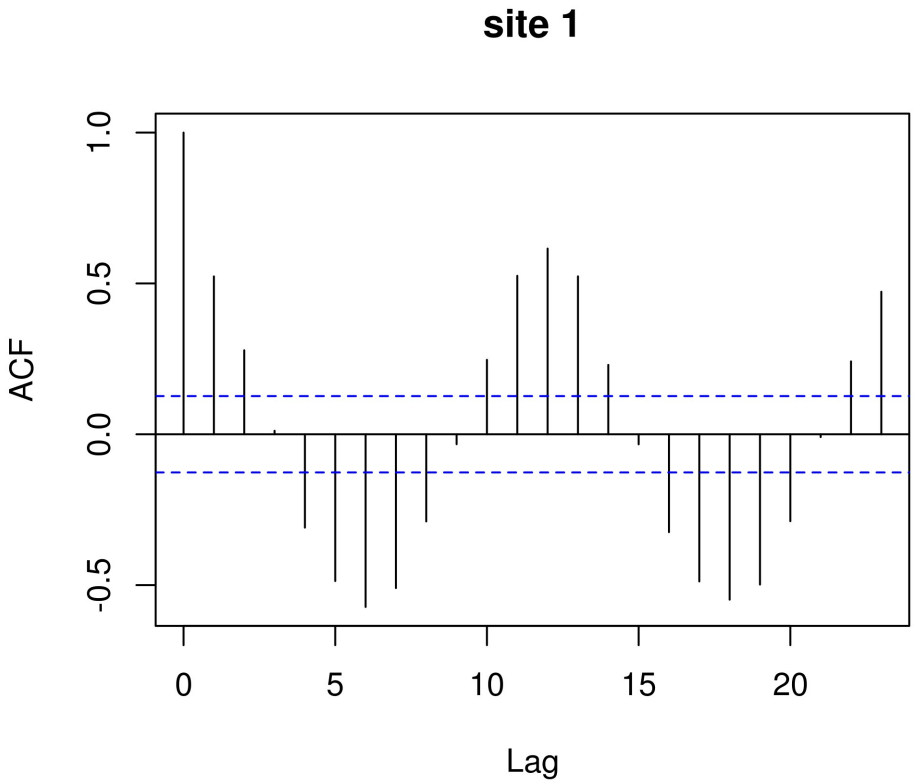
```
> library(lme4)
> mod <- lmer(y ~ time + (time | site), data=d)
> summary(mod)
Linear mixed model fit by REML ['lmerMod']
Formula: y ~ time + (time | site)
   Data: d

REML criterion at convergence: 27434.44

Random effects:
 Groups   Name        Variance  Std.Dev. Corr
 site     (Intercept) 9.416e+00 3.068524
          time        2.799e-06 0.001673 1.00
 Residual             7.294e+01 8.540726
Number of obs: 3840, groups: site, 16

Fixed effects:
             Estimate Std. Error t value
(Intercept) 10.368705   0.815445  12.715
time         0.013692   0.002033   6.736

Correlation of Fixed Effects:
     (Intr)
time -0.094
```

Indépendance des résidus

Exemple : corrélation temporelle

Fonction d'auto-corrélation pour un seul site

```
# ACF pour un site
res <- mod@frame
res$resid <- resid(mod, type="pearson")
acf(res[res$site == "site_01","resid"], main = "site 1")
```

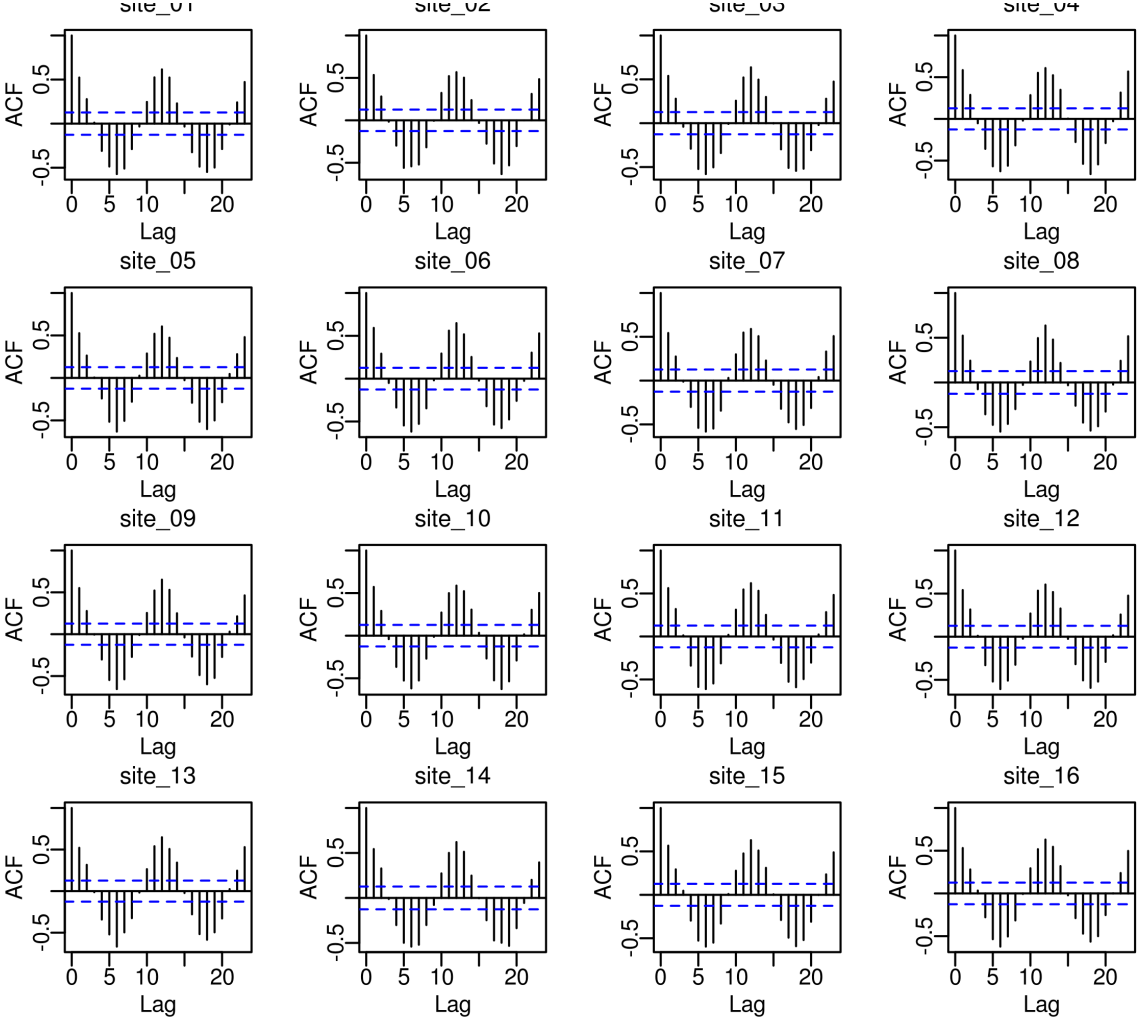


Indépendance des résidus

Exemple : corrélation temporelle

Fonction d'auto-corrélation pour chaque site

```
# ACF pour chaque site
par(mfrow=c(4,4), mar = c(3,3,1,1), mgp = c(1.4, 0.4, 0))
for (s in levels(res$site)) {
    acf(res[res$site == s,"resid"])
    mtext(s, line = 0.5, cex = 0.7)
}
```



Indépendance des résidus

La fonction ACF du package nlme permet d'avoir un seul acf pour l'ensemble des sites mais il ne fonctionne qu'avec des modèles estimés avec gls et lme. Ce serait possible ici (avec lme) mais voici la fonction adaptée pour fonctionner avec d'autres données groupées :

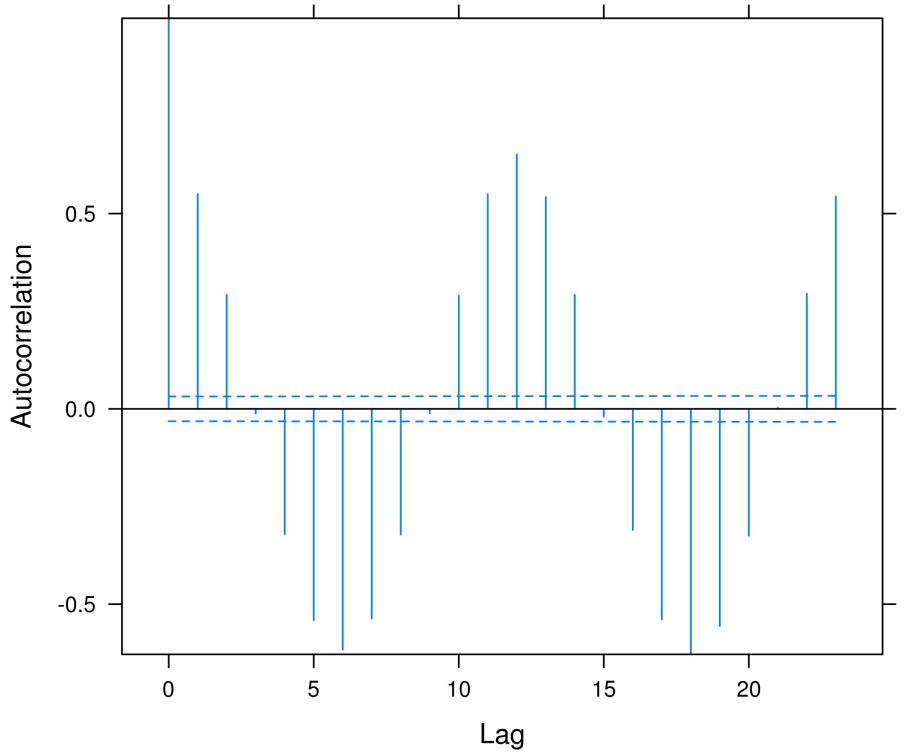
```
ACF.grouped <- function (res, time, group, maxLag, ...)
{
    d <- data.frame(res, time, group)
    d <- d[order(group, time),]
    res <- split(d$res, d$group)
    if (missing(maxLag)) {
        maxLag <- min(c(maxL <- max(sapply(res, length)) - 1,
                        as.integer(10 * log10(maxL + 1))))
    }
    val <- lapply(res, function(el, maxLag) {
        N <- maxLag + 1
        tt <- double(N)
        nn <- integer(N)
        N <- min(c(N, n <- length(el)))
        nn[1:N] <- n + 1 - 1:N
        for (i in 1:N) {
            tt[i] <- sum(el[1:(n - i + 1)] * el[i:n])
        }
        array(c(tt, nn), c(length(tt), 2))
    }, maxLag = maxLag)
    val0 <- apply(sapply(val, function(x) x[, 2]), 1, sum)
    val1 <- apply(sapply(val, function(x) x[, 1]), 1, sum)/val0
    val2 <- val1/val1[1]
    z <- data.frame(lag = 0:maxLag, ACF = val2)
    attr(z, "n.used") <- val0
    class(z) <- c("ACF", "data.frame")
    z
}
```

Indépendance des résidus

Exemple : corrélation temporelle

Fonction d'auto-corrélation globale

```
# ACF global
par(mfrow=c(1,1), mar = c(3,3,1,1), mgp = c(1.4, 0.4, 0))
modacf <- ACF.grouped(resid(mod, type="pearson"), d$time, d$site)
library(nlme)
plot(modacf, alpha = 0.05)
```



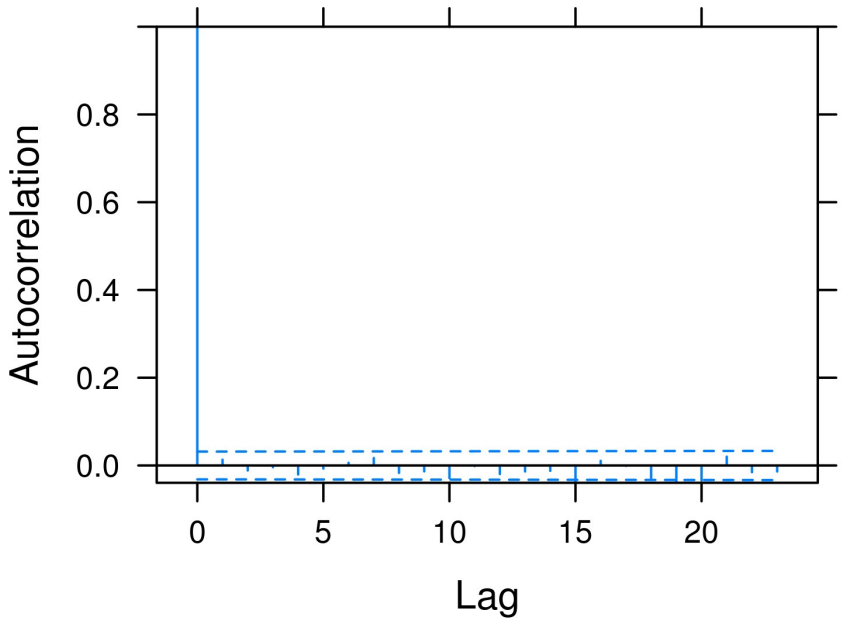
Indépendance des résidus

Exemple : corrélation temporelle

Ajouter une variable explicative supplémentaire (le mois) résout complètement le problème ici.

NB : normalement on s'arrêterait ici.

```
mod <- lmer(y ~ time + month + (time | site), data=d)

par(mfrow=c(1,1), mar = c(3,3,1,1), mgp = c(1.4, 0.4, 0))
modacf <- ACF.grouped(resid(mod, type="pearson"), d$time, d$site)
library(nlme)
plot(modacf, alpha = 0.05)
```



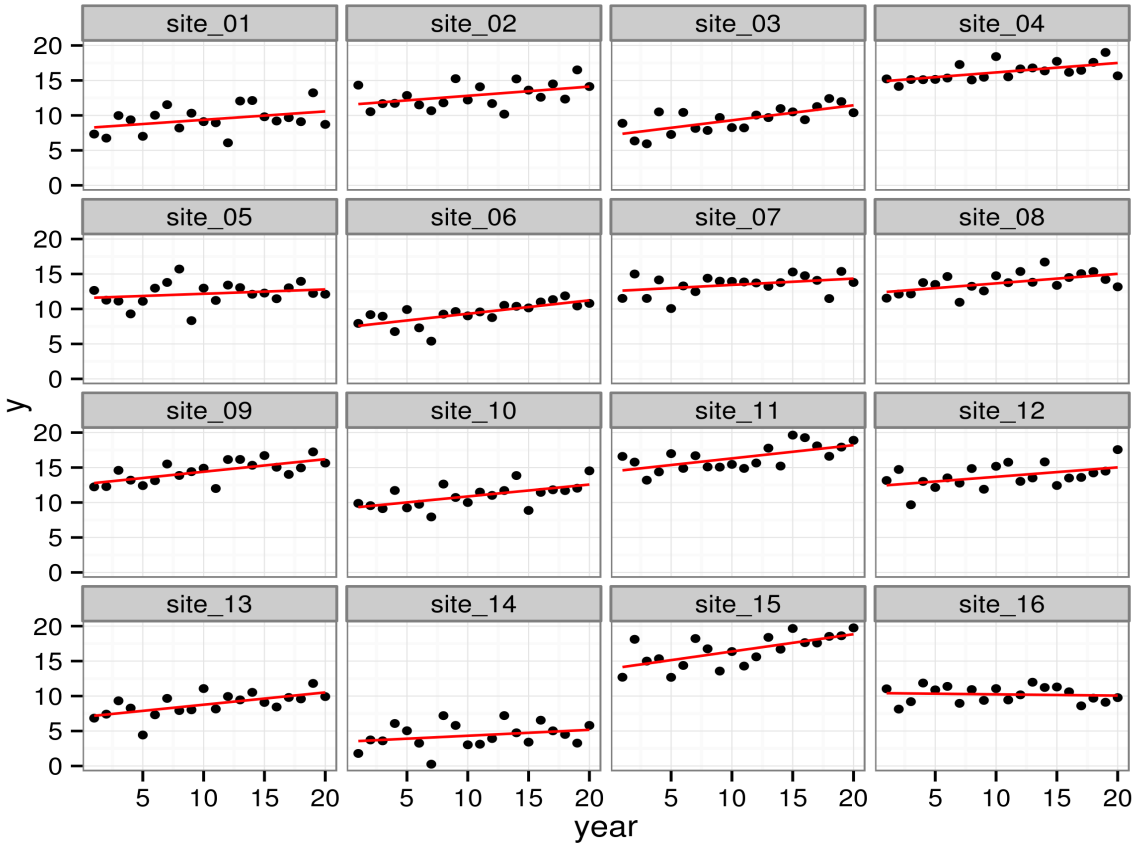

```
> summary(mod)
Fixed effects:
             Estimate Std. Error t value
(Intercept)  1.031033   0.836817    1.23
time         0.011700   0.001351    8.66
month2       0.617864   0.396286    1.56
month3       2.441835   0.396291    6.16
month4       8.552499   0.396300   21.58
month5      12.312154   0.396312   31.07
month6      13.362646   0.396327   33.72
month7      16.685032   0.396346   42.10
month8      21.398938   0.396369   53.99
month9      17.664085   0.396394   44.56
month10     12.566196   0.396424   31.70
month11      7.265472   0.396456   18.33
month12      2.067074   0.396493    5.21
```

Indépendance des résidus

Exemple : corrélation temporelle

Utiliser la moyenne par année plutôt que les données mensuelles

```
d2 <- aggregate(d["y"], by = d[c("site", "year")], mean)

ggplot(data=d2, aes(y = y, x = year)) +
    geom_point(shape = 20) +
    stat_smooth(method = "lm", se = FALSE, color = "red") +
    facet_wrap(~site) + theme_bw()
```



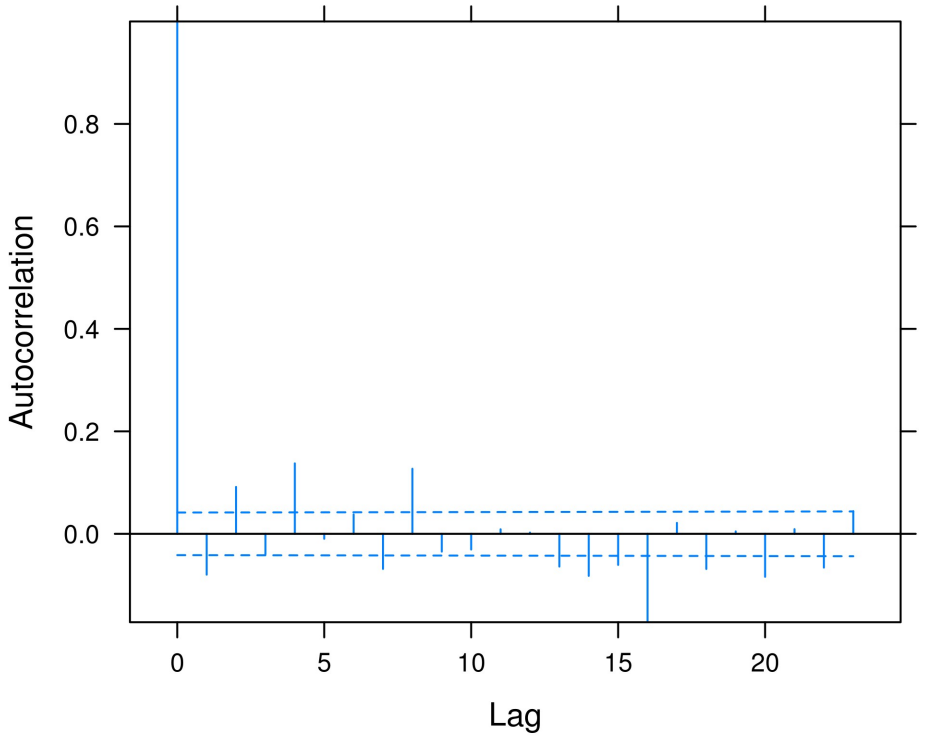
Indépendance des résidus

Exemple : corrélation temporelle

Utiliser la moyenne par année plutôt que les données mensuelles.

Réduit fortement la corrélation dans ce cas

```
mod <- lmer(y ~ year + (year | site), data=d2)
summary(mod)
modacf <- ACF.grouped(resid(mod, type="pearson"), d$time, d$site)
plot(modacf, alpha = 0.01)
```



Indépendance des résidus

Exemple : corrélation temporelle

Introduire une structure de corrélation temporelle dans la matrice de variance covariance des résidus.

Possible entre autres avec le package `nlme`

```
> library(lme4)
> mod <- lmer(y ~ time + (time | site), data=d)
> summary(mod)
Random effects:
 Groups   Name        Variance  Std.Dev. Corr
 site     (Intercept) 9.416e+00 3.068524
          time        2.799e-06 0.001673 1.00
 Residual             7.294e+01 8.540726

Fixed effects:
             Estimate Std. Error t value
(Intercept) 10.368705   0.815445  12.715
time         0.013692   0.002033   6.736
```

```
> library(nlme)
> mod0 <- lme(y ~ time,
        random = ~ time | site, data=d)
> summary(mod)
Random effects:
 Groups   Name        Variance  Std.Dev. Corr
 site     (Intercept) 9.416e+00 3.068524
          time        2.799e-06 0.001673 1.00
 Residual             7.294e+01 8.540726

Fixed effects:
             Estimate Std. Error t value
(Intercept) 10.368705   0.815445  12.715
time         0.013692   0.002033   6.736
>
```


Modèle mixte "normal" : comparaison avec lme4

Indépendance des résidus

Exemple : corrélation temporelle

Introduire une structure de corrélation temporelle dans la matrice de variance covariance des résidus.

Comparaison de différentes structures de corrélations disponibles

NB : aucune de ces structure n'est adaptée réellement à un phénomène aussi cyclique

```
mod1 <- lme(y ~ time, random = ~ time | site, correlation = corAR1(), data=d)
mod2 <- lme(y ~ time, random = ~ time | site, correlation = corARMA(p=1, q=1), data=d)
mod3 <- lme(y ~ time, random = ~ time | site, correlation = corARMA(p=6, q=0), data=d)
```

```
> anova(mod0, mod1, mod2,  mod3)
     Model df      AIC      BIC    logLik   Test   L.Ratio p-value
mod0     1  6 27446.91 27484.42 -13717.45
mod1     2  7 26042.09 26085.85 -13014.04 1 vs 2 1406.8226  <.0001
mod2     3  8 26043.78 26093.80 -13013.89 2 vs 3    0.3093  0.5781
mod3     4 12 24726.21 24801.24 -12351.10 3 vs 4 1325.5694  <.0001
```

Le modèle avec la structure la plus complexe est nettement meilleur mais cela reste insuffisant pour "enlever" toute autocorrélation

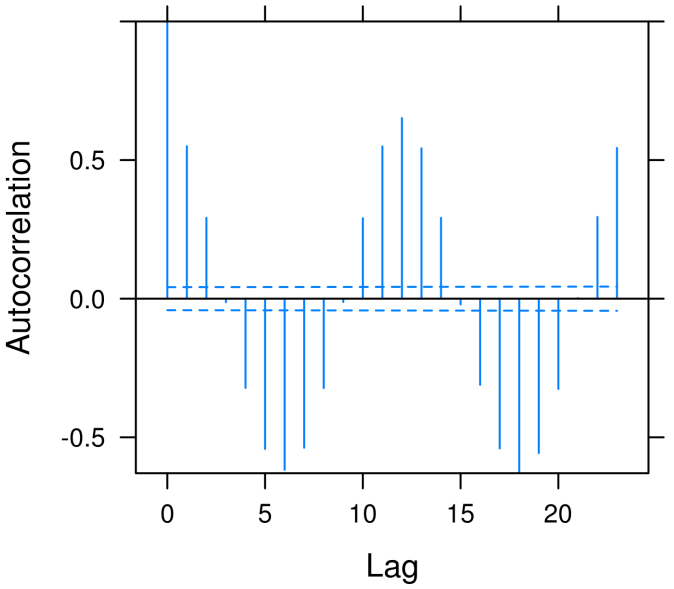
Indépendance des résidus

Exemple : corrélation temporelle

Introduire une structure de corrélation temporelle dans la matrice de variance covariance des résidus.

Le modèle avec la structure la plus complexe est nettement meilleur mais cela reste insufisant pour "enlever" toute autocorrélation

```
modacf <- ACF.grouped(resid(mod3, type="pearson"), d$time, d$site)
plot(modacf, alpha = 0.01)
```



Indépendance des résidus

Exemple : corrélation temporelle

Introduire des vecteurs propres spatiaux comme variables explicatives dans le modèle :
dbMEM "distance based Moran's Eigenvector Maps"

Parfois aussi appelés :
PCNM = Principal Coordinates of Neighbour Matrices

```
# install.packages("PCNM", repos="http://R-Forge.R-project.org")
# install.packages("AEM", repos="http://R-Forge.R-project.org")
# install.packages("packfor", repos="http://R-Forge.R-project.org")

require(vegan)
require(PCNM)
require(packfor)
require(ade4)
library(lme4)
```

Méthodes récentes en plein développement.
Packages a installer directement depuis R forge (pas sur le CRAN)

Indépendance des résidus

Exemple : corrélation temporelle

dbMEM "distance based Moran's Eigenvector Maps"

On travaille sur le site 1 pour commencer

```
mod <- lmer(y ~ time + (time | site), data=d)
summary(mod)

d2 <- d[d$site == "site_01",]
time.dist <- dist(d2$time)
MEM <- PCNM(time.dist, thresh = 24, dbMEM=TRUE, moran=TRUE, all=TRUE) # 160 secondes
> dim(MEM$vectors)
[1] 240 239
```

Création de 240 vecteurs propres indépendants sur base de la matrice de distance entre les 240 points du site 1

On limite les calculs à une distance de 24 mois pour gagner du temps et parce que le phénomène est clairement cyclique

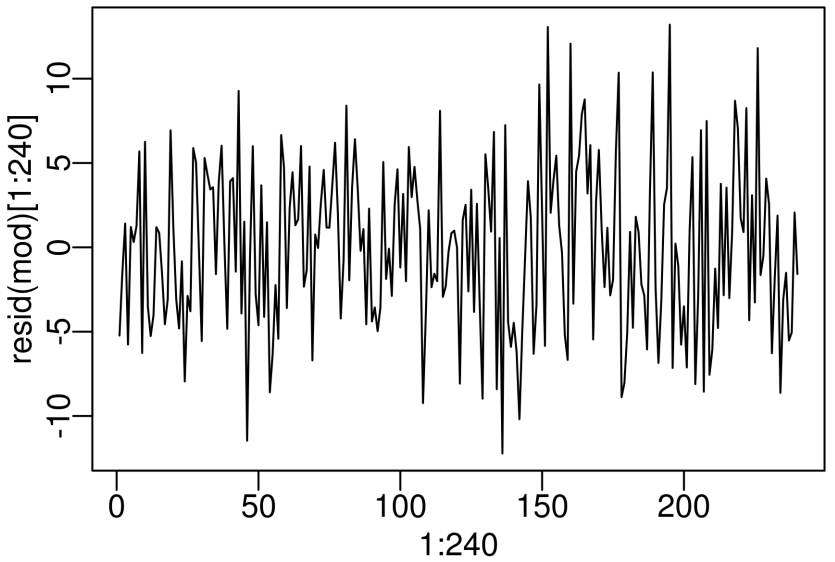
Indépendance des résidus

Exemple : corrélation temporelle

dbMEM "distance based Moran's Eigenvector Maps"

Les MEM sont simplement des variables calculées uniquement sur base des distances et qui peuvent potentiellement modéliser la corrélation spatiale ou temporelle.
Ils peuvent modéliser des relations positives ou négatives et à des échelles très différentes

```
plot(y=resid(mod)[1:240], x=1:240, type="l")
```



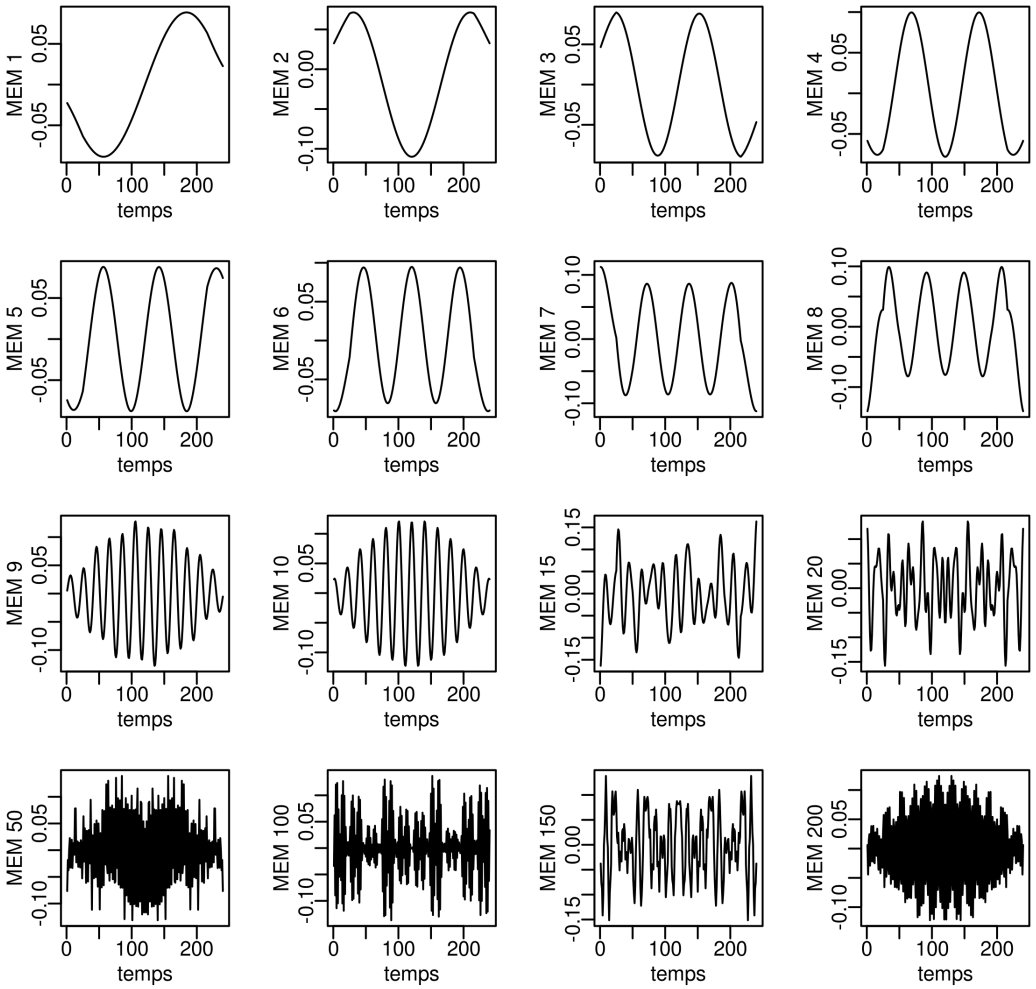
résidus du site 1

Indépendance des résidus

Exemple : corrélation temporelle

dbMEM "distance based Moran's Eigenvector Maps"

```
par(mfrow=c(4,4), mar = c(3,3,1,1), mgp = c(1.4, 0.4, 0))
for (i in c(1:10, 15, 20, 50, 100, 150, 200)) {
    plot(x = d2$time, y = MEM$vectors[,i], type = "l", ylab = paste("MEM",i), xlab = "temps")
}
```



plot de 16 MEM sur les 240 montrant les différentes échelles potentielles

Indépendance des résidus

Exemple : corrélation temporelle

dbMEM "distance based Moran's Eigenvector Maps"

Sélection des MEM qui expliquent une partie de la structure des résidus pour le site 1

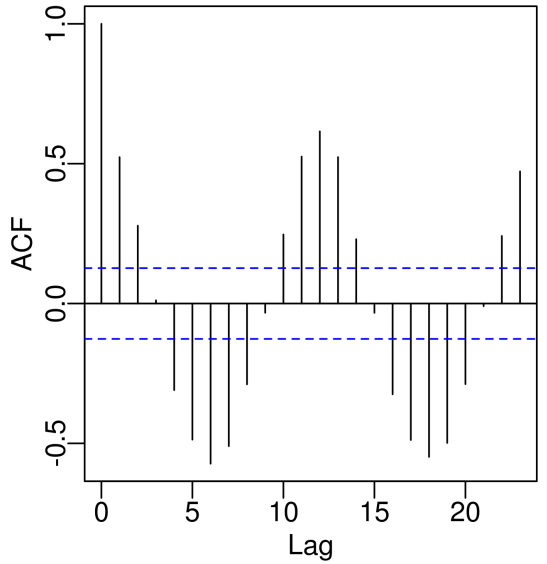
```
> d2 <- d[d$site == "site_01",]
> mod <- lm(y ~ time , data=d2)
> MEM.sel <- forward.sel(resid(mod), MEM$vectors, alpha=0.10)
Testing variable 1
Testing variable 2
Testing variable 3
(…)
Testing variable 127
Procedure stopped (R2more criteria): variable 127 explains only 0.000970 of the variance.

> MEM.sel
    variables order          R2     R2Cum  AdjR2Cum         F  pval
1         V36    36 0.131489916 0.1314899 0.1278407 36.032512 0.001
2         V94    94 0.073723454 0.2052134 0.1985063 21.983835 0.001
3         V77    77 0.056939807 0.2621532 0.2527738 18.212174 0.001
4         V95    95 0.056102897 0.3182561 0.3066519 19.338905 0.001
5         V37    37 0.042420295 0.3606764 0.3470156 15.526329 0.002
6         V24    24 0.028995636 0.3896720 0.3739554 11.069430 0.003
(…)
125       V88    88 0.001003369 0.9625276 0.9214395  3.052492 0.092
126      V195   195 0.001002857 0.9635305 0.9228654  3.107331 0.084

> MEM.id <- MEM.sel$order[MEM.sel$pval<=0.01] # N° des vecteurs les plus significatifs
```

Indépendance des résidus

Exemple : corrélation temporelle

dbMEM "distance based Moran's Eigenvector Maps"

```
> mod <- lm(y ~ time , data=d2)
> summary(mod)
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 8.007683   1.107529   7.230 6.54e-12 ***
time        0.011804   0.007968   1.481     0.14

Residual standard error: 8.552 on 238 degrees of freedom
Multiple R-squared:  0.009137,    Adjusted R-squared:  0.004974
F-statistic: 2.195 on 1 and 238 DF,  p-value: 0.1398

> acf(resid(mod))
```



Indépendance des résidus

Exemple : corrélation temporelle

dbMEM "distance based Moran's Eigenvector Maps"

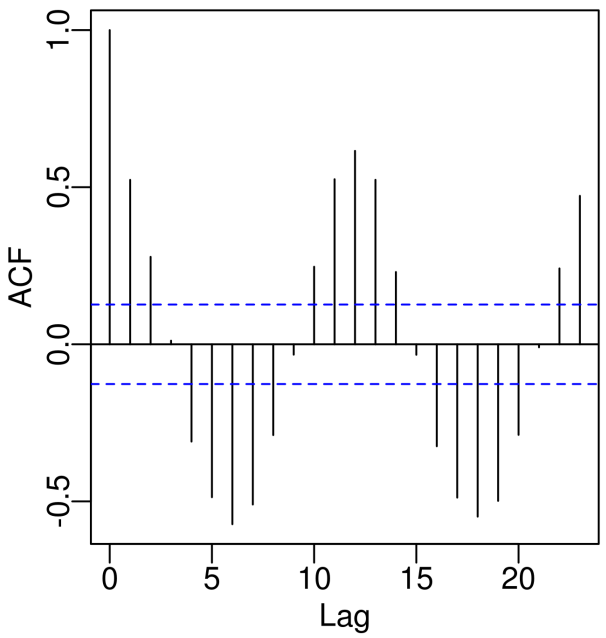
```
> MEMs <- MEM$vectors[,MEM.id]
> mod2 <- lm(y ~ time + MEMs[,1] + MEMs[,2] + MEMs[,3] + MEMs[,4] + MEMs[,5]
+ + MEMs[,6] + MEMs[,7] + MEMs[,8] + MEMs[,9] + MEMs[,10] + MEMs[,11], data=d2)
> summary(mod2)

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   8.018158   0.815251   9.835  < 2e-16 ***
time          0.011717   0.005865   1.998 0.046937 *
MEMs[, 1]    47.841724   6.295126   7.600 7.74e-13 ***
MEMs[, 2]   -35.823122   6.295126  -5.691 3.89e-08 ***
MEMs[, 3]   -31.482455   6.295126  -5.001 1.14e-06 ***
MEMs[, 4]    31.250232   6.295126   4.964 1.35e-06 ***
MEMs[, 5]   -27.173537   6.295128  -4.317 2.37e-05 ***
MEMs[, 6]   -22.466060   6.295126  -3.569 0.000438 ***
MEMs[, 7]    20.286587   6.295126   3.223 0.001457 **
MEMs[, 8]   -20.141059   6.295126  -3.199 0.001573 **
MEMs[, 9]   -18.440977   6.295189  -2.929 0.003743 **
MEMs[, 10]   16.246572   6.295127   2.581 0.010486 *
MEMs[, 11]  -14.387619   6.295134  -2.286 0.023206 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 6.295 on 227 degrees of freedom
Multiple R-squared:  0.4879,  Adjusted R-squared:  0.4609
F-statistic: 18.03 on 12 and 227 DF,  p-value: < 2.2e-16
```

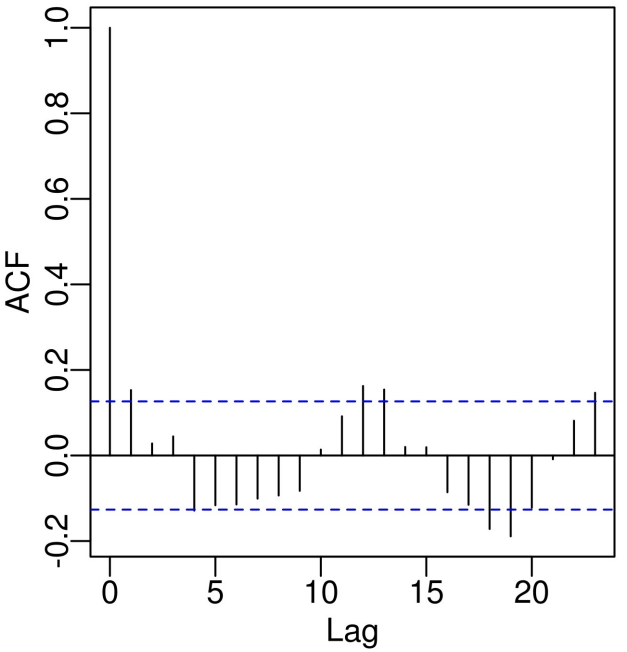
Indépendance des résidus

Exemple : corrélation temporelle

dbMEM "distance based Moran's Eigenvector Maps"

```
> acf(resid(mod))
> acf(resid(mod2))
```





Indépendance des résidus

Exemple : corrélation temporelle

dbMEM "distance based Moran's Eigenvector Maps"

Sélection des MEM qui expliquent une partie de la structure des résidus pour tous les sites. On répète les mêmes MEM pour les 16 sites et on refait la sélection sur base des résidus du modèle mixte.
NB on pourrait aussi faire la sélection site par site et utiliser l'ensemebles des MEM sélectionnés dans l'analyse globale

```
> MEM2 <- apply(MEM$vectors, 2, rep, times = 16)
> dim(MEM2)
[1] 3840  239
> mod <- lmer(y ~ time + (time | site), data=d)
> MEM.sel <- forward.sel(resid(mod), MEM2, alpha=0.10)
Testing variable 1
Testing variable 2
(...)
> MEM.sel
   variables order          R2     R2Cum  AdjR2Cum         F  pval
1        V36    36 0.112861025 0.1128610 0.1126299 488.26692 0.001
2        V94    94 0.065341962 0.1782030 0.1777746 305.08399 0.001
3        V77    77 0.045518676 0.2237217 0.2231146 224.93174 0.001

> MEM.id <- MEM.sel$order[MEM.sel$pval<=0.01] # N° des vecteurs les plus significatifs
```

Indépendance des résidus

Exemple : corrélation temporelle

dbMEM "distance based Moran's Eigenvector Maps"

```
> mod <- lmer(y ~ time  + (time | site), data=d)
> summary(mod)
Linear mixed model fit by REML ['lmerMod']
Formula: y ~ time + (time | site)
   Data: d

REML criterion at convergence: 27434.44

Random effects:
 Groups   Name        Variance  Std.Dev. Corr
 site     (Intercept) 9.416e+00 3.068524
          time        2.799e-06 0.001673 1.00
 Residual             7.294e+01 8.540726
Number of obs: 3840, groups: site, 16

Fixed effects:
             Estimate Std. Error t value
(Intercept) 10.368705   0.815445  12.715
time         0.013692   0.002033   6.736
```

Indépendance des résidus

Exemple : corrélation temporelle

dbMEM "distance based Moran's Eigenvector Maps"

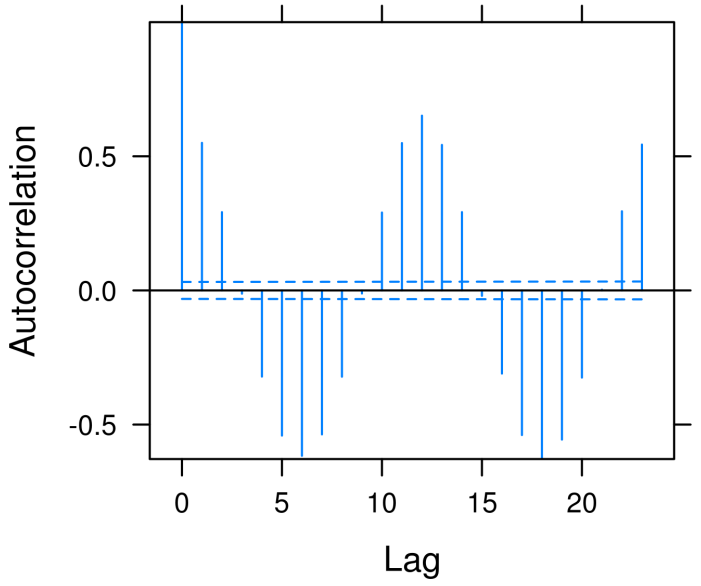
```
> MEMvec <- MEM2[,MEM.id]
> mod <- lmer(y ~ time + MEMvec[,1] + MEMvec[,2] + MEMvec[,3] + MEMvec[,4] + MEMvec[,5]
+              + MEMvec[,6] + MEMvec[,7] + MEMvec[,8] + MEMvec[,9] + MEMvec[,10]
+              + MEMvec[,11] + MEMvec[,12] + MEMvec[,13] + MEMvec[,14] + MEMvec[,15]
+              + MEMvec[,16] + MEMvec[,17] + MEMvec[,18] + MEMvec[,19] + MEMvec[,20]
              + (time | site), data=d)
> summary(mod)
Fixed effects:
               Estimate Std. Error t value
(Intercept)   10.380783   0.795671  13.047
time           0.013592   0.001424   9.547
MEMvec[, 1]   44.353966   1.459487  30.390
MEMvec[, 2]  -33.748661   1.459487 -23.124
MEMvec[, 3]  -28.167952   1.459487 -19.300
MEMvec[, 4]   26.879892   1.459487  18.417
MEMvec[, 5]  -26.767448   1.459487 -18.340
MEMvec[, 6]  -24.141668   1.459487 -16.541
MEMvec[, 7]  -23.231102   1.459487 -15.917
MEMvec[, 8]   22.959008   1.459487  15.731
MEMvec[, 9]  -16.986147   1.459487 -11.638
MEMvec[, 10] -16.647003   1.459510 -11.406
MEMvec[, 11] -16.083987   1.459498 -11.020
MEMvec[, 12] -14.822308   1.459497 -10.156
(...)
```

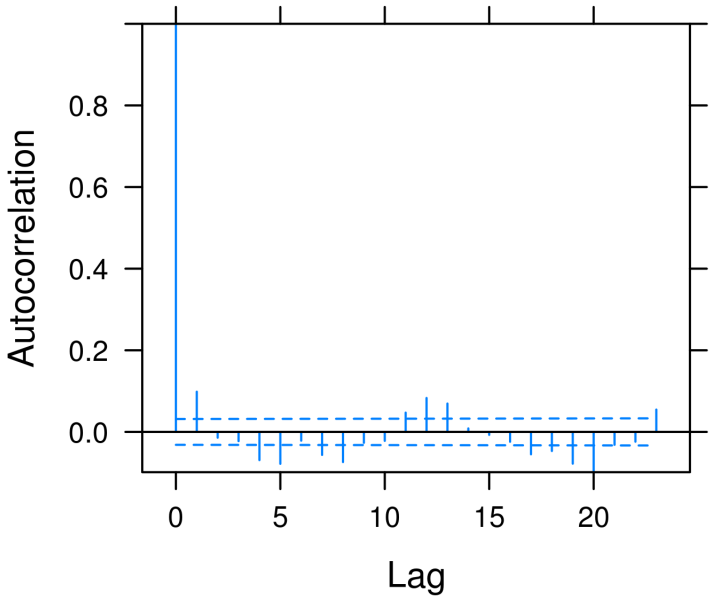
Indépendance des résidus

Exemple : corrélation temporelle

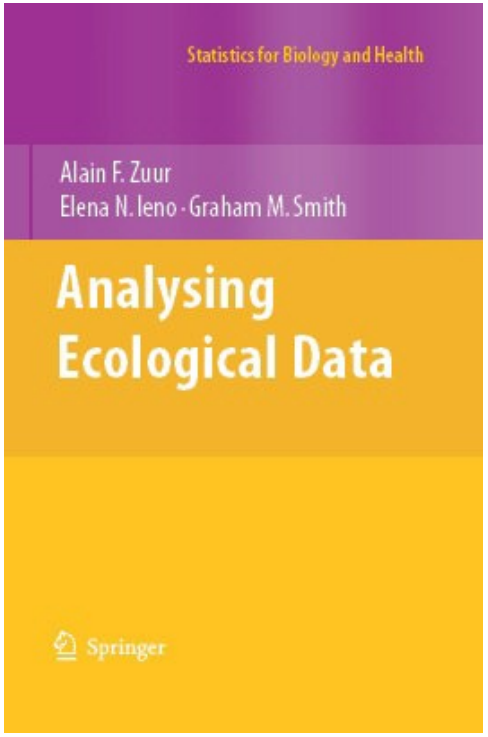
dbMEM "distance based Moran's Eigenvector Maps"

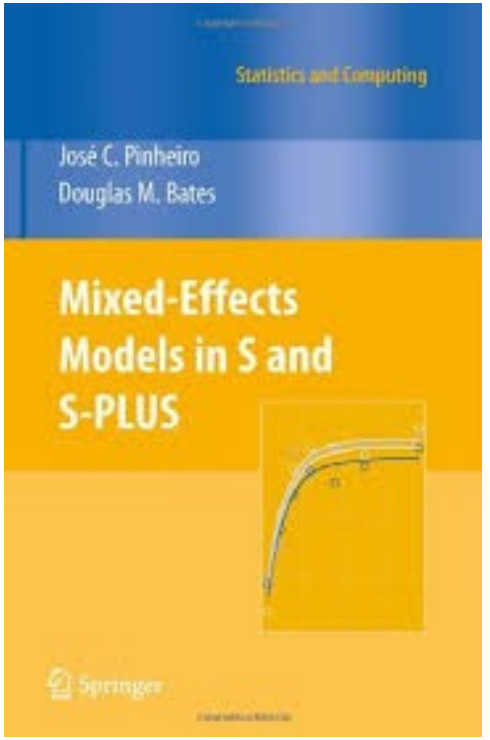
```
par(mfrow=c(1,1), mar = c(3,3,1,1), mgp = c(1.4, 0.4, 0))
modacf <- ACF.grouped(resid(mod, type="pearson"), d$time, d$site)
plot(modacf, alpha = 0.05)
```

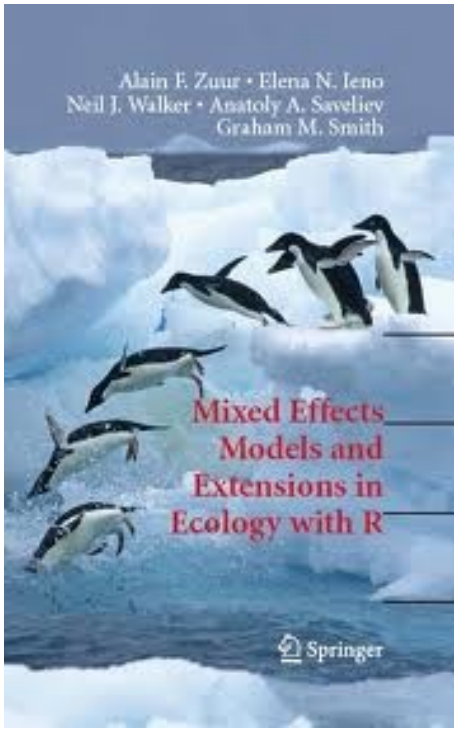




Quelques ressources utiles



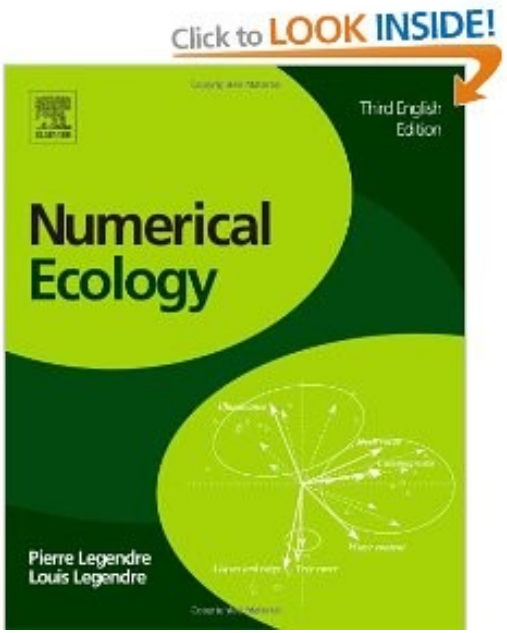


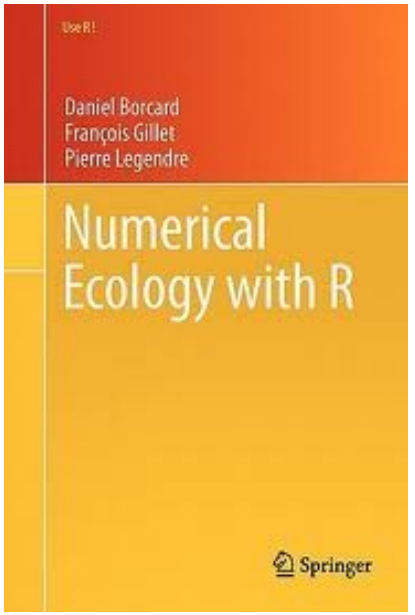


Vue d'ensemble de nombreuses méthodes disponibles pour l'analyse spatiale et temporelle (en plus du reste!)
Nombreux exemples sur des vrais jeux de données

En particulier pour les méthodes permettant d'inclure la corrélation spatiale et temporelle dans la matrice de variance covariance des résidus

Quelques ressources utiles





Toute la théorie (mais très accessible) + de nombreux exemples d'applications écologiques + nombreuses références + liste des fonctions R mais aucun code

Comment faire en pratique dans R ...

Deux livres très complémentaires et <u>particulièrement utiles et accessibles</u>
Traitent également en détail des méthodes multivariées pas abordées ici
NB : Legendre & Legendre 2012 contient de nombreuses nouveautés en particulier en terme d'analyse spatiale par rapport à l'édition de 2008