

# Analyse Spatiale

Jean Gaudart

Aix-Marseille Université

UMR 912, SESSTIM (AMU, INSERM, IRD)

# plan

1. Introduction
2. Classes spatiales
3. Visualisation
4. Interpolation et Géostatistique
5. Analyses de co-facteurs
6. Autocorrélation

# I. Introduction

Cartographie

⇒ Informations sur un phénomène géographique

⇒ Mais pas suffisant pour extraire ses caractéristiques et comprendre le processus sous-jacent

Inférence statistique indispensable:

⇒ « Pattern » spatial de l'incidence d'une maladie => existe-t-il une agrégation (cluster)?

⇒ En cas de clusters, sont-ils expliqués par des co-facteurs (âge, pauvreté, pollution, environnement...)?

⇒ En analysant des échantillons de sol, peut-on déterminer quelle zone est polluée?

⇒ En analysant des prélèvements d'air (pour mesurer sa qualité), comment estimer le nombre de personnes exposées à de haut niveau de particules fines? Et où vivent-elles?

⇒ Est-ce que les gouvernements ont tendance à appliquer des politiques semblables à leurs voisins ou bien sont-ils indépendants?

⇒...

Utiliser R pour l'analyse spatiale:

- ⇒ Nombre de packages +++ adaptés aux différentes hypothèses sur l'organisation spatiale des données
- ⇒ Package « sp » (Pebesma & Bivand, 2005)
- ⇒ Classes et méthodes spécifiques
- ⇒ Interfaces avec des Systèmes d'Informations Géographiques (SIG)

<http://cran.r-project.org/web/views/Spatial.html>

## Système d'informations Géographique (SIG/GIS):

⇒ Ensemble d'outils pour collecter, conserver, extraire à volonté, transformer, croiser, manipuler, analyser des données spatialisées issues du monde réel, afin de répondre un ensemble de questions

⇒ Base de données particulière: géographie

⇒ Ex: ArcGIS, QGIS, GRASS...

⇒ Nombreuses données accessibles gratuitement sur internet

- Christman N. (2002) *Exploring Geographic Information Systems*, Wiley.
- Heywood I, Cornelius S, Carver S. (2006) *An introduction to Geographic Information Systems*. Pearson Education.
- Longley PA, Goodchild MF, Maguire DJ, Rhind DW. (2005) *Geographic Information Systems and Science*. Wiley.

# Types de données spatiales

Données spatiales => références spatiales: Coordonnées dans un système de référence.

Exemple: localisation des volcans dans le monde. Coordonnées:

⇒ longitude degrés décimaux / méridien 0 (Greenwich)

⇒ Latitude degrés décimaux / 0 équateur

Système: World Geodetic System WGS84

Cartographier les volcans actifs 1980-2000:

⇒ Points (long,lat)

⇒ Carte aplatie => problème de projection:

transformer un système sphérique en un système plan

Par ex. projection de *Mollweide*

Données supplémentaires associées, non spatiales : *Attributs*

• *Points*: localisation ponctuelle, GPS, adresse géocodées

• *Lignes*: ensemble de points ordonnés, connectés par des segments de droites

• *Polygones*: zones définies par des lignes fermées (+- trous)

• *Grilles*: ensemble de points ou de cellules rectangulaires organisés en un treillis régulier

Vectoriel:  
représentation la plus  
précise et fiable possible

Raster:  
Représentation de surfaces  
continues par une tessalation  
régulière

Attributs d'une cellule: moyenne, valeur centrale, %, ...

# II. Classe pour données spatiales -R

```
chooseCRANmirror(ind=6)
install.packages("sp")
library(sp)
```

## II.1 Objet spatial

La classe spatial de base comprend 2 slots (parties):

- Un cadre délimité (bounding box `bbox`): matrice de coordonnées numérique, colonnes (min, max), lignes (Est=x, Nord=Y)
- Système de projection (`proj4string`): CRS-class coordinate reference system

```
getClass("Spatial")
```

```
Class "Spatial" [package "sp" ]
Slots:
Name:      bbox proj4string
Class:     matrix          CRS
Known Subclasses:
Class "SpatialPoints", directly
Class "SpatialGrid", directly
Class "SpatialLines", directly
Class "SpatialPolygons", directly
...
```



## Construction d'un objet Spatial simple:

```
m1<-matrix(c(0,0,1,1),nc=2,nr=2,dimnames=list(NULL,c("min","max")))
crs1<-CRS(projargs=as.character(NA))
SpOb1<-Spatial(bbox=m1, proj4string=crs1)
```

```
SpOb1
      An object of class "Spatial"
      Slot "bbox":
            min max
      [1,]    0    1
      [2,]    0    1
      Slot "proj4string":
      CRS arguments: NA
```

```
m2<-matrix(c(350,85,360,90),nc=2,nr=2,dimnames=list(NULL,c("min","max")))
crs2<-CRS("+proj=longlat +datum=WGS84")
SpOb2<-Spatial(bbox=m2, proj4string=crs2)
```

```
SpOb2
      An object of class "Spatial"
      Slot "bbox":
            min max
      [1,] 350 360
      [2,]  85  90
      Slot "proj4string":
      CRS arguments:
      +proj=longlat +datum=WGS84 +ellps=WGS84 +towgs84=0,0,0
```

## II.2 SpatialPoints

Sous-classe de la classe Spatial

Un point est précisément localisé sur le globe par ses coordonnées (x,y), mesurées en degrés

y=90° à -90° Nord->Sud

X=0° à 360° ou 180 à -180 Est->Ouest

Ex. localisation de CRANmirrors

```
CRANdf<-read.table("CRAN051001a.txt",header=TRUE)
CRAN_mat<-cbind(CRANdf$long,CRANdf$lat)
row.names(CRAN_mat)<-1:nrow(CRAN_mat)
crs3<-CRS("+proj=longlat +ellps=WGS84")
CRAN_sp<-SpatialPoints(CRAN_mat,proj4string=crs3)
summary(CRAN_sp)
```

Object of class SpatialPoints

Coordinates:

	min	max
coords.x1	-122.95000	153.0333
coords.x2	-37.81667	57.0500

Is projected: FALSE

proj4string : [+proj=longlat +ellps=WGS84]

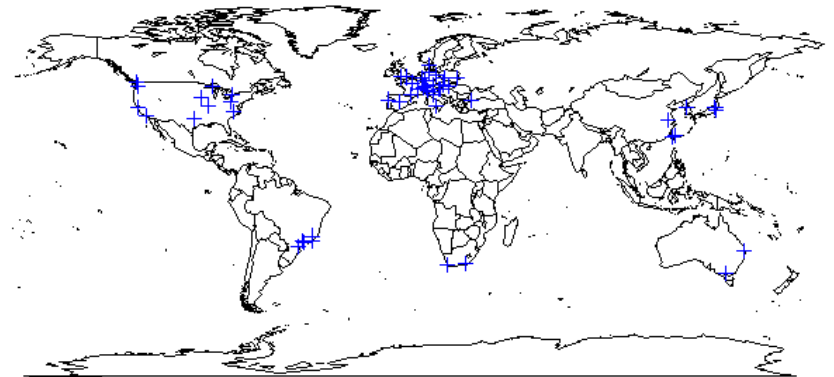
Number of points: 54

Pour retrouver les caractéristique d'un objet Spatial:

```
bbox(CRAN_sp)  
proj4string(CRAN_sp)  
  
plot(CRAN_sp)
```

```
min      max  
coords.x1 -122.95000 153.0333  
coords.x2  -37.81667  57.0500  
[1] "+proj=longlat +ellps=WGS84"
```

```
wrld<-map("world")  
plot(CRAN_sp,add=T,col="blue")
```



## Extraire les coordonnées d'un point particulier

```
bresil<-which(CRANdf$loc=="Brazil")
bresil
coordinates(CRAN_sp)[bresil,]
      coords.x1 coords.x2
4 -49.26667 -25.41667
5 -42.86667 -20.75000
6 -43.20000 -22.90000
7 -47.63333 -22.71667
8 -46.63333 -23.53333

summary(CRAN_sp[bresil,])
  Object of class SpatialPoints
Coordinates:
      min      max
coords.x1 -49.26667 -42.86667
coords.x2 -25.41667 -20.75000
Is projected: FALSE
proj4string : [+proj=longlat +ellps=WGS84]
Number of points: 5
```

## II.3 SpatialLines

```
getClass("Line")
Class "Line" [package "sp"]
Slots:
Name:   coords
Class:  matrix
Known Subclasses: "Polygon"

getClass("SpatialLines")
Class "SpatialLines" [package "sp"]
Slots:
Name:   lines      bbox proj4string
Class:  list       matrix      CRS
Extends: "Spatial", "SpatialLinesNULL"
Known Subclasses: "SpatialLinesDataFrame"
```

### Exemple Japon

```
library(maps)
Japan<-map("world","japan",plot=FALSE)
proj<-CRS("+proj=longlat +ellps=WGS84")
Sp_japan<-map2SpatialLines(japan,proj4string=proj)

plot(Sp_japan)
```

## II.4 SpatialPolygons

```
getClass("Polygon")
```

```
Class "Polygon" [package "sp"]  
Slots:  
Name:      labpt      area      hole ringDir  coords  
Class:     numeric   numeric logical integer  matrix  
Extends:   "Line"
```

```
getClass("SpatialPolygons")
```

```
Class "SpatialPolygons" [package "sp"]  
Slots:  
Name:      polygons  plotOrder      bbox proj4string  
Class:      list      integer      matrix      CRS  
Extends:    "Spatial", "SpatialPolygonsNULL"  
Known Subclasses: "SpatialPolygonsDataFrame"
```

## II.5 SpatialGrid

```
getClass("GridTopology")
```

```
Class "GridTopology" [package "sp"]
```

```
Slots:
```

```
Name:  cellcentre.offset      cellsize      cells.dim
Class:          numeric          numeric          integer
```

```
getClass("SpatialGrid")
```

```
Class "SpatialGrid" [package "sp"]
```

```
Slots:
```

```
Name:          grid          bbox  proj4string
Class: GridTopology          matrix          CRS
Extends: "Spatial"
Known Subclasses: "SpatialGridDataFrame"
```

# III. Visualiser des données spatiales

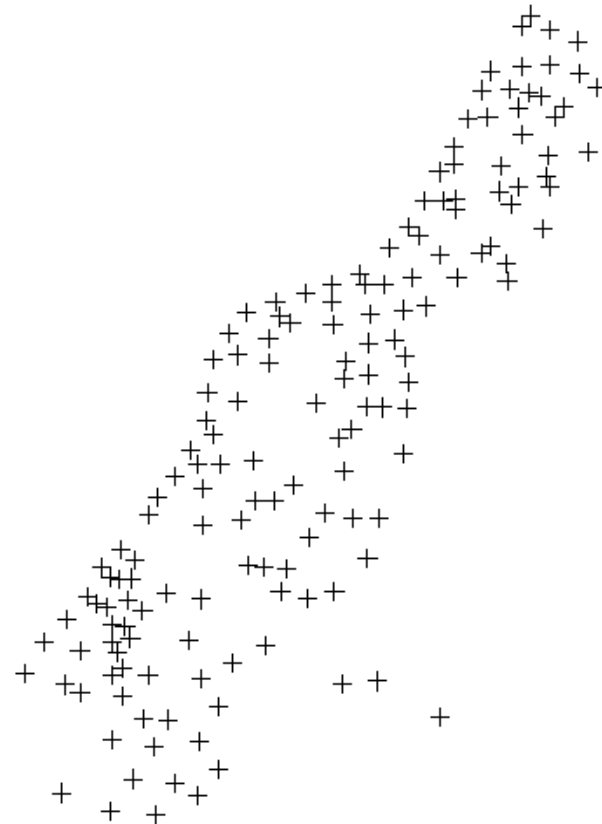
Commandes `plot` et `image`

Exemple: bassin de la Meuse (pollutions par les métaux lourds)

Points

```
library(sp)
data(meuse)
coordinates(meuse) <- c("x", "y")
plot(meuse)
title("Points")
```

Points

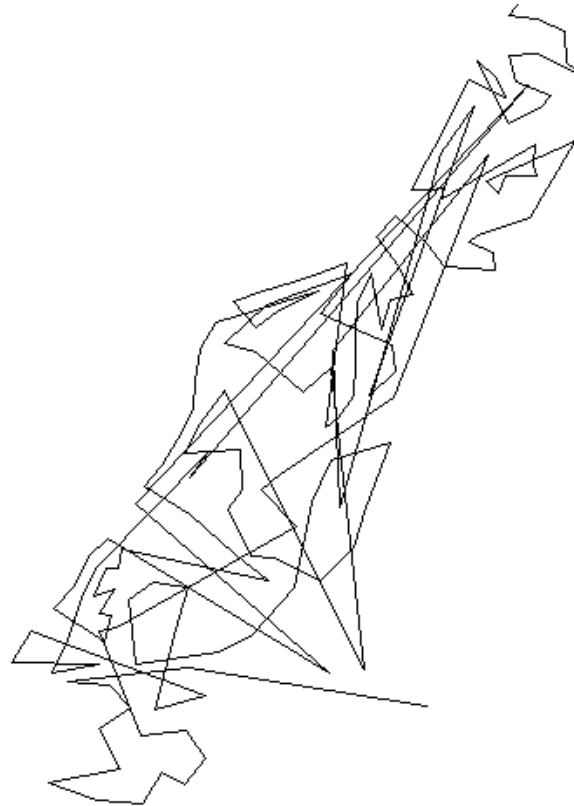




## Lignes

```
coord.meuse<-coordinates(meuse)
list.l.meuse<-list(Lines(list(Line(coord.meuse)), "1"))
lines.meuse<-SpatialLines(list.l.meuse)
plot(lines.meuse)
title("Lines")
```

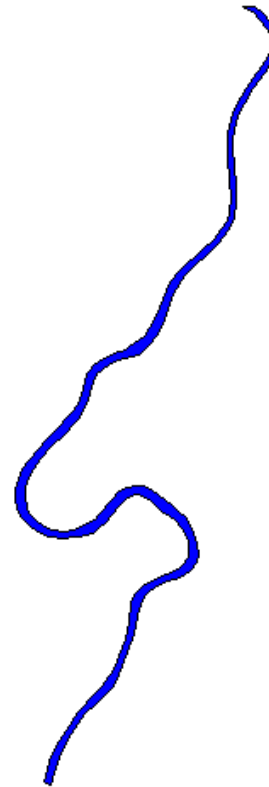
**Lines**



## Polygones

```
data(meuse.riv)
list.p.meuse<-list(Polygons(list(Polygon(meuse.riv)), "meuse.riv"))
pol.meuse<-SpatialPolygons(list.p.meuse)
plot(pol.meuse,col="blue")
title("Polygones")
```

**Polygones**



## Grilles

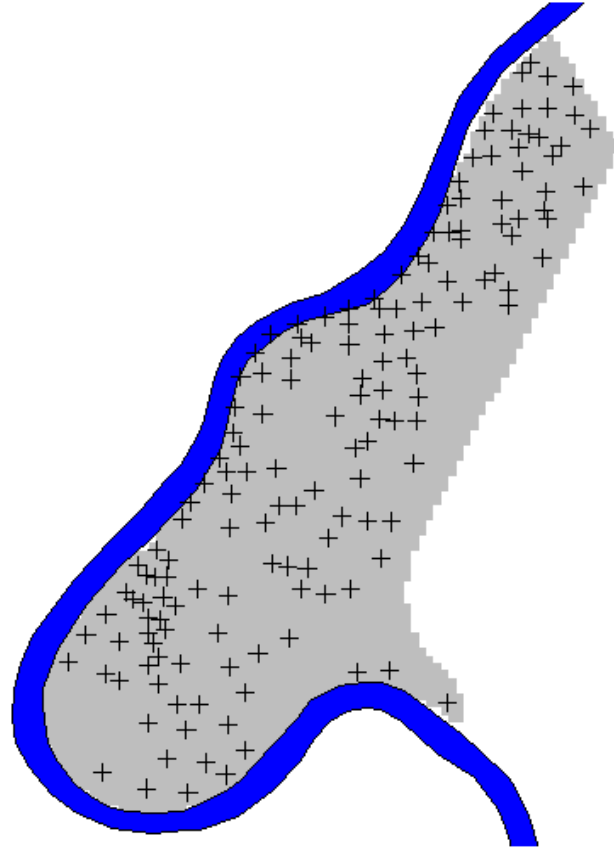
```
data(meuse.grid)  
coordinates(meuse.grid) <- c("x", "y")  
meuse.grid <- as(meuse.grid, "SpatialPixels")  
image(meuse.grid, col="grey")  
title("Grille")
```

**Grille**



## Combinaison des éléments

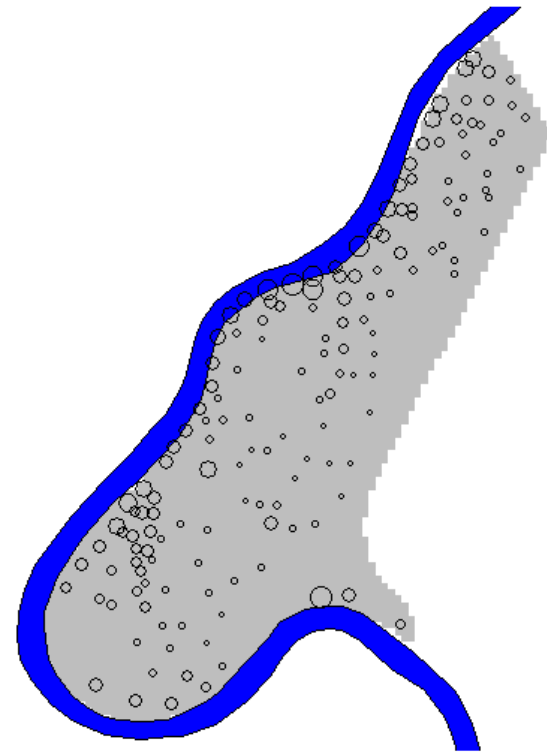
```
image(meuse.grid, col="grey")  
plot(pol.meuse, col="blue", add=TRUE)  
plot(meuse, add=TRUE)
```



## Combinaison des éléments, avec un attribut

### Ex. contamination par le zinc

```
image(meuse.grid, col="grey")  
plot(pol.meuse, col="blue", add=TRUE)  
plot(meuse, pch=1, cex=sqrt(meuse$zinc)/20, add=TRUE)
```



# IV. Interpolation et Géostatistique

Observations en un nombre limité de localisations=> échantillons spatiaux

Ex: particules de l'air, teneur aurifère ...

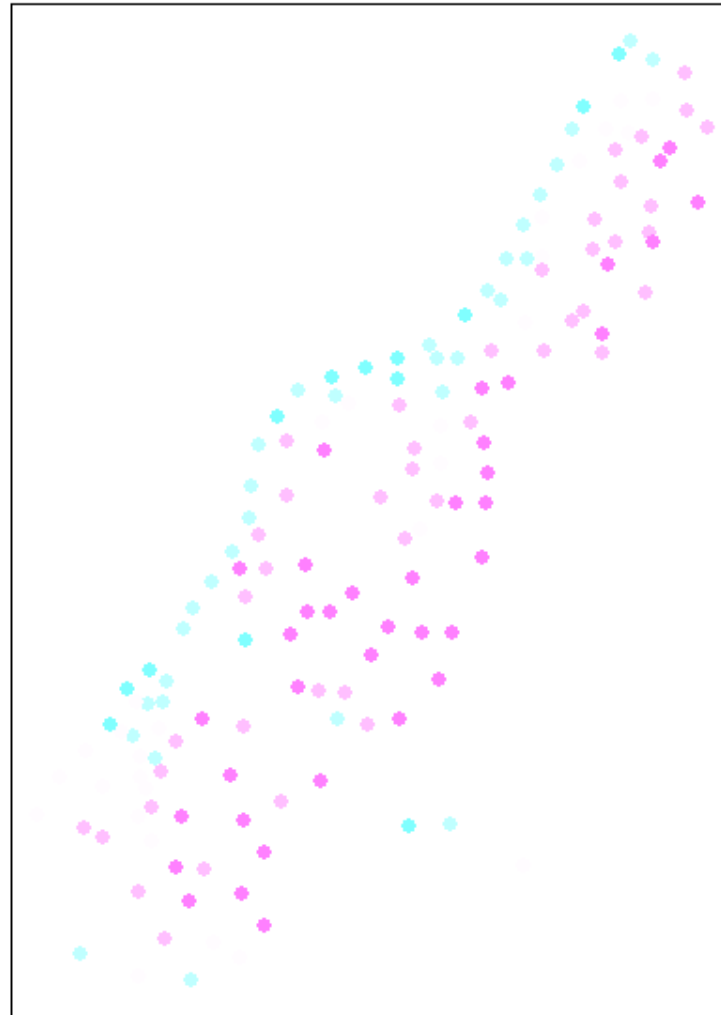
Objectif: connaître la distribution spatiale de la variable, à partir d'échantillons spatiaux

Géostatistique: analyse de champs aléatoires  $Z(s)$ , avec  $Z$  une VA aléatoire sur une localisation  $s$  non aléatoire.

Estimer et prédire  $Z(s)$  sur une région définie, en tenant compte de l'autocorrélation spatiale

## Exemple contamination par le zinc, Meuse

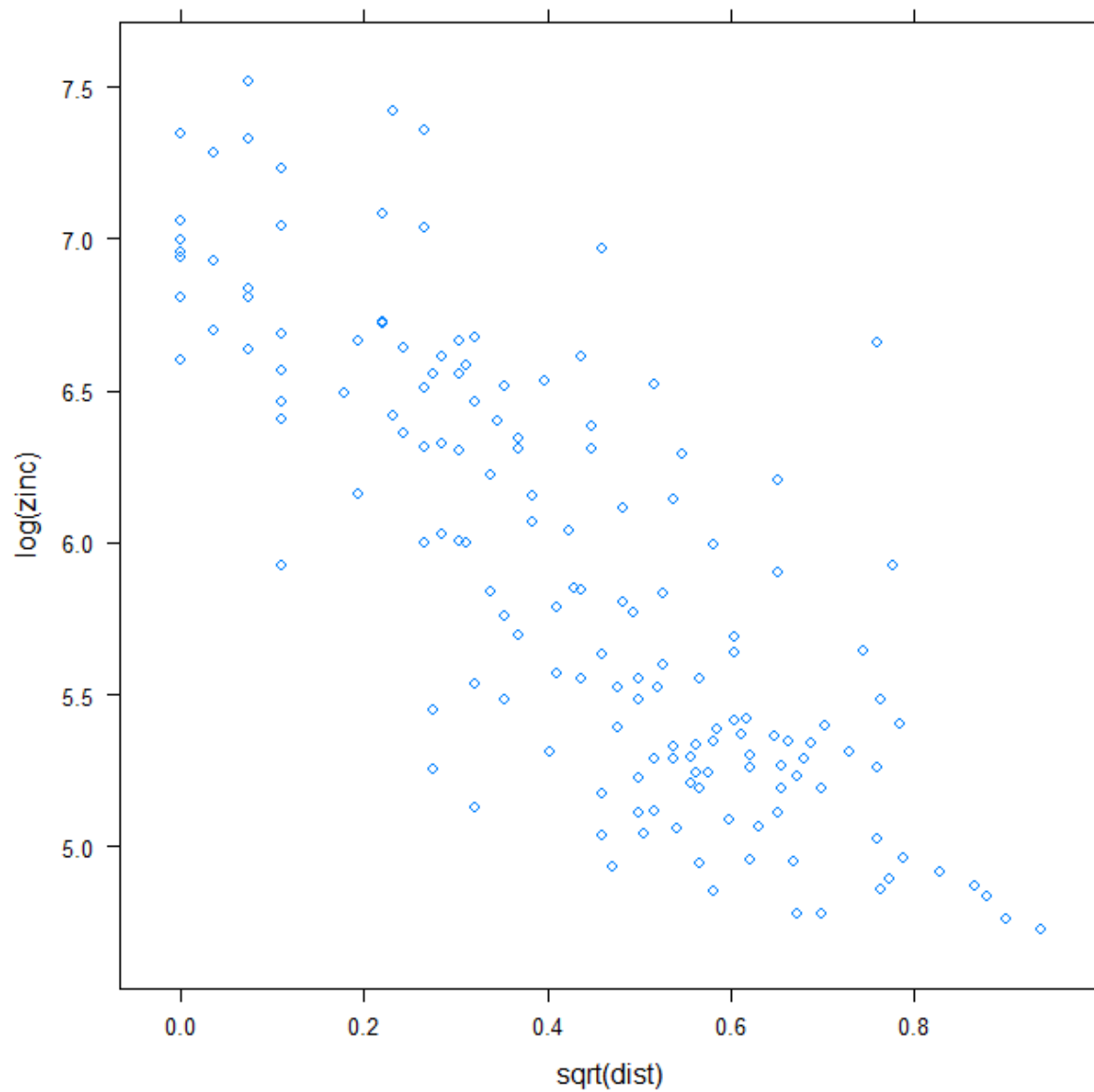
```
spplot(meuse, "zinc", do.log=TRUE, add=T)
```



(c)2013, Jean Gaudart (ANRS-MIAS) Univ, UMR912

## Contamination fonction de la distance à la Meuse

```
xypplot(log(zinc)~sqrt(dist), as.data.frame(meuse))
```

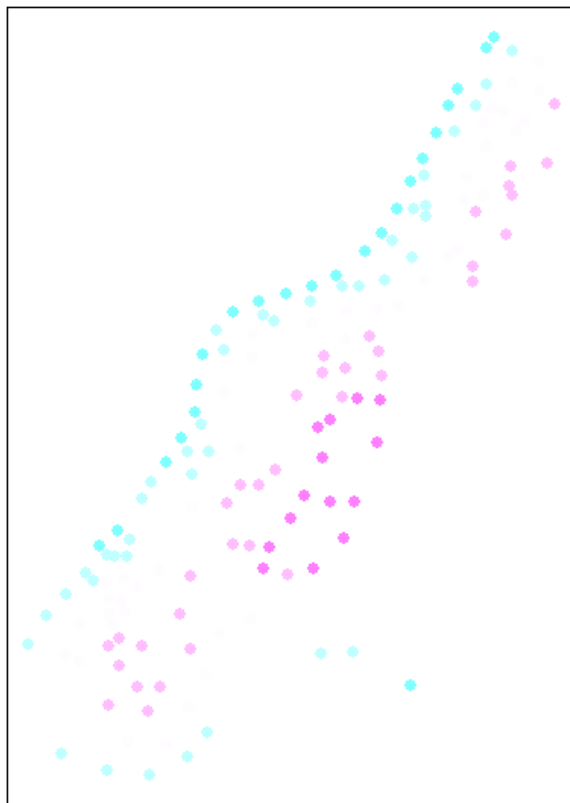




## Contamination fonction de la distance à la Meuse : modèle linéaire

```
zn.lm <- lm(log(zinc)~sqrt(dist), meuse)
meuse$estim<-predict(zn.lm, meuse)
meuse$res <- zn.lm$res
spplot(meuse, "estim",main="Estimations")
spplot(meuse, "res",main="Résidus")
```

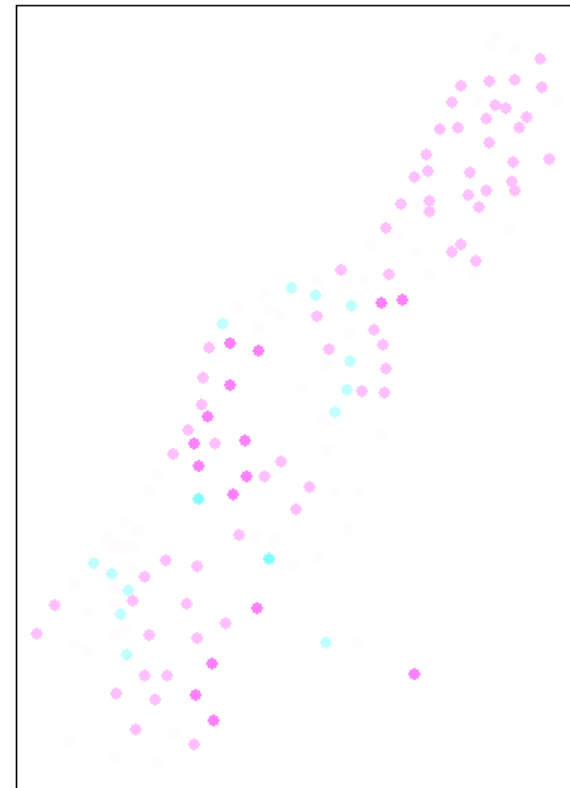
**Estimations**



● [4.602,5.081]  
● (5.081,5.559]  
● (5.559,6.038]  
● (6.038,6.516]  
● (6.516,6.994]

(c)2013, Jean Gaudart

**Résidus**



● [-1.046,-0.5176]  
● (-0.5176,0.011]  
● (0.011,0.5396]  
● (0.5396,1.068]  
● (1.068,1.597]

## Interpolation:

### 1/création de la surface : grille d'interpolation

```
data(meuse.grid)  
coordinates(meuse.grid) <- c("x", "y")  
meuse.grid <- as(meuse.grid, "SpatialPixelsDataFrame")
```

### 2/Choix de la méthode d'interpolation:

#### Méthodes non-géostatistiques

- Inverse Distance Weighted Interpolation
- Régression linéaire

#### Méthodes géostatistiques:

- Variogramme
- Modèle additif

## IV.1 Inverse Distance Weighted Interpolation:

Prédiction par une moyenne pondérée de la distance

$$\hat{Z}(s_0) = \frac{\sum_{i=1}^n w(s_i) Z(s_i)}{\sum_{i=1}^n w(s_i)}$$

Poids:

$$w(s_i) = \frac{1}{d(s_i - s_0)^p}$$

← distance

$p$ : paramètre de puissance.

Le choix de  $p$ , fonction

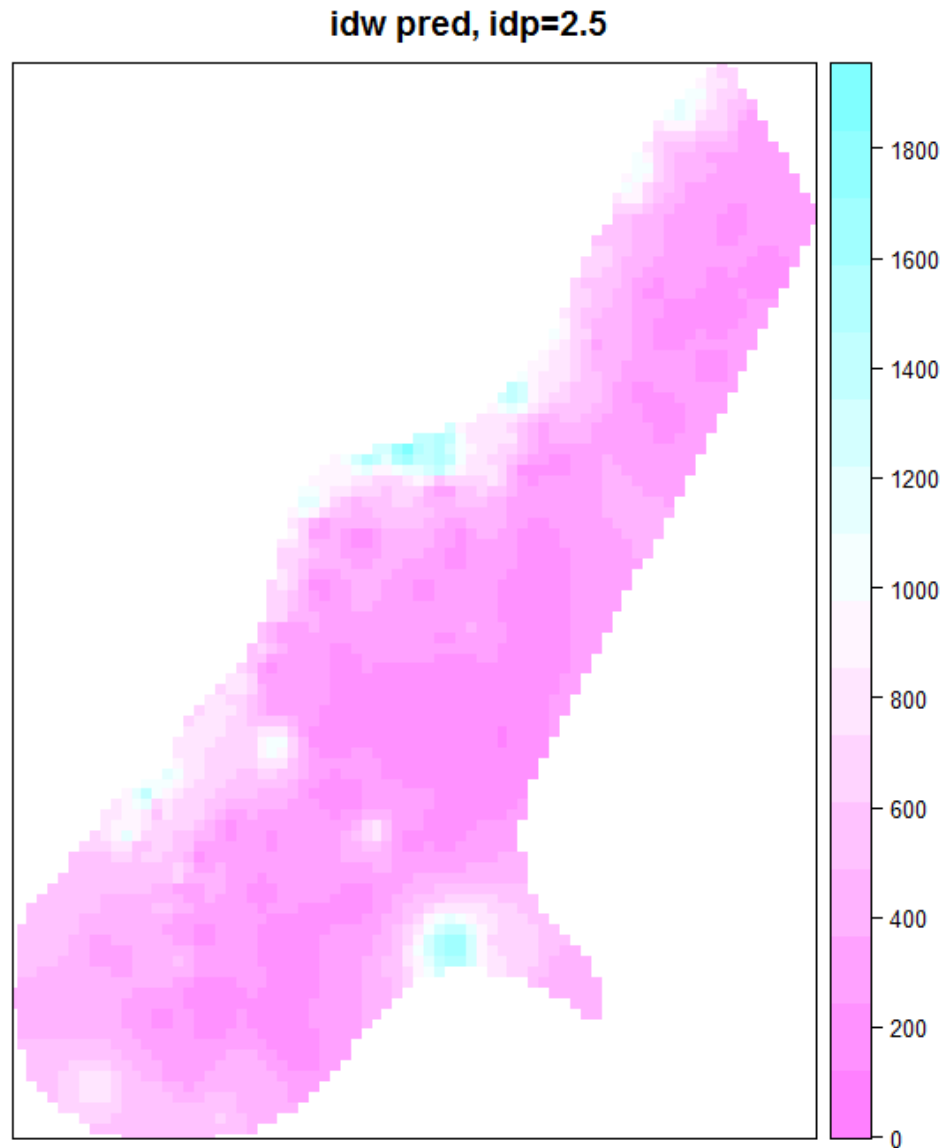
- degré de lissage désiré pour l'interpolation,
- densité et distribution des échantillons interpolés,
- distance maximum d'influence des points environnants.

grandes valeurs => influence++ valeurs proches du point interpolé.

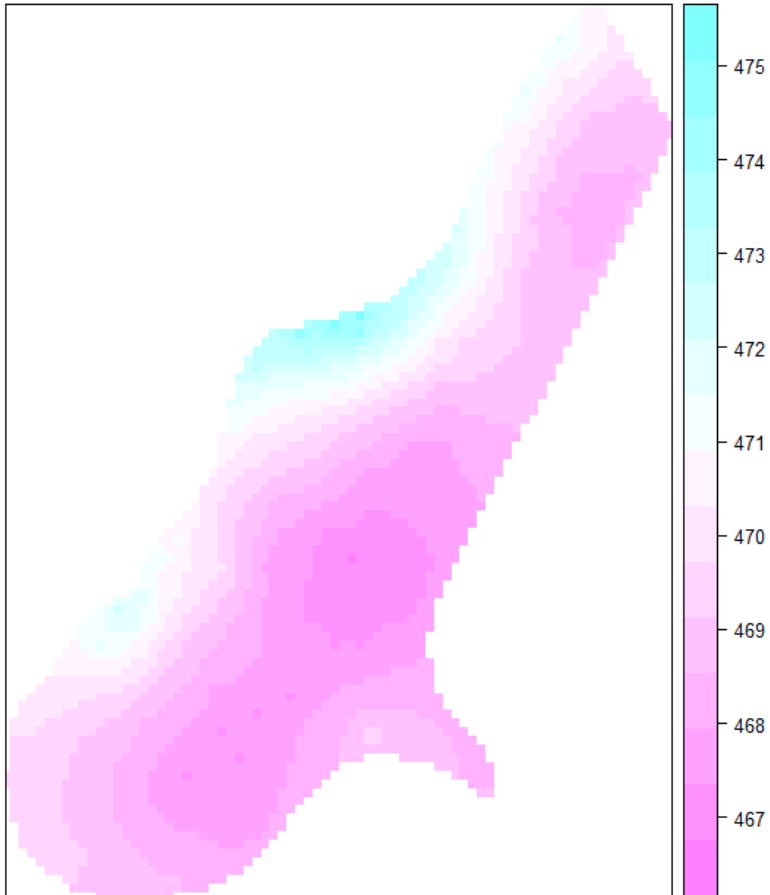
Pour  $p > 1$  le pic plus pointu au-dessus du point interpolé,

Pour  $0 < p < 1$  => pics lissés au-dessus du point interpolé.

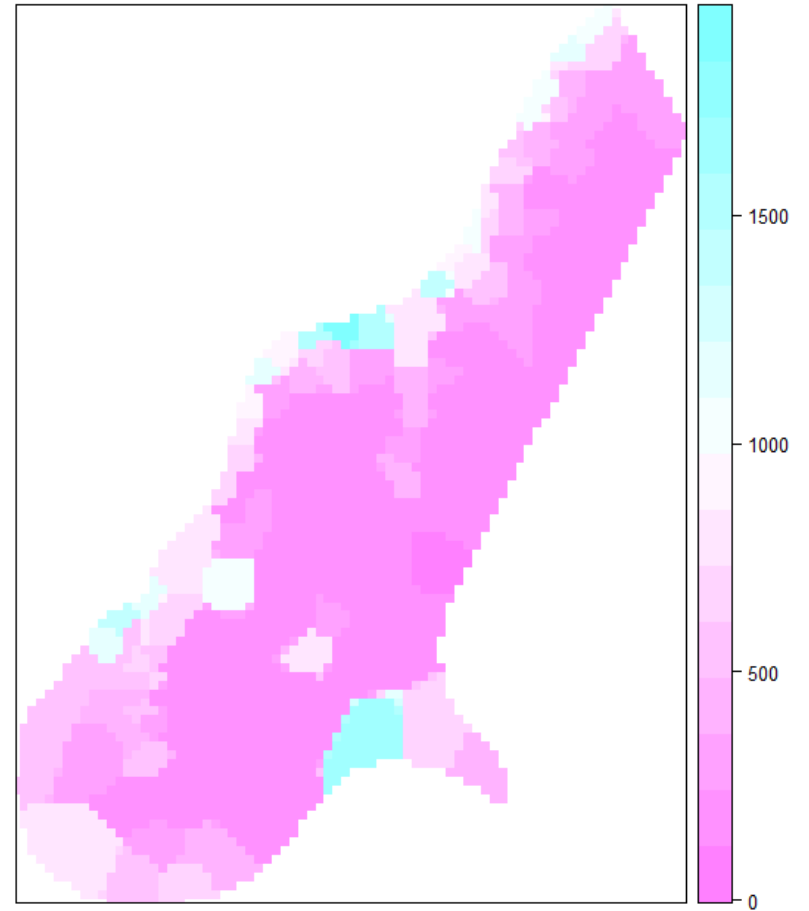
```
idw.meuse<-idw(zinc~1,meuse,meuse.grid,idp=2.5)  
spplot(idw.meuse,"var1.pred",main="idw pred, idp=2.5")
```



idw pred, idp=0.05



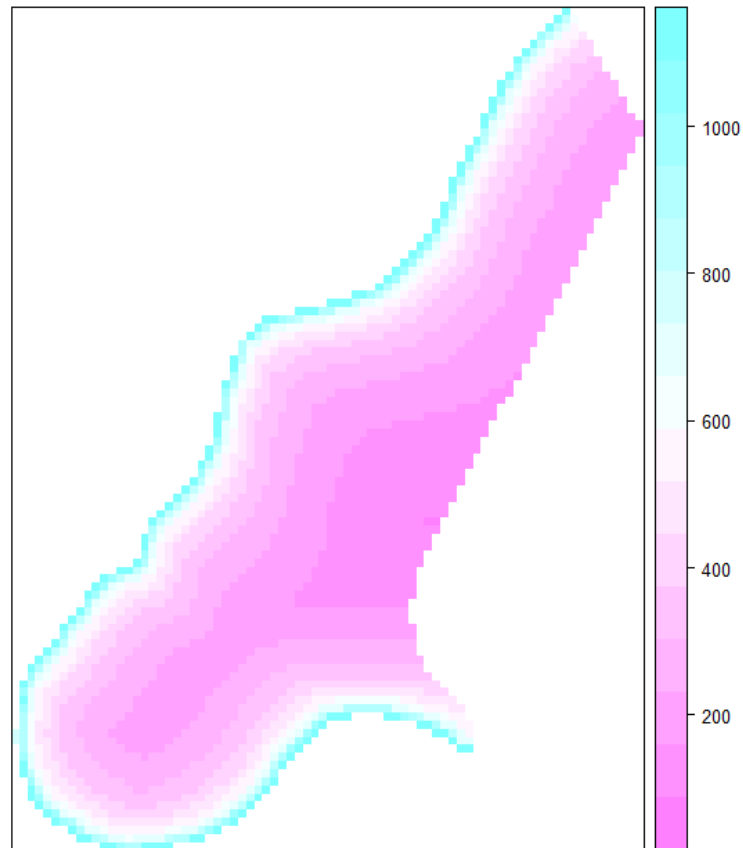
idw pred, idp=50



## IV.2 Interpolation par Régression linéaire:

$$\hat{Z}(s_0) = \alpha + \beta \sqrt{d(s_0 - s_i)} + \varepsilon$$

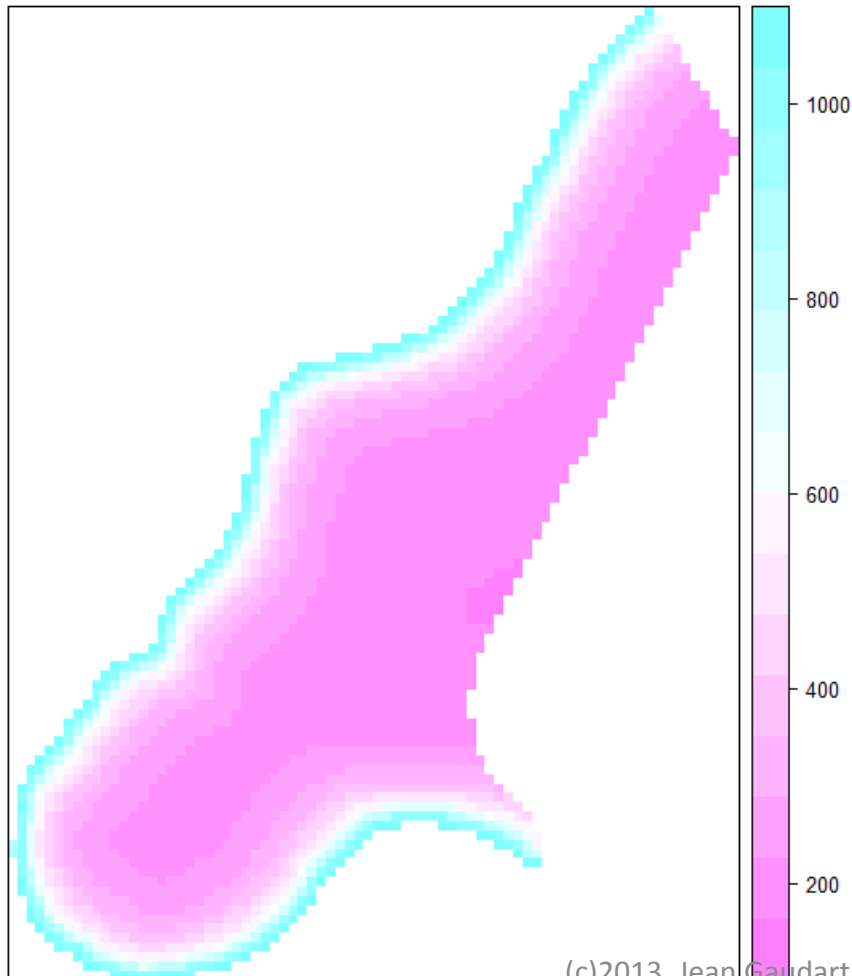
```
zn.lm <- lm(log(zinc)~sqrt(dist), meuse)
meuse.grid$pred <- exp(predict(zn.lm, meuse.grid))
spplot(meuse.grid, "pred", main="reg lin")
```



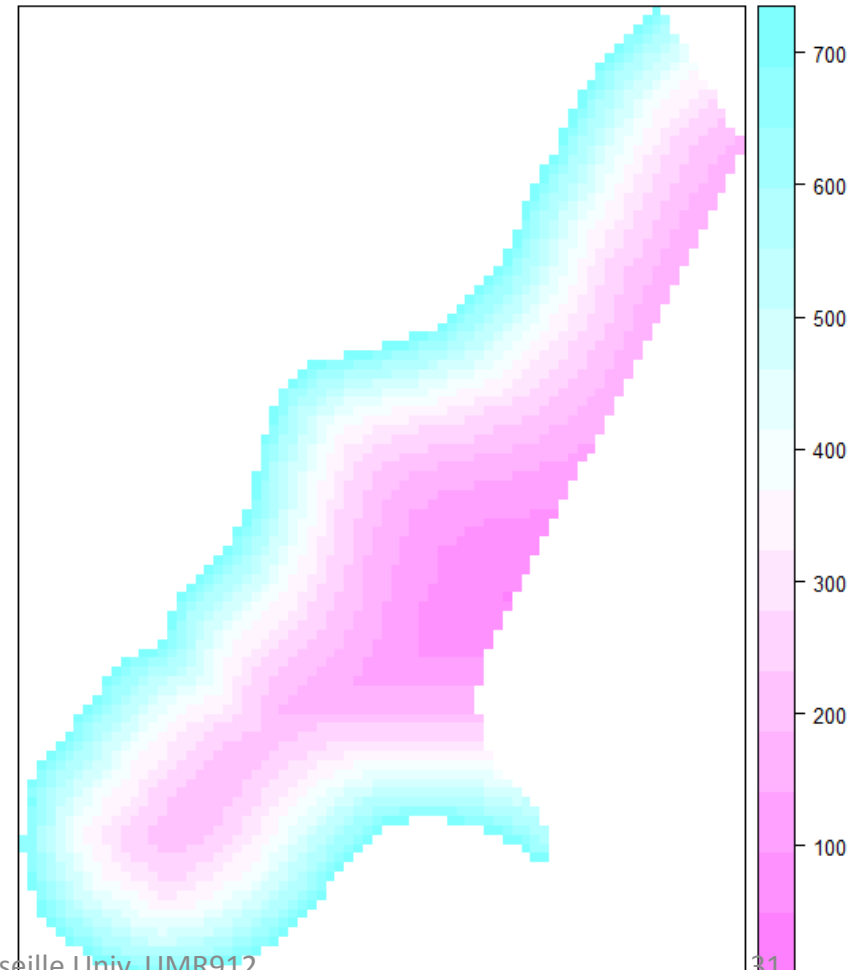
$$\hat{Z}(s_0) = \alpha + \beta \exp\left\{\frac{-d(s_0 - s_i)}{\delta}\right\} + \varepsilon$$

```
zn.lm2 <- lm(log(zinc)~exp(-dist/0.2), meuse)
meuse.grid$pred2 <- exp(predict(zn.lm2, meuse.grid))
spplot(meuse.grid,"pred2",main="reg lin, exp,0.2")
```

reg lin, exp,0.2



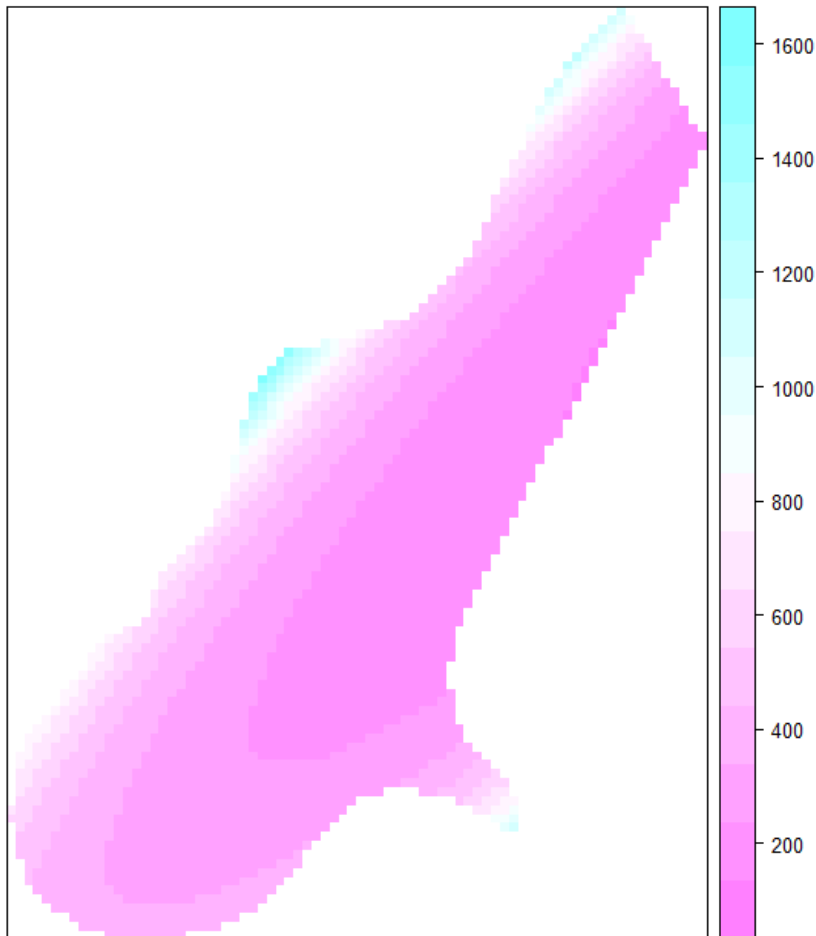
reg lin, exp,50



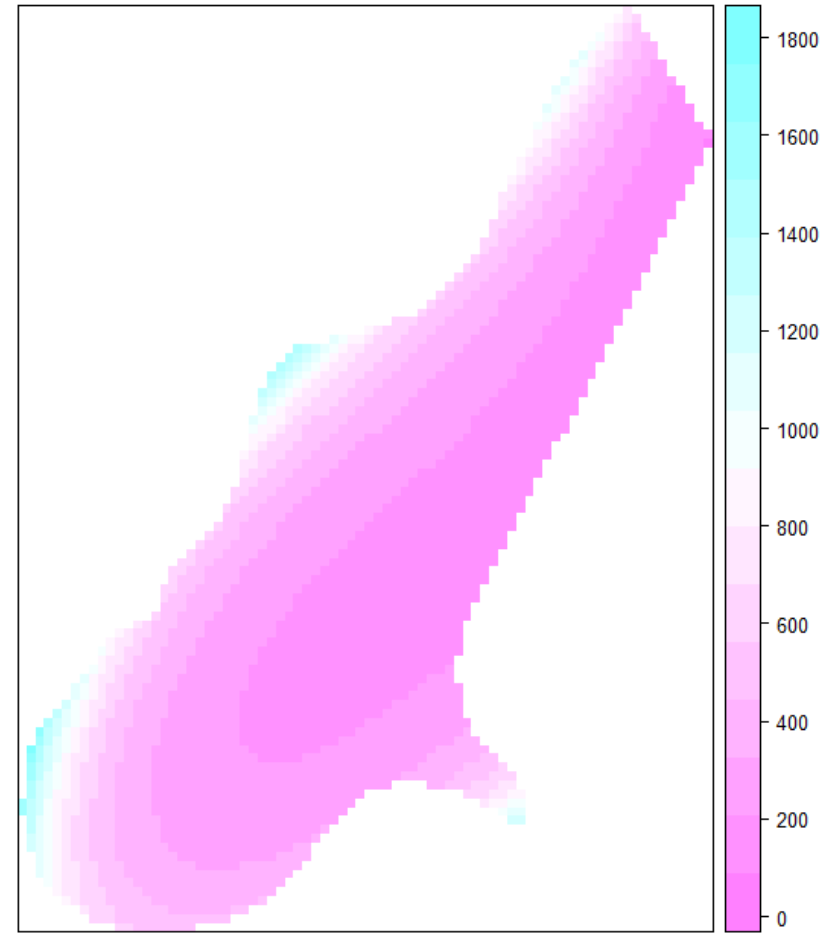
$$\hat{Z}(s_0) = \alpha + \beta_1 x + \beta_2 y + \beta_3 x^2 + \beta_4 y^2 + \beta_5 x \times y + \varepsilon$$

```
zn.lm3 <- lm(log(zinc)~x+y+I(x^2)+I(y^2)+I(x*y), meuse)  
meuse.grid$pred3 <- exp(predict(zn.lm3, meuse.grid))  
spplot(meuse.grid, "pred3", main="reg lin, polynome2")
```

reg lin, polynome2



reg lin, polynome4





## IV.3 Variogramme

⇒ Corrélation entre 2 observations à une distance donnée

Rappel: série temporelle, plusieurs observations espacées d'un lag  $k$

Ex:  $t=0 \leftrightarrow t=k$ ;  $t=10 \leftrightarrow t=10+k \dots$

Mais données spatiales, ce n'est pas toujours le cas: une distance de  $h$  peut être observée une seule fois.

⇒ Hypothèse de stationnarité obligatoire

$$Z(s) = m + \varepsilon(s)$$

$$E[Z(s)] = m = \text{cst } \forall s$$

Variogramme (ou semi-variogramme)

$$\gamma(h) = \frac{1}{2} E[Z(s) - Z(s+h)]^2$$

⇒ Variance constante et variogramme indépendant de  $s$

Si on suppose que le variogramme est indépendant de la direction (isotropie),  
Et si on regroupe des lags similaires =>  $N_h$  paires

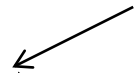
Estimation du variogramme

$$\hat{\gamma}(\tilde{h}_j) = \frac{1}{2N_h} \sum_{i=1}^{N_h} [Z(s) - Z(s+h)]^2 \quad \forall h \in \tilde{h}_j$$

Le modèle linéaire devient

$$Z(s) = \alpha + \beta X + \varepsilon(s)$$

Stationnaire,  
Variogramme à modéliser

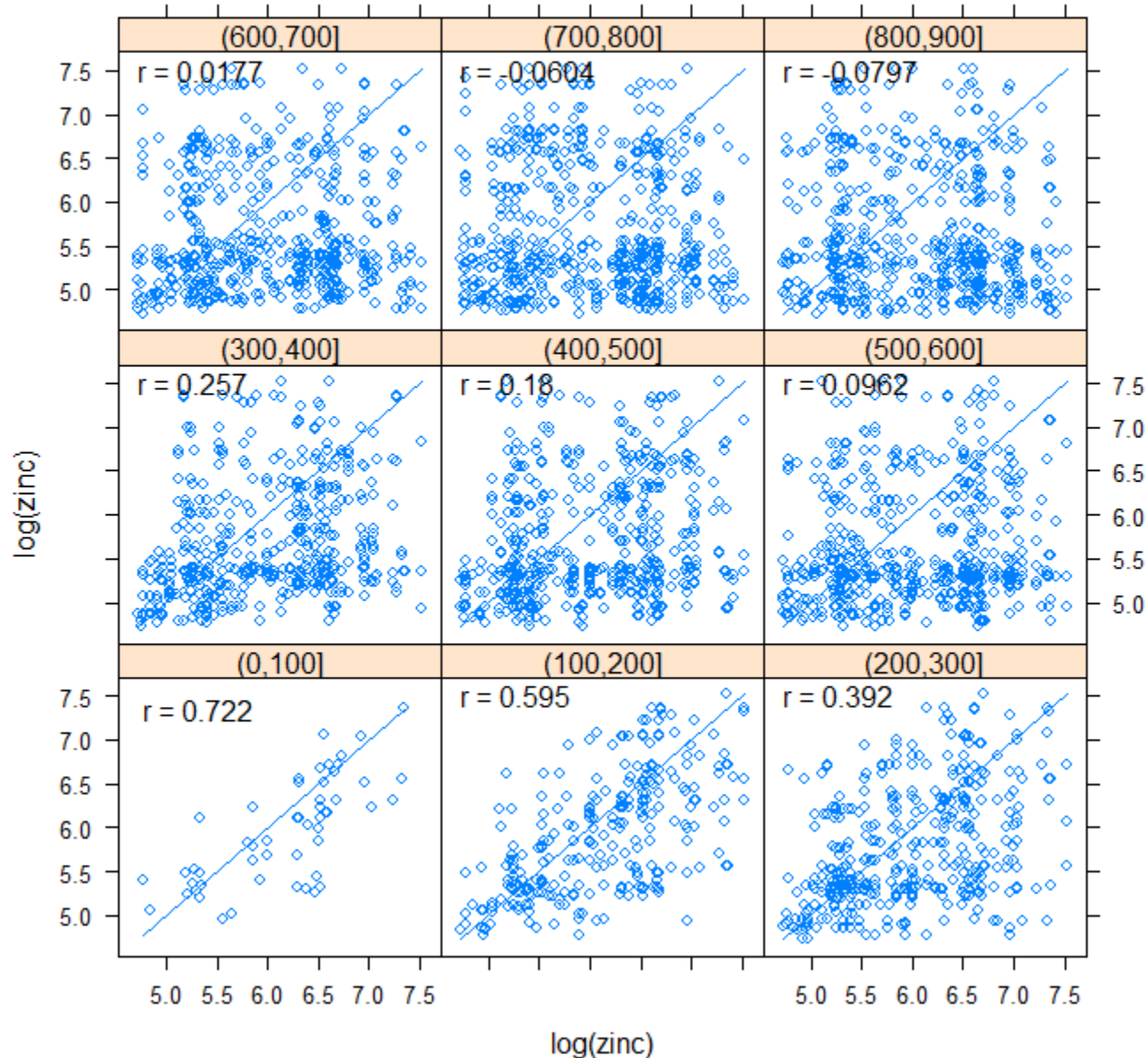


# Exploration du variogramme

## lagged scatterplots

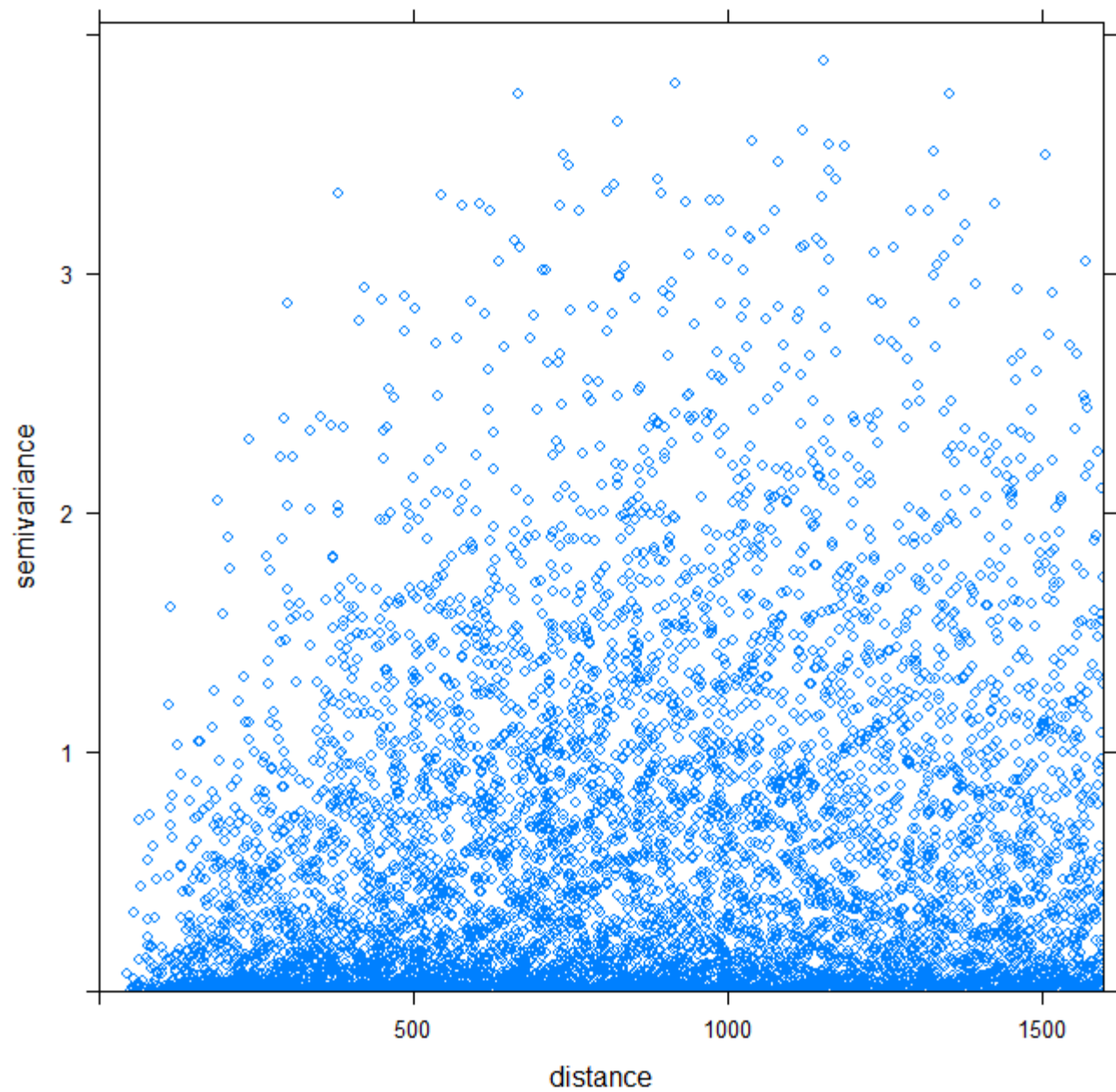
`hscat(log(zinc)~1,meuse,(0:9)*100)`

5.0 5.5 6.0 6.5 7.0 7.5



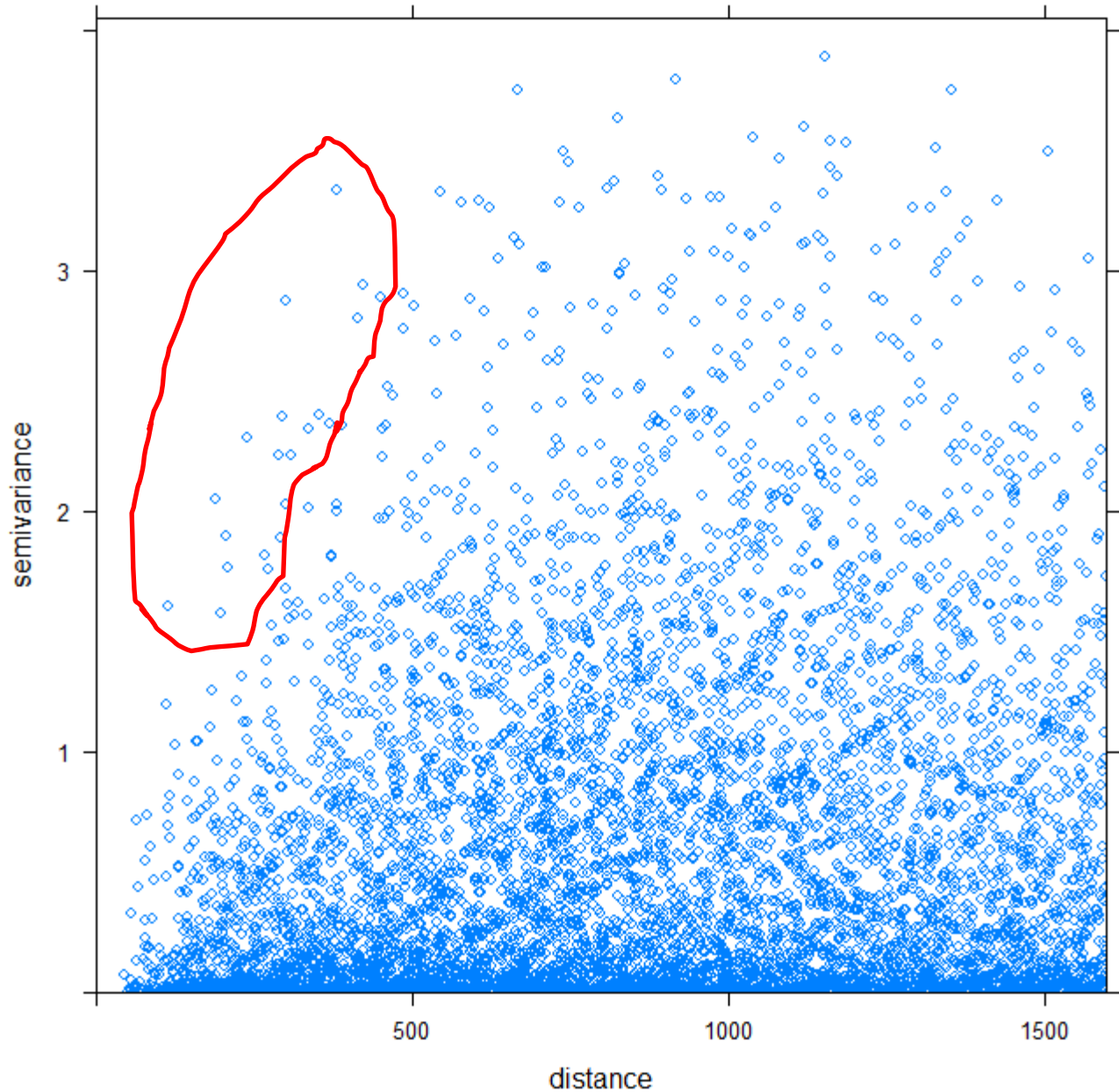
?interprétez?

```
plot(variogram(log(zinc)~1,meuse,cloud=TRUE))
```

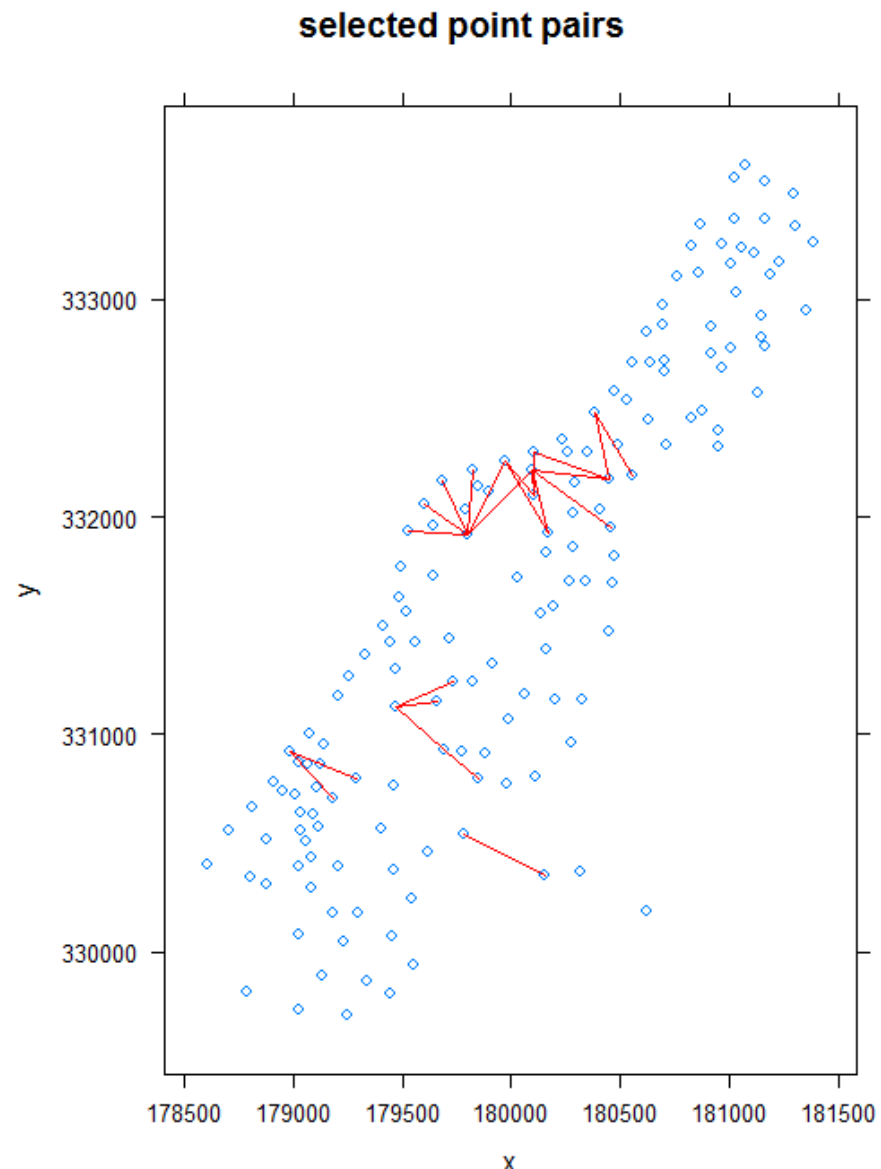


```
plot(variogram(log(zinc)~1,meuse,cloud=TRUE))
```

Quels sont les  
paires  
correspondantes?



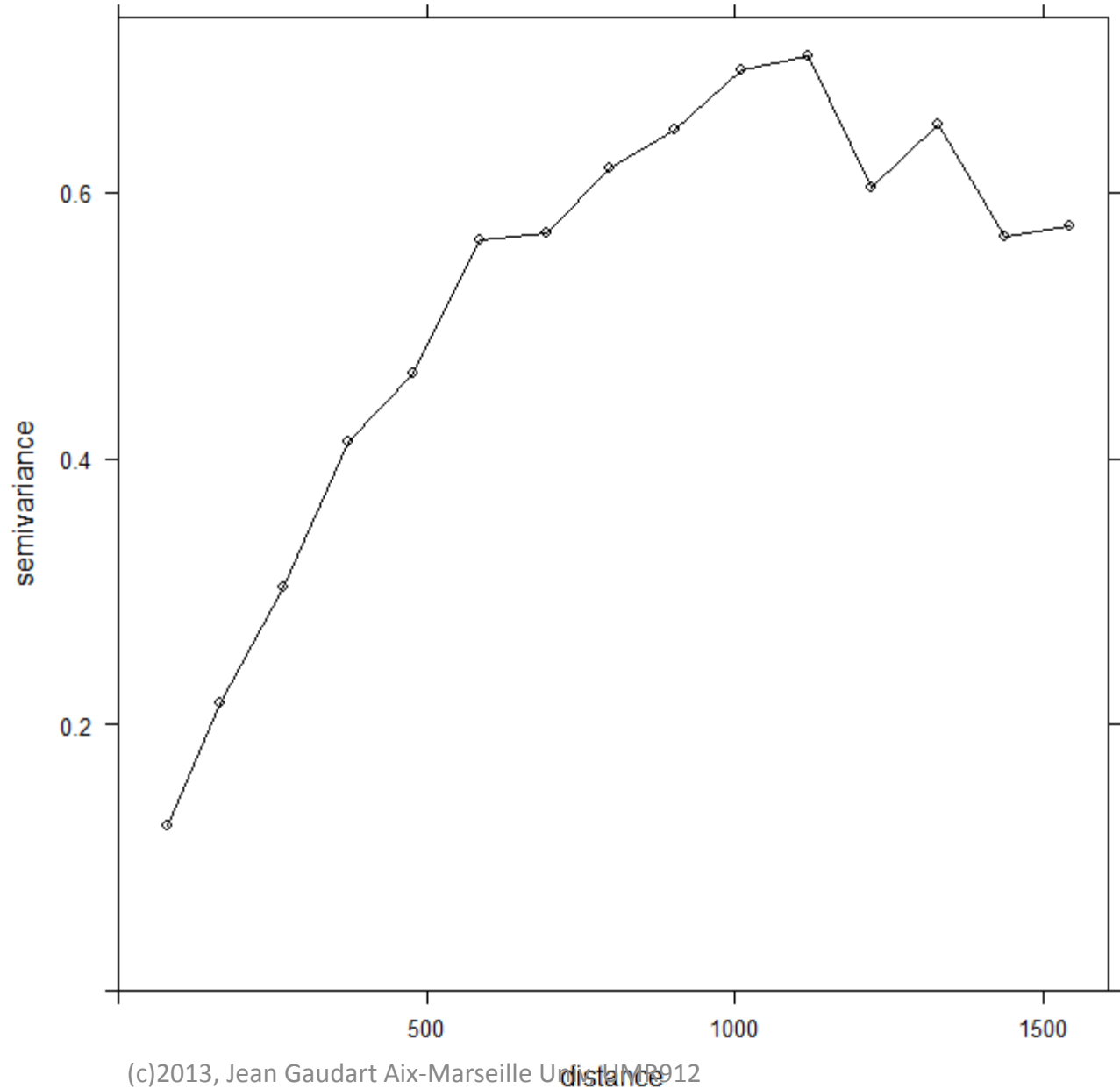
```
sele<-  
plot(variogram(log(zinc)~1,meuse,cloud=TRUE),digitize=TRUE)  
plot(sele,meuse)
```



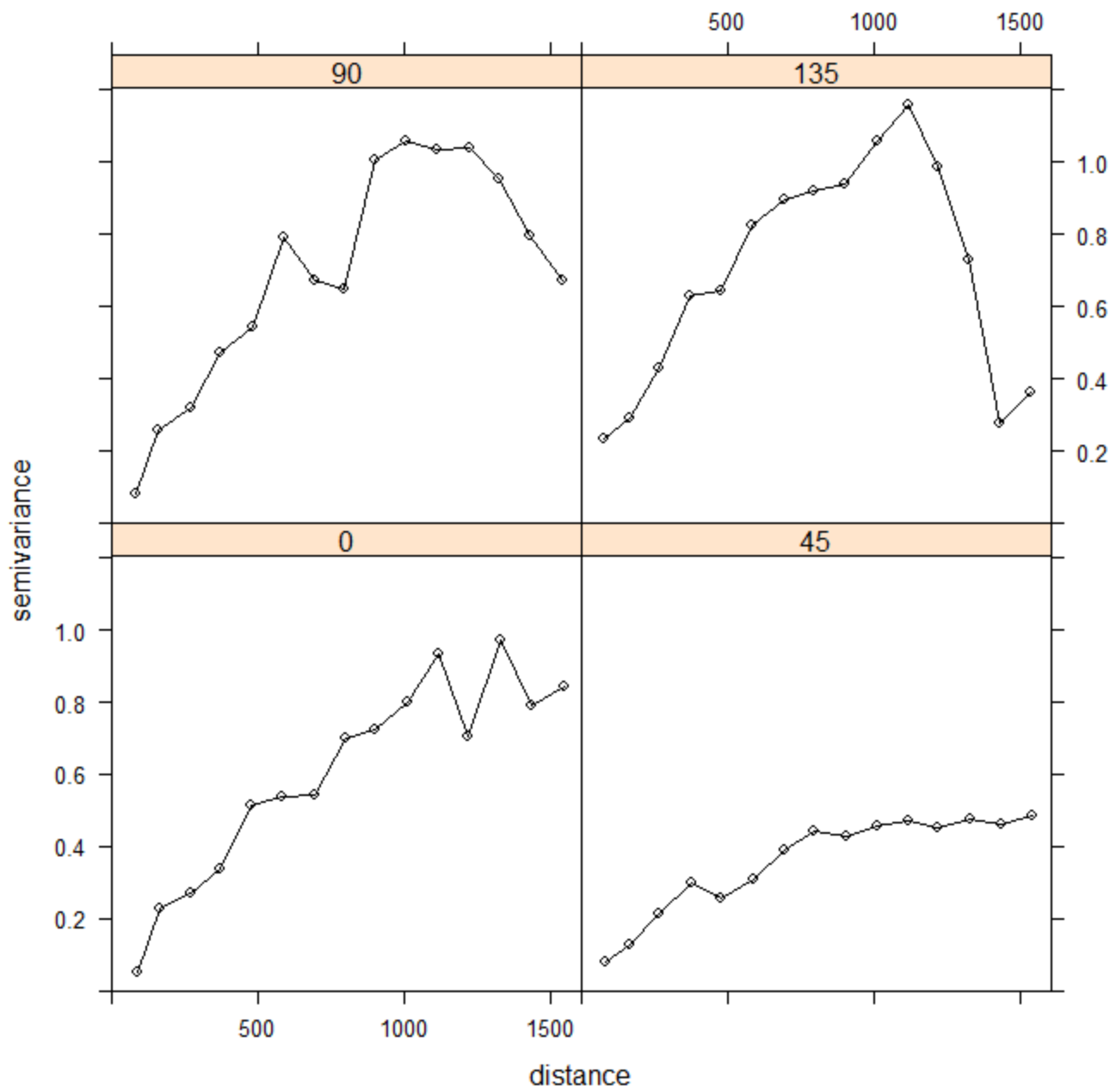
## Forme du variogramme

```
plot(variogram(log(zinc)~1,meuse,cloud=FALSE))
```

Attention:  
Nombre de paires



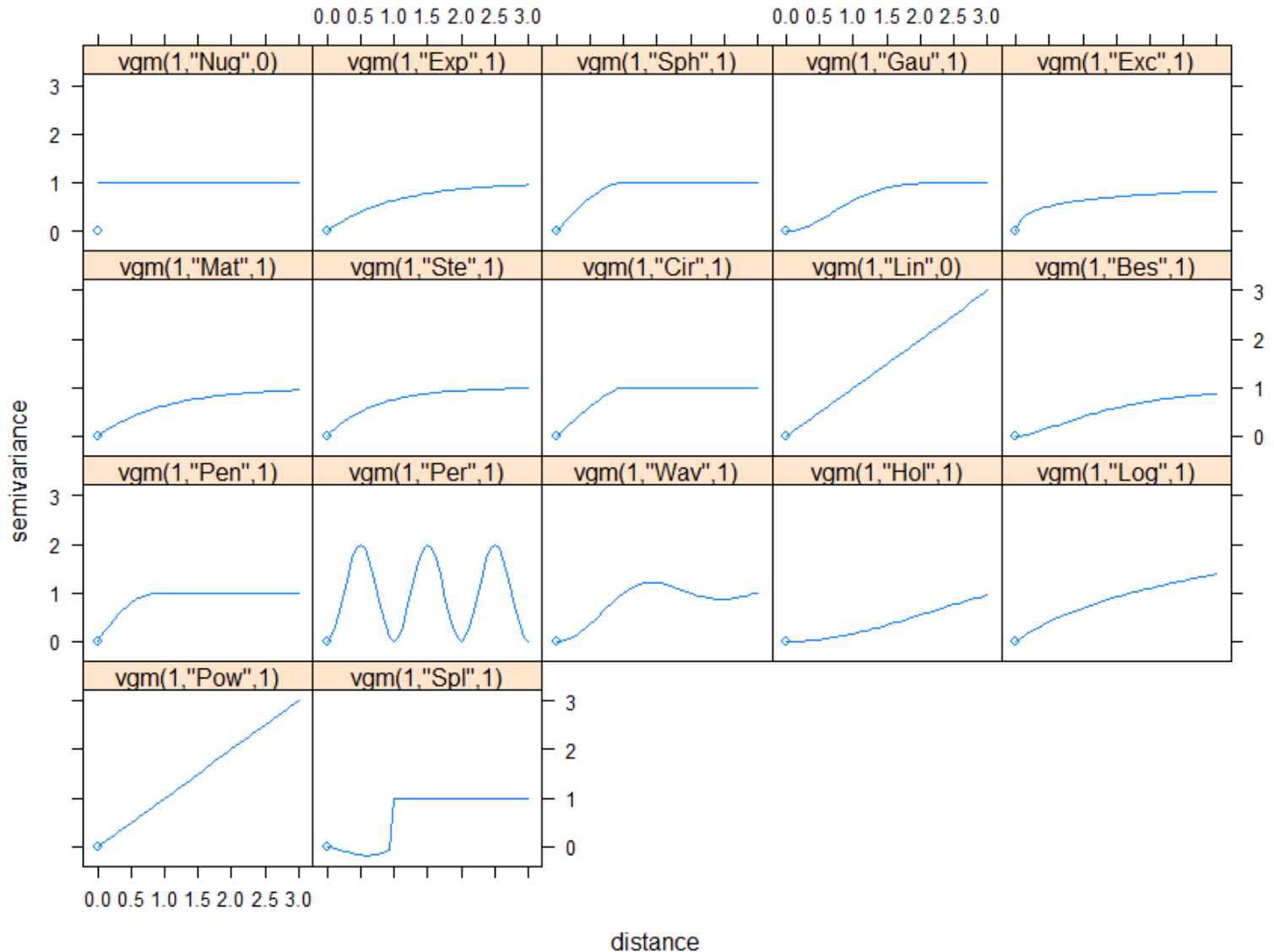
# Forme du variogramme dans 4 directions (anisotropie)



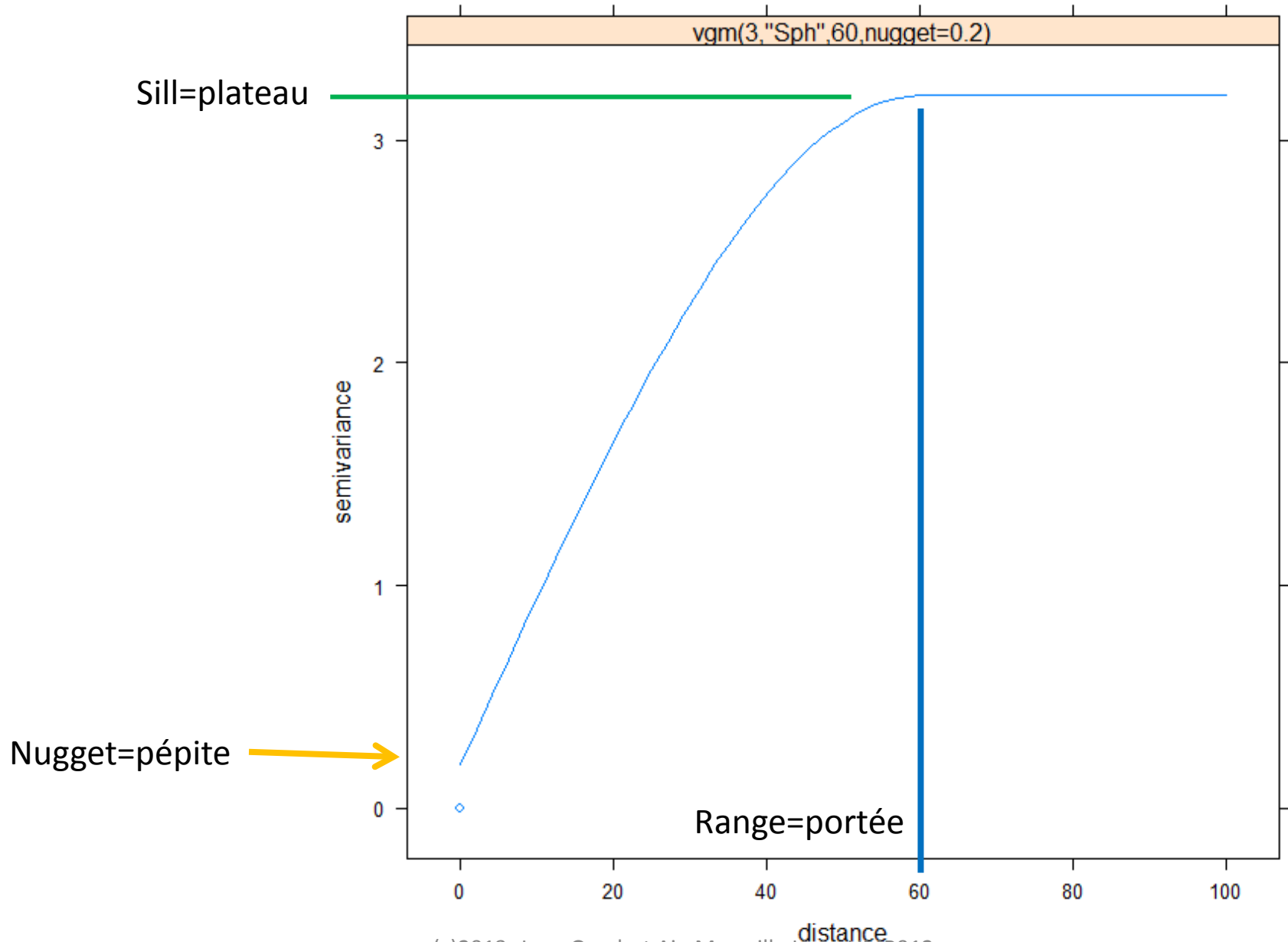


# Modélisation du variogramme

`show.vgms()`



```
show.vgms (model="Sph", nugget=0.2, range=60, sill=3, max=100)
```

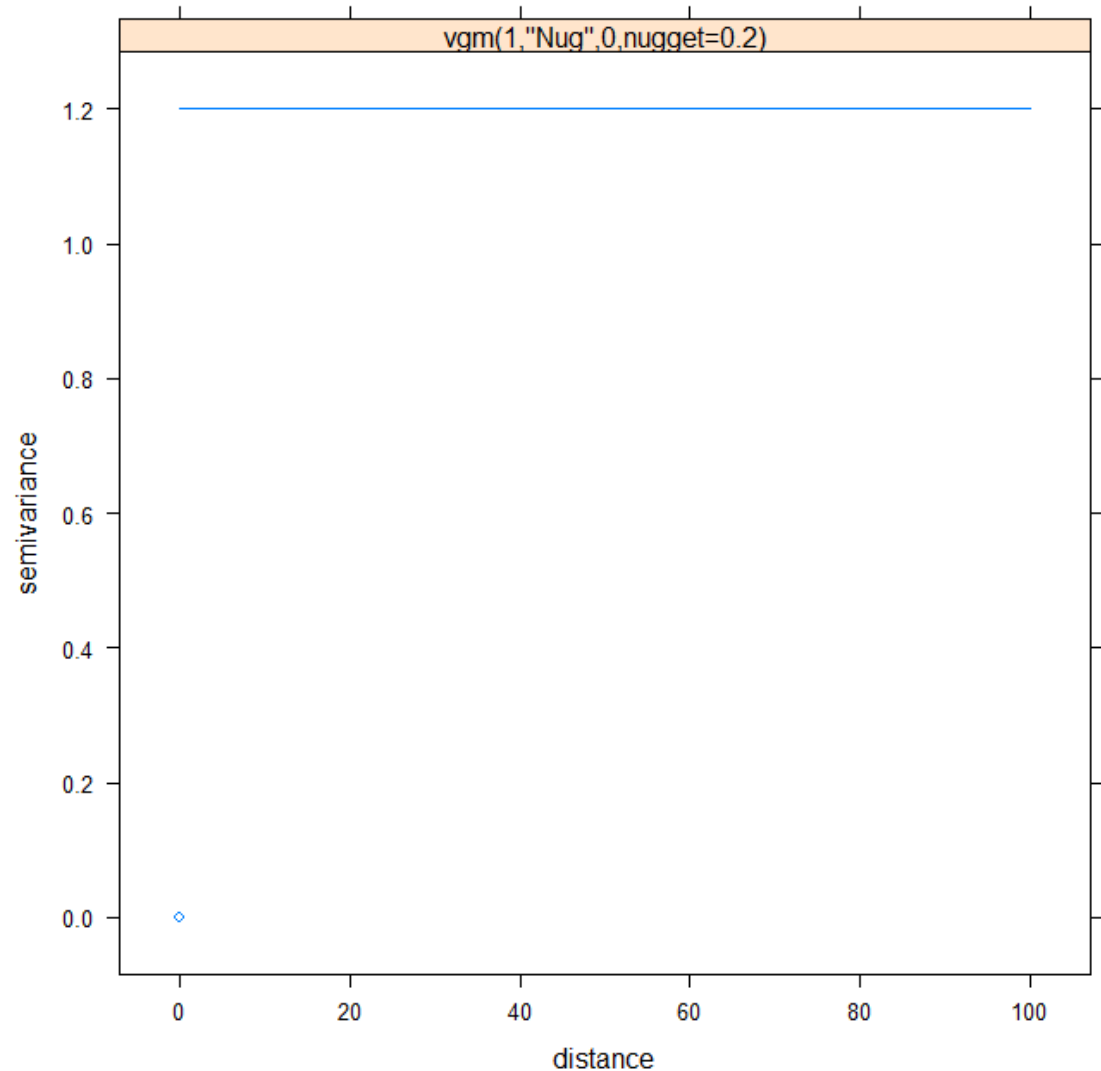


## Variogramme « pépite pure » (~bruit blanc)

```
show.vgms (model="Nug", nugget=0.2, sill=1, max=100)
```

$$\gamma(h) = \begin{cases} C, & \text{si } h > 0 \\ 0 & \text{si } h = 0 \end{cases}$$

C=pépite

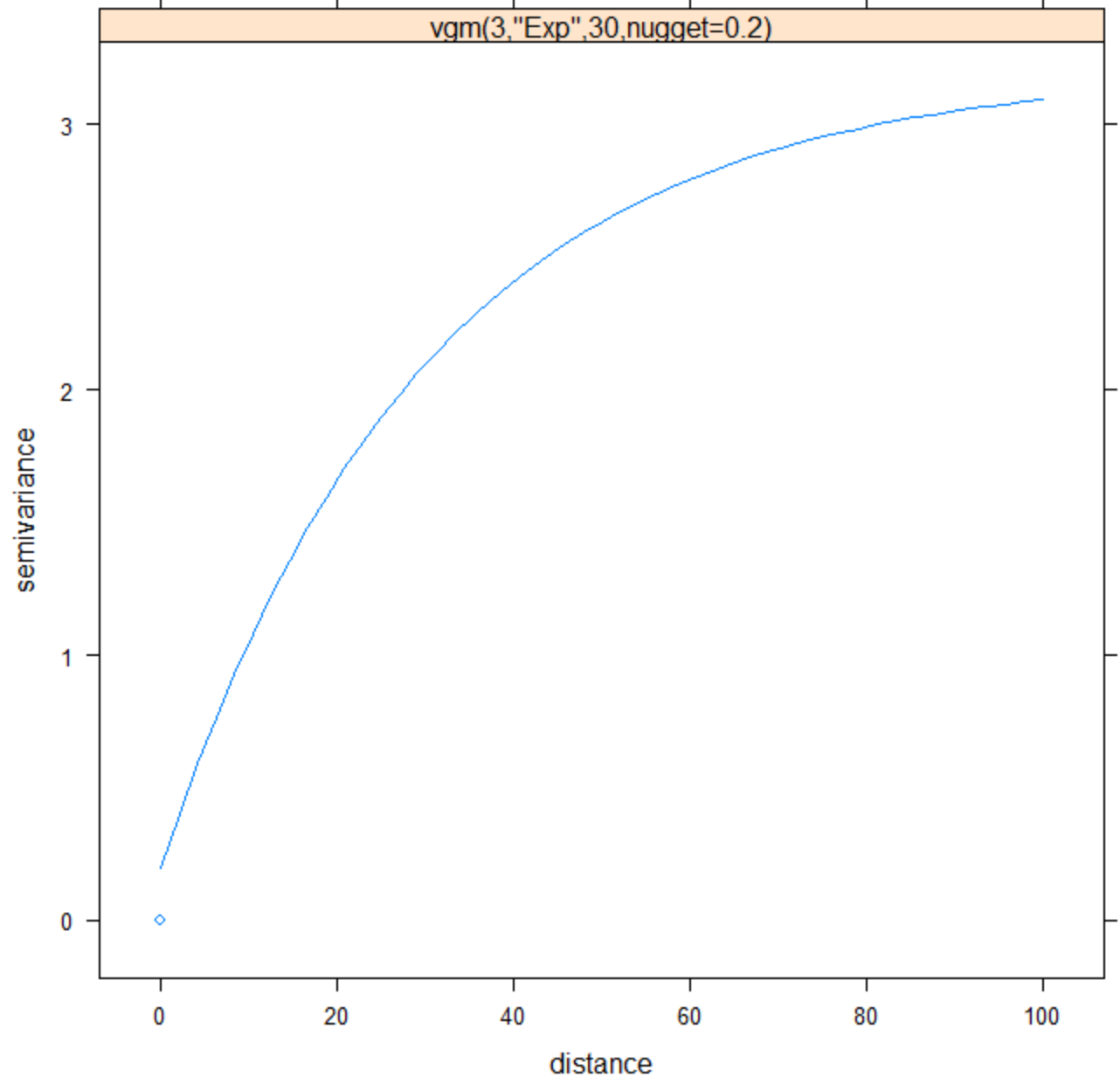


## Variogramme Exponentiel

```
show.vgms (model="Exp", nugget = 0.2, sill=3, range=30, max=100)
```

$$\gamma(h) = C \left( 1 - e^{-\frac{h}{a}} \right)$$

C=pépite  
a=portée

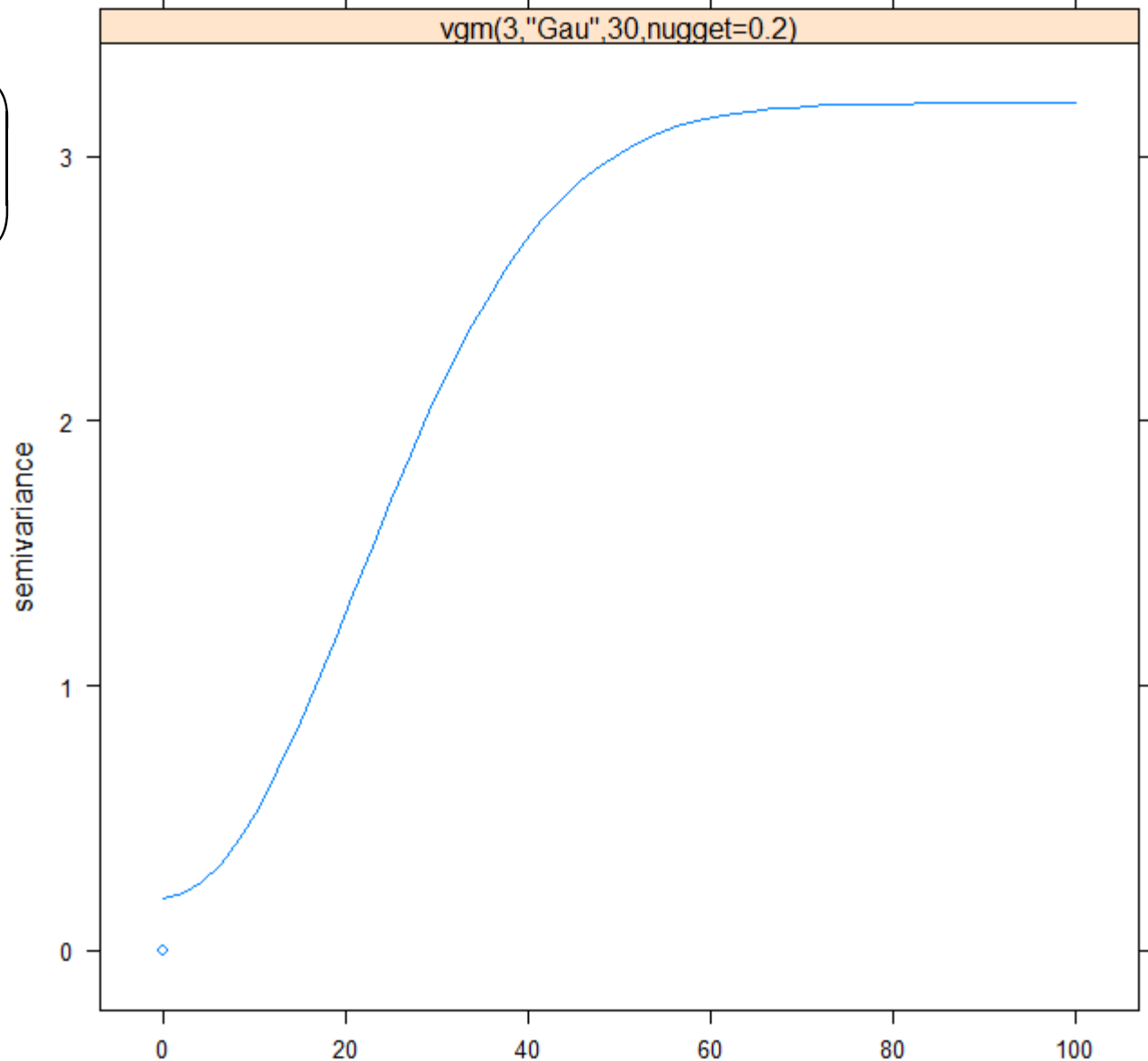


## Variogramme Gaussien

```
show.vgms (model="Gau", nugget=0.2, sill=3, range=30, max=100)
```

$$\gamma(h) = C \left( 1 - e^{-\left(\frac{h}{a}\right)^2} \right)$$

C=pépite  
a=portée

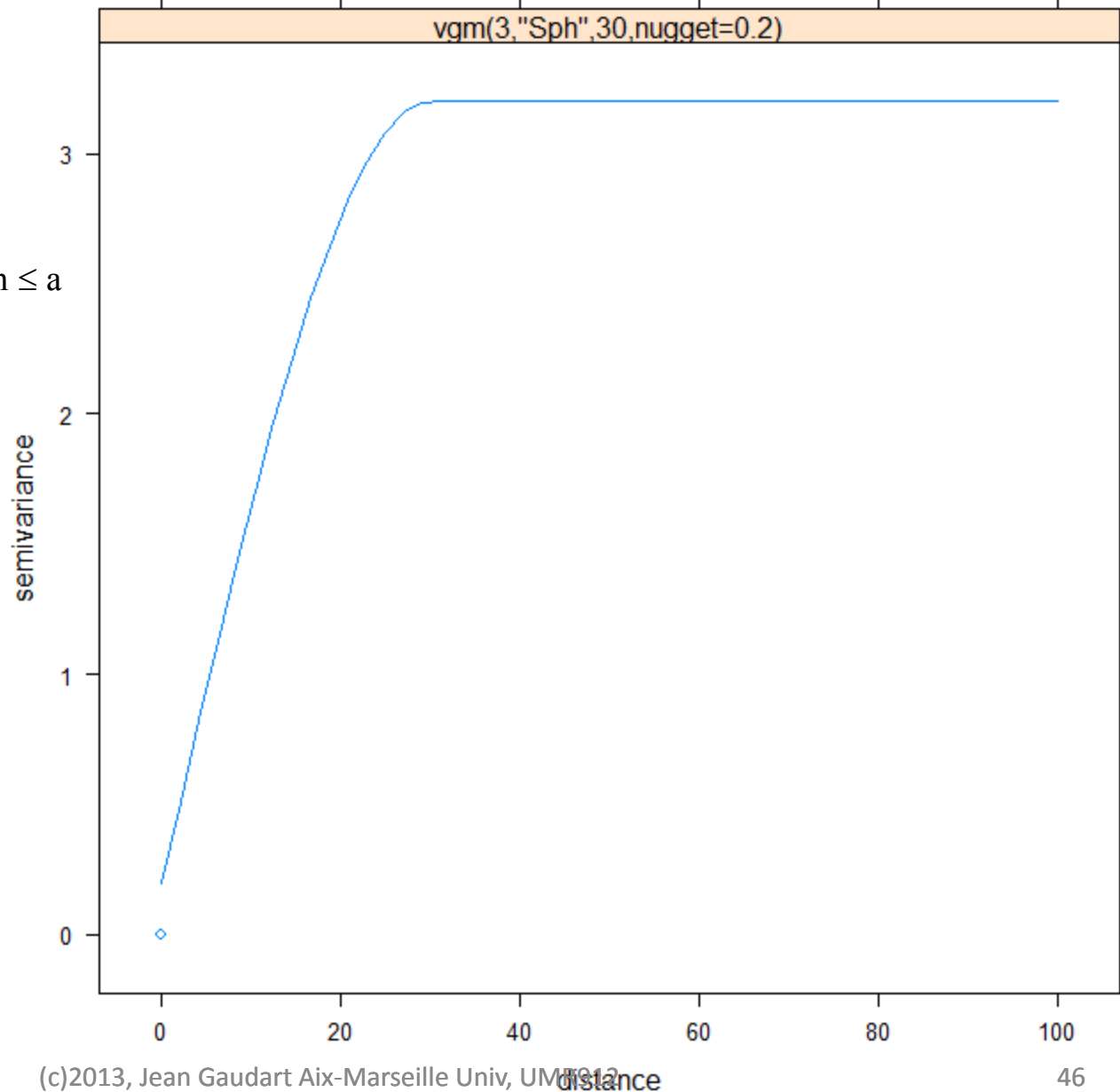


# Variogramme Sphérique

```
show.vgms (model="Sph", nugget=0.2, sill=3, range=30, max=100)
```

$$\gamma(h) = \begin{cases} C, & \text{si } h \geq a \\ C \left( \frac{3h}{2a} - \frac{1}{2} \left( \frac{h}{a} \right)^3 \right), & \text{si } 0 \leq h \leq a \end{cases}$$

C=pépite  
a=portée

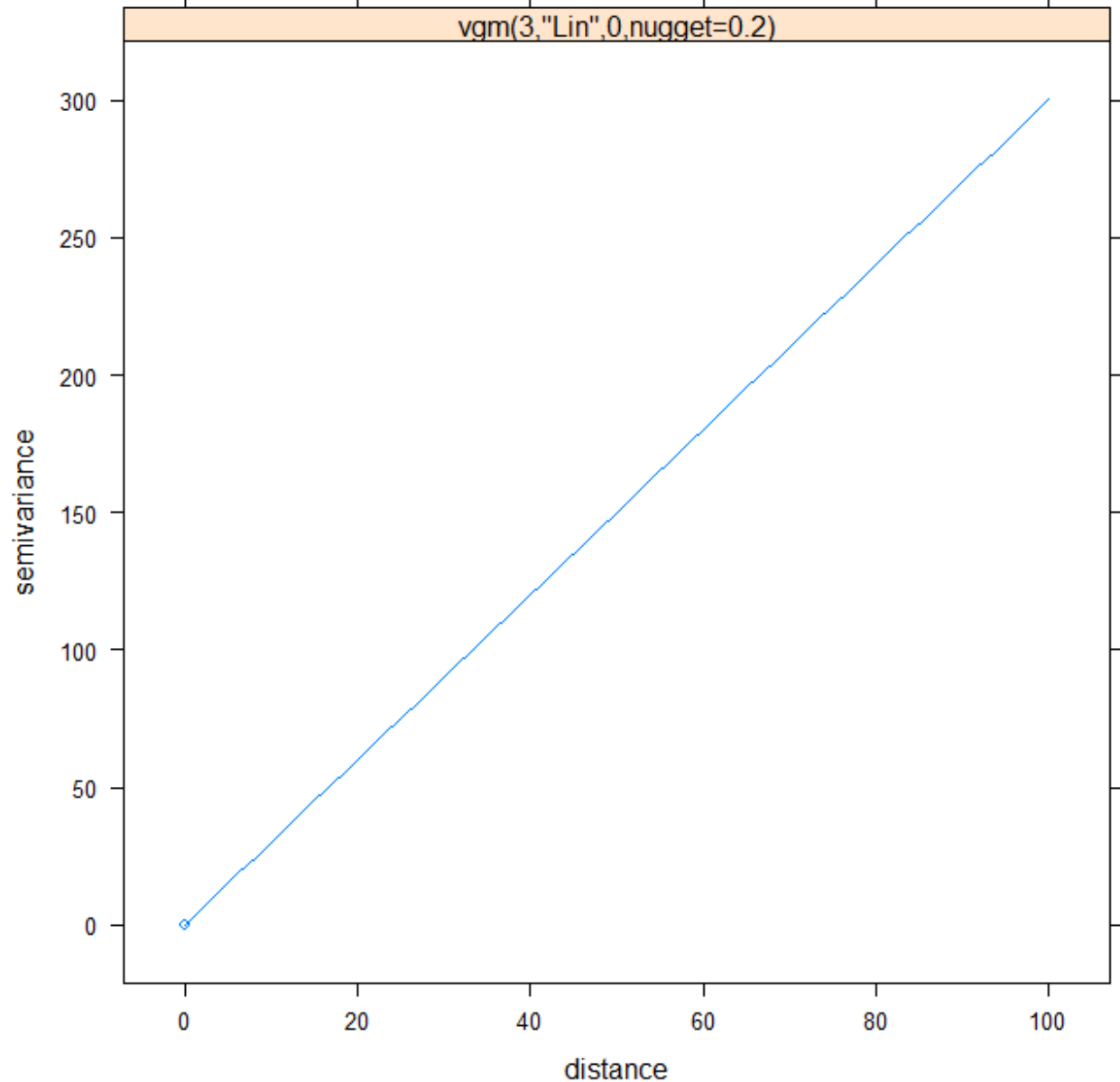


## Variogramme Linéaire

```
show.vgms(model="Lin", nugget=0.2, sill=3, range=30, max=100)
```

$$\gamma(h) = C \frac{h}{a}$$

C=pépite  
a=portée

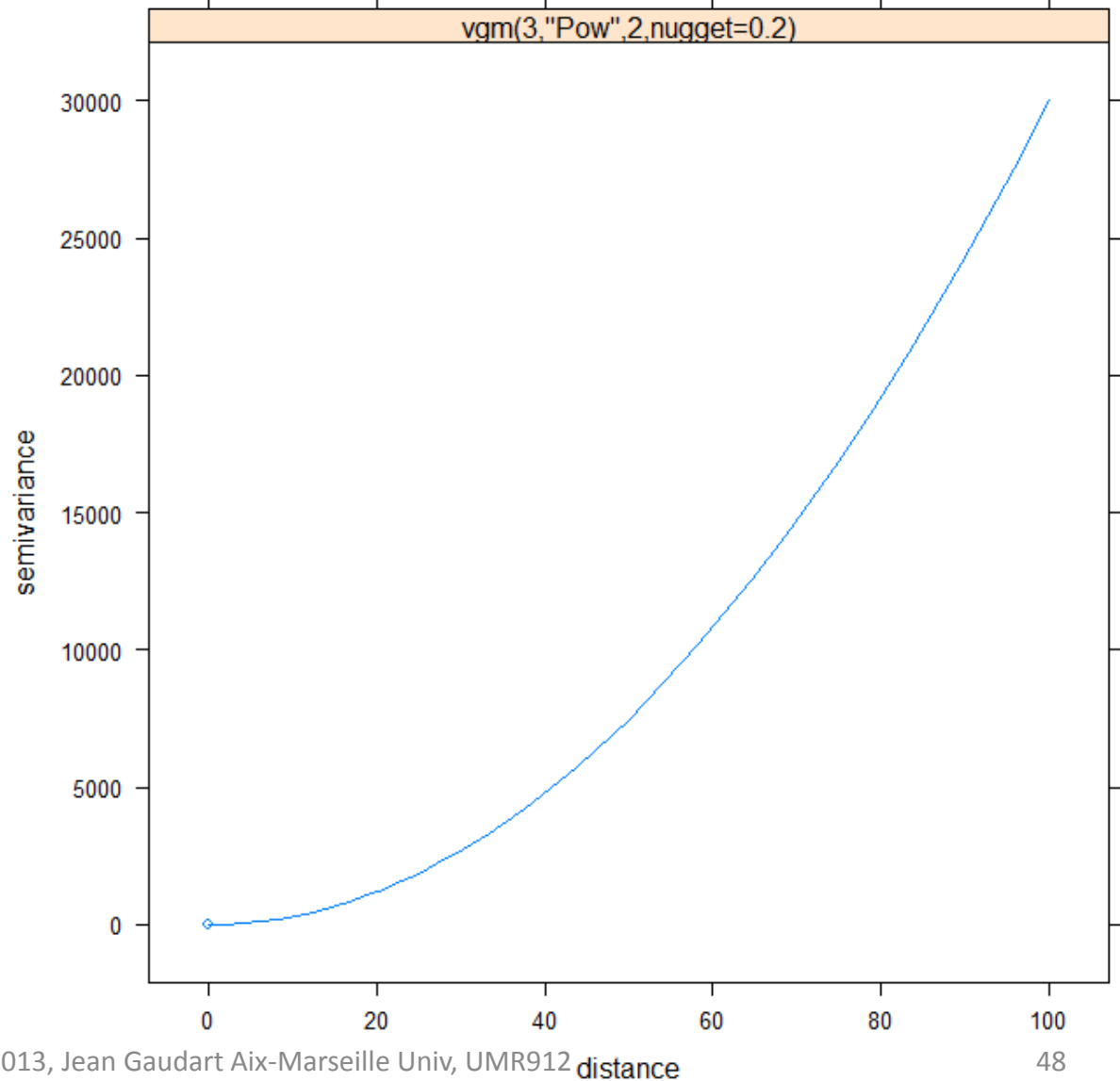


## Variogramme Puissance

```
show.vgms (model="Pow", nugget=0.2, sill=3, range=2, max=100)
```

$$\gamma(h) = C \left( \frac{h}{a} \right)^b$$

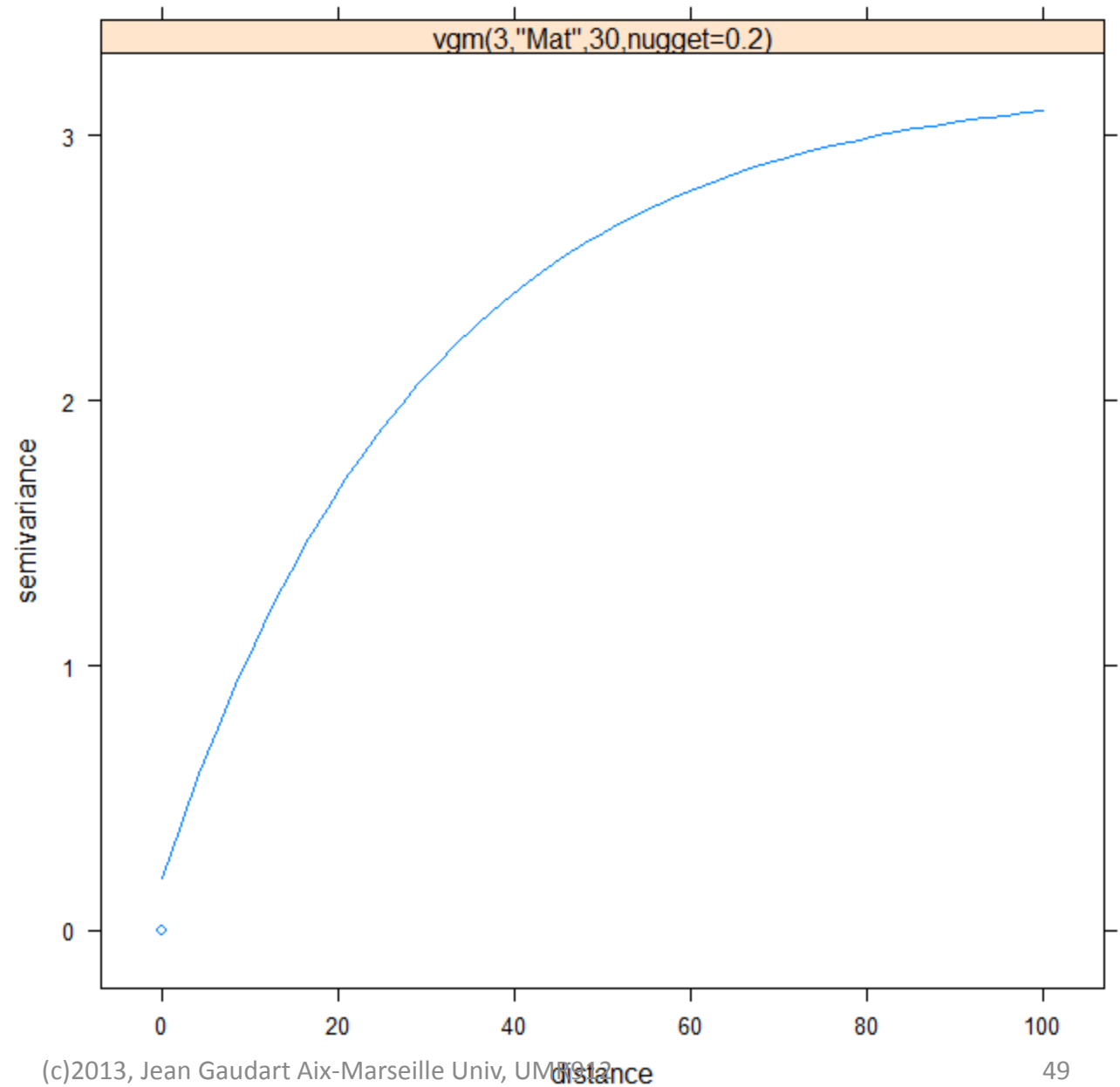
C=pépite  
a=portée





## Variogramme Matérn

```
show.vgms (model="Mat", nugget=0.2, sill=3, range=30, max=100)
```



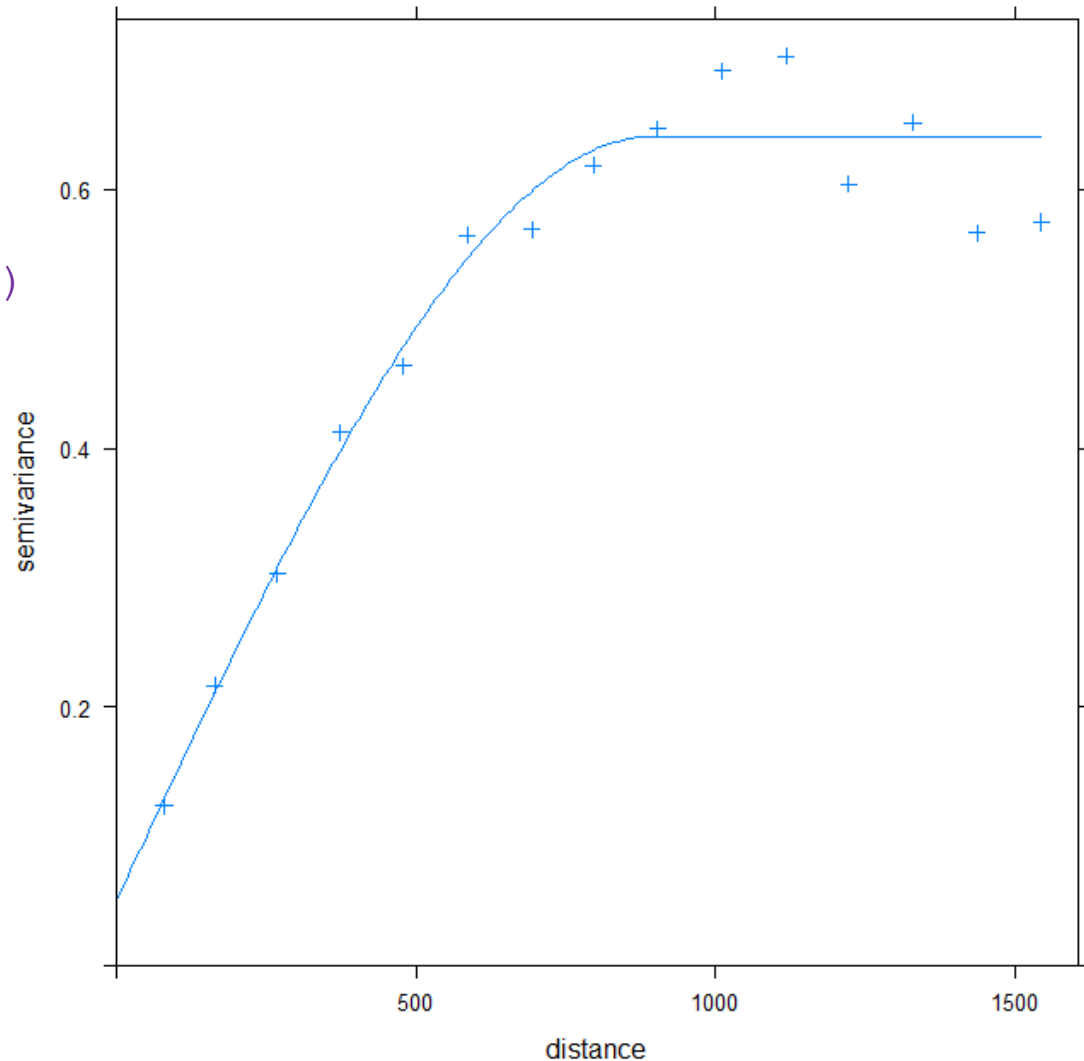
Modèles de variogramme les plus utilisés:  
Sphérique, Gaussien, Exponentiel, Matérn, Puissance

1. Choisir un modèle, en fonction du variogramme observé
2. Choisir des valeurs initiales: portée, pépité, plateau
3. Modéliser les observations par le modèle choisie

## Modèle sphérique

```
svgm <- variogram(log(zinc) ~ 1, meuse)  
v.fit <- fit.variogram(svgm, vgm(1, "Sph", 800, 1))  
plot(svgm, v.fit, pch = 3)
```

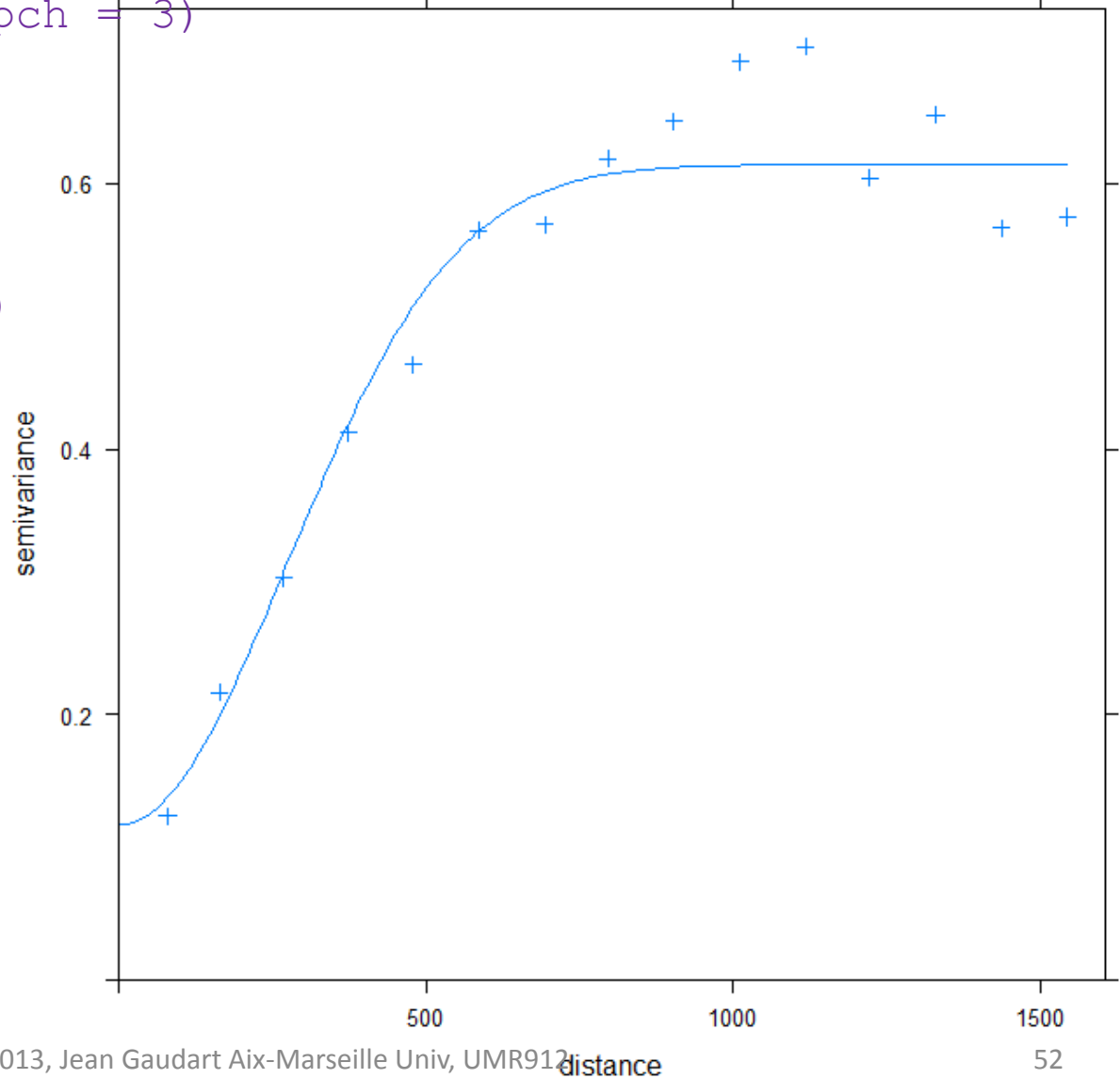
```
attr(v.fit, "SSErr")  
[1] 9.011194e-06
```



## Modèle Gaussien

```
svgm <- variogram(log(zinc) ~ 1, meuse)  
v.fit2 <- fit.variogram(svgm, vgm(1, "Gau", 800, 1))  
plot(svgm, v.fit2, pch = 3)
```

```
attr(v.fit2, "SSErr")  
[1] 1.915069e-05
```



## Interpolation:      Krigeage Ordinaire:

```
meuse.zinc.ok<-krige(log(zinc)~1,meuse,meuse.grid,v.fit)
summary(meuse.zinc.ok)
```

Object of class SpatialPixelsDataFrame

Coordinates:

	min	max
x	178460	181540
y	329620	333740

Is projected: NA

proj4string : [NA]

Number of points: 3103

Grid attributes:

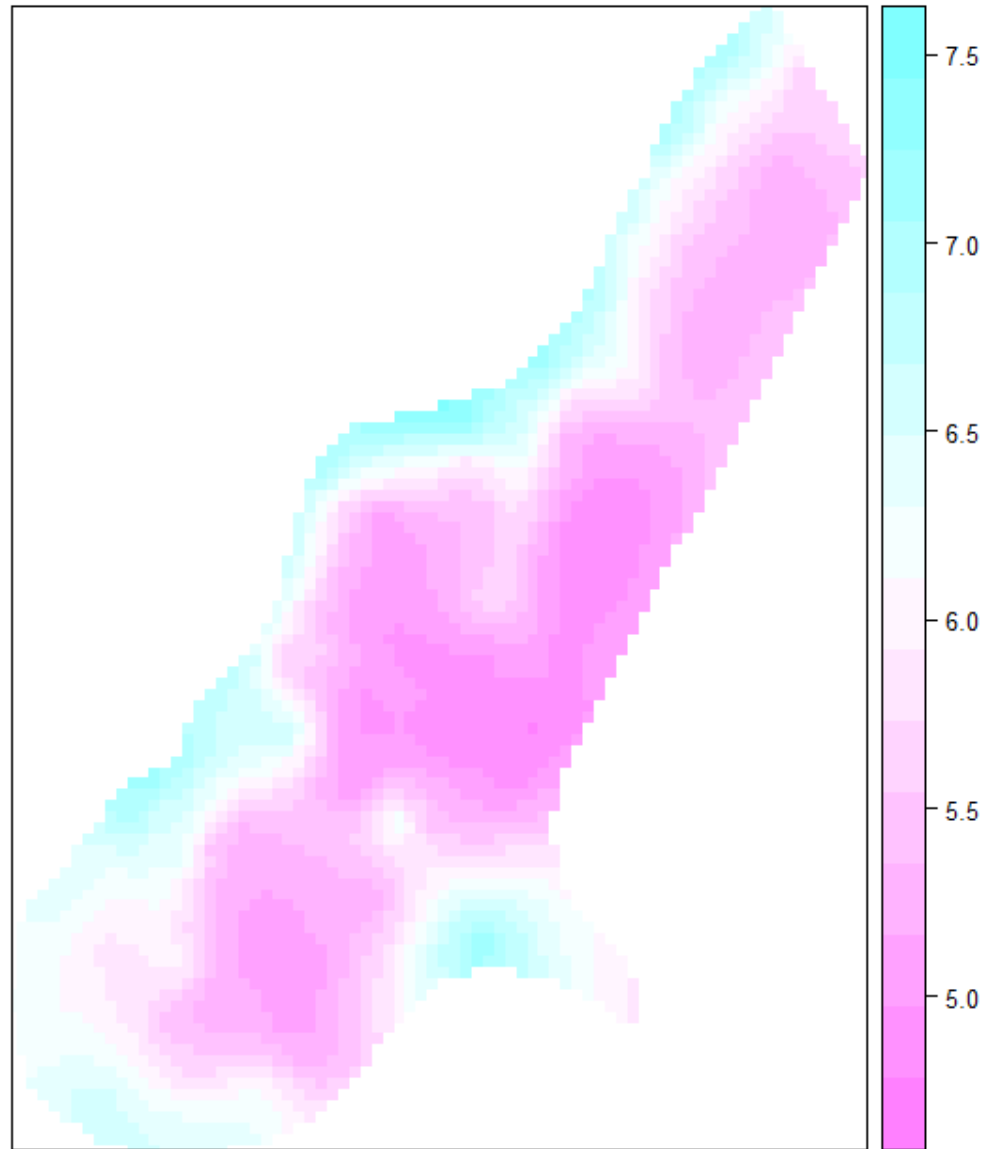
	cellcentre.offset	cellsize	cells.dim	
x		178460	40	78
y		329620	40	104

Data attributes:

	var1.pred	var1.var
Min.	:4.777	Min. :0.08549
1st Qu.	:5.238	1st Qu.:0.13728
Median	:5.573	Median :0.16218
Mean	:5.707	Mean :0.18533
3rd Qu.	:6.172	3rd Qu.:0.21161
Max.	:7.440	Max. :0.50028

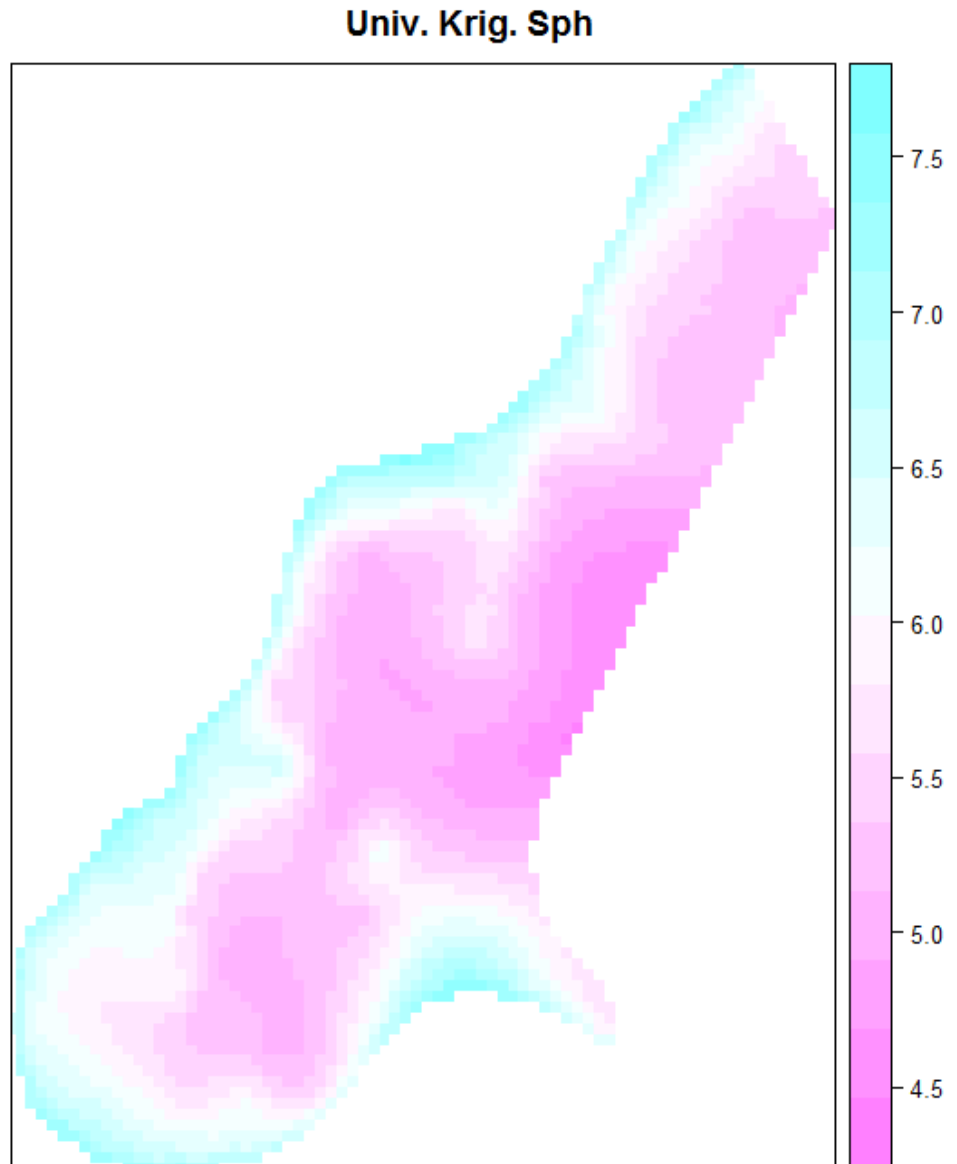
```
spplot(meuse.zinc.ok, "var1.pred", main="Ord. Krig. Sph")
```

**Ord. Krig. Sph**



## Krigeage Universel:

```
meuse.zinc.uk<-krige(log(zinc)~sqrt(dist),meuse,meuse.grid,v.fit)  
spplot(meuse.zinc.uk,"var1.pred",main="Univ. Krig. Sph")
```

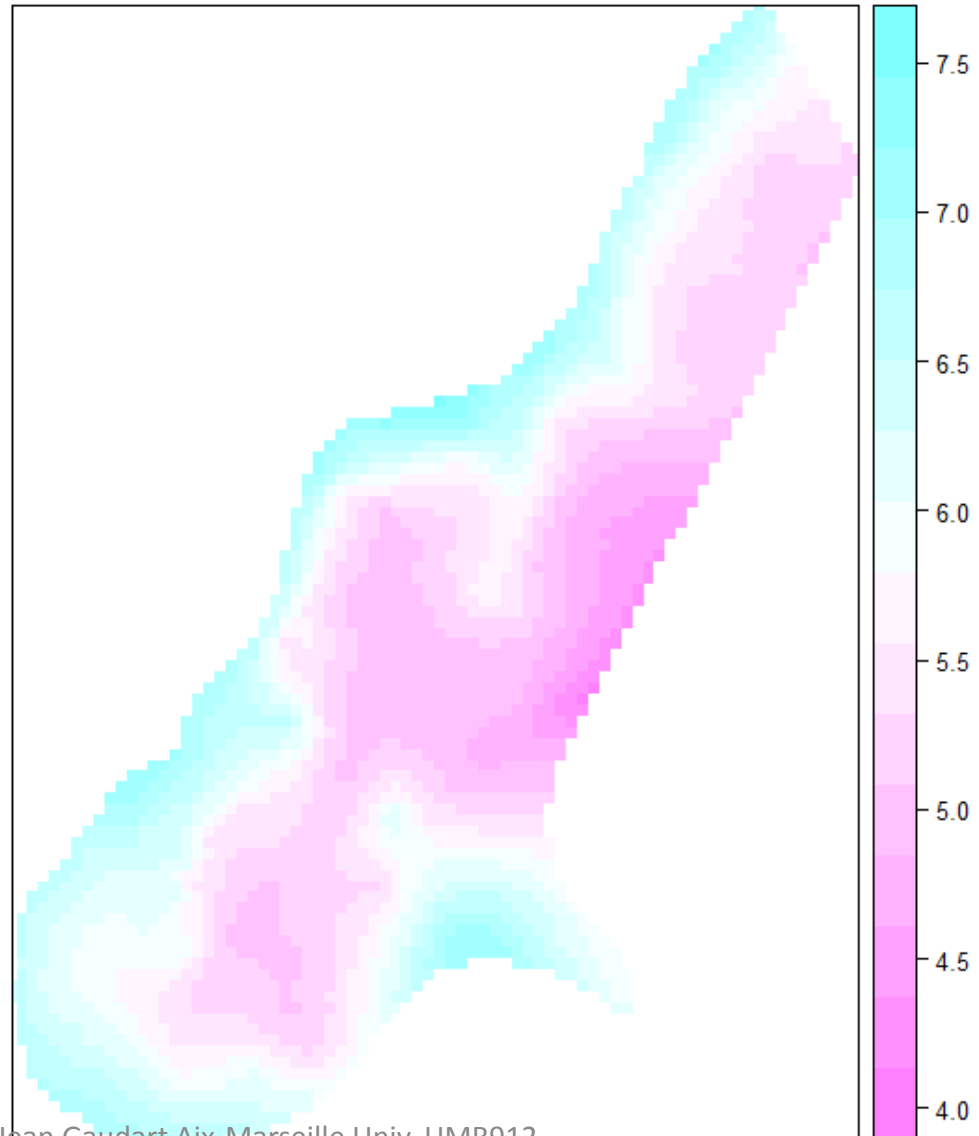


## Krigeage Universel (suite):

```
meuse.uk2<-krige(log(zinc)~exp(-dist/25),meuse,meuse.grid,v.fit)  
spplot(meuse.uk2,"var1.pred",main="Univ. Krig. Sph")
```

**Univ. Krig. Sph**

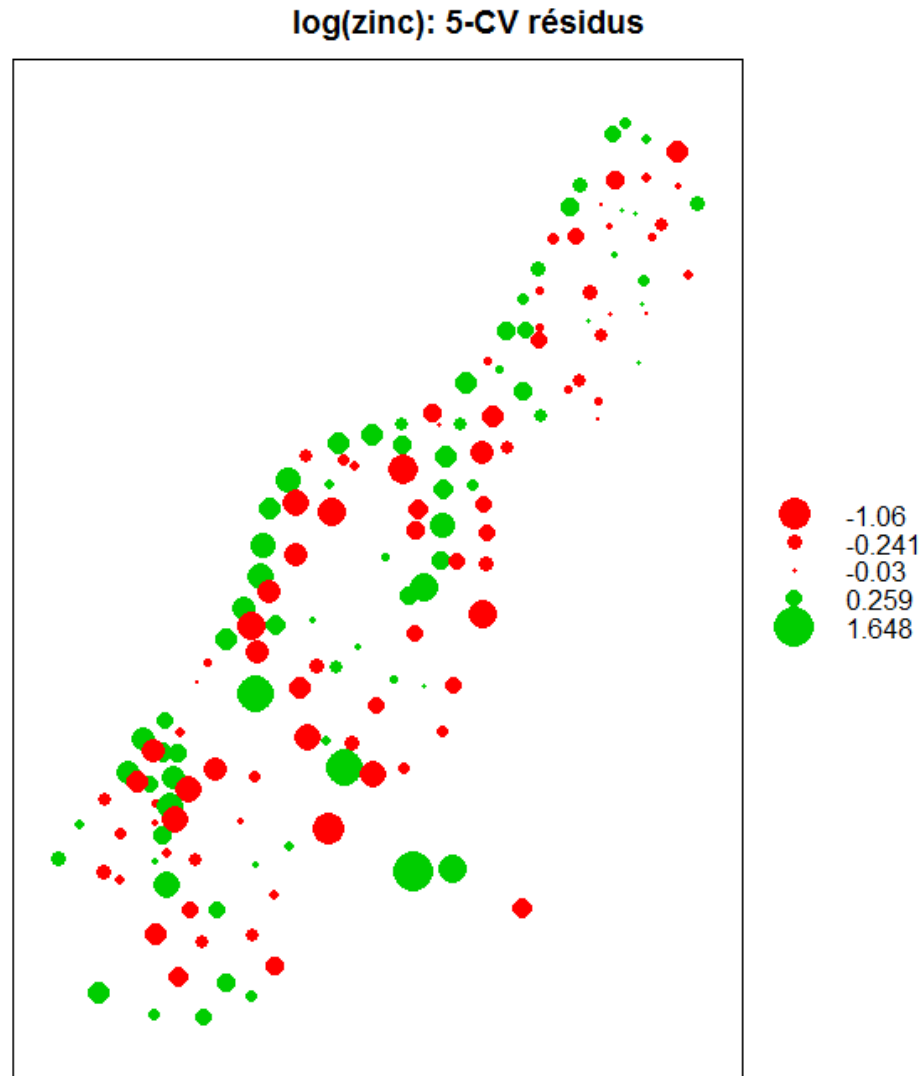
*?Que remarquez vous?*



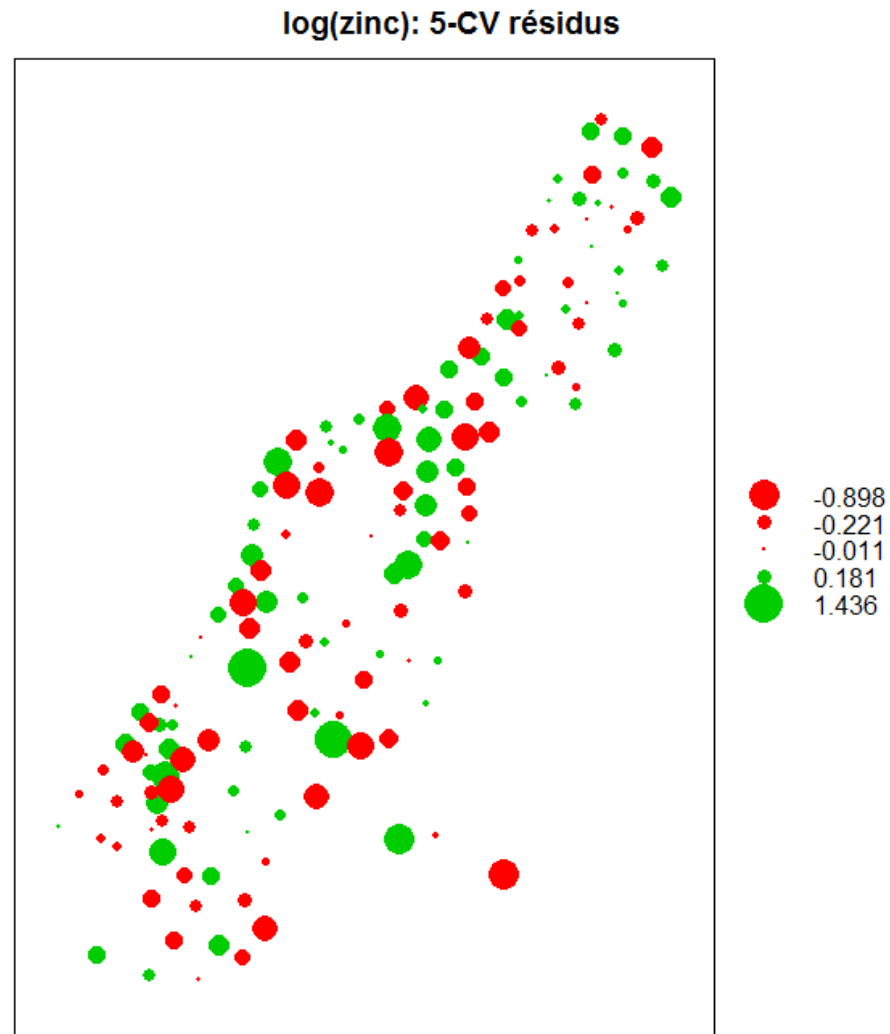


## Diagnostic de la modélisation: validation croisée

```
meuse.cv<-krige.cv(log(zinc)~1,meuse,v.fit,nfold=5)  
bubble(meuse.cv, "residual", main = "5-CV résidus")
```



```
meuse.cv2<-  
krige.cv(log(zinc)~sqrt(dist),meuse,v.fit,nfold=5)  
bubble(meuse.cv2, "residual", main = "5-CV résidus")
```



# V. Analyse de Co-facteur

Contrairement à l'interpolation, l'objectif ici est d'étudier le lien entre des facteurs de risque X et une variable à expliquer Y, en tenant compte de la spatialisation

Approche par les modèles GAM

- GLM:  $\mu_i = E(Y_i)$  avec  $g(\mu_i) = X_i\beta$   
Loi de la famille exponentielle (Gaussian, Poisson, Binomial ...)
- Modèle Additif Généralisé (GAM)

$$g(\mu_i) = X_i\beta + f_1(x_{1,i}) + f_2(x_{2,i}) + f_3(x_{1,i}, x_{2,i})$$

où les  $f(\cdot)$  sont des fonctions de lissage (splines)

=> modélise des relations non linéaires

- Si  $x_1$  et  $x_2$  sont les coordonnées de chaque localisation

$$g(\mu_i) = X_i\beta + f(x_{1,i}, x_{2,i})$$

=> modélise la variabilité spatiale des  $Y_i$

- « *Thin plate splines* » : Splines à plaque mince
  - Estimer une fonction de plusieurs prédicteurs
  - Sans besoin de spécifier les nœuds ni la base
  - Recherche une surface lissée la moins déformée possible:  
=> minimiser le degré de courbure tout en passant à proximité des observations

$$\|y - f\|^2 + \lambda \iint \left( \frac{\partial^2 f}{\partial x_1^2} \right)^2 + 2 \left( \frac{\partial^2 f}{\partial x_1 \partial x_2} \right)^2 + \left( \frac{\partial^2 f}{\partial x_2^2} \right)^2 dx_1 dx_2$$

- Logiciel R, package *mgcv*

```
install.packages("mgcv")
library(mgcv)
summary(meuse)

mod.lm1<-lm(log(zinc)~dist+elev+soil, data=meuse)
summary(mod.lm1)
```

Call: lm(formula = log(zinc) ~ dist + elev + soil, data = meuse)

Residuals:

	Min	1Q	Median	3Q	Max
	-1.05511	-0.26034	0.01597	0.25479	1.08302

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )	
(Intercept)	8.49161	0.30166	28.149	< 2e-16	***
dist	-1.96409	0.24341	-8.069	2.11e-13	***
elev	-0.25911	0.03951	-6.557	8.31e-10	***
soil2	-0.09565	0.09167	-1.043	0.298	
soil3	0.12545	0.16560	0.758	0.450	

**ATTENTION!**

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4273 on 150 degrees of freedom

Multiple R-squared: 0.6587, Adjusted R-squared: 0.6496

F-statistic: 72.38 on 4 and 150 DF, p-value: < 2.2e-16

```
mod.lm2<lm(log(zinc)~
  dist+elev+soil+x+y+I(x^2)+I(y^2)+I(x*y),data=meuse)
summary(mod.lm2)
```

Residuals:

Min	1Q	Median	3Q	Max
-0.90671	-0.23200	0.02764	0.22892	1.08983

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )	
(Intercept)	3.894e+03	5.404e+03	0.721	0.47234	
dist	-1.005e+00	3.523e-01	-2.854	0.00496	**
elev	-2.793e-01	3.924e-02	-7.118	4.68e-11	***
soil2	-1.033e-01	1.246e-01	-0.829	0.40846	
soil3	-3.598e-02	1.854e-01	-0.194	0.84643	
x	1.224e-01	4.560e-02	2.684	0.00812	**
y	-9.005e-02	4.975e-02	-1.810	0.07232	.
I(x^2)	3.536e-07	1.928e-07	1.834	0.06870	.
I(y^2)	3.412e-07	1.320e-07	2.585	0.01072	*
I(x * y)	-7.545e-07	2.650e-07	-2.847	0.00505	**

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
 Residual standard error: 0.4103 on 145 degrees of freedom  
 Multiple R-squared: 0.6958, Adjusted R-squared: 0.6769  
 F-statistic: 36.84 on 9 and 145 DF, p-value: < 2.2e-16

```
mod.gam1<-gam(log(zinc)~
dist+elev+soil, family="gaussian", data=meuse)
summary(mod.gam1)
```

Family: gaussian  
Link function: identity

Formula:  
log(zinc) ~ dist + elev + soil

Parametric coefficients:

	Estimate	Std. Error	t value	Pr(> t )	
(Intercept)	8.49161	0.30166	28.149	< 2e-16	***
dist	-1.96409	0.24341	-8.069	2.11e-13	***
elev	-0.25911	0.03951	-6.557	8.31e-10	***
soil2	-0.09565	0.09167	-1.043	0.298	
soil3	0.12545	0.16560	0.758	0.450	

---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) = 0.65      Deviance explained = 65.9%  
GCV score = 0.18867      Scale est. = 0.18258      n = 155

```
mod.gam1p<-gam(zinc~
dist+elev+soil, family="poisson", data=meuse)
summary(mod.gam1p)
```

```
Family: poisson
Link function: log
```

```
Formula:
zinc ~ dist + elev + soil
```

```
Parametric coefficients:
```

	Estimate	Std. Error	z value	Pr(> z )	
(Intercept)	8.494003	0.029794	285.090	<2e-16	***
dist	-2.513710	0.031208	-80.546	<2e-16	***
elev	-0.232647	0.004036	-57.639	<2e-16	***
soil2	-0.146465	0.011446	-12.796	<2e-16	***
soil3	-0.034154	0.023993	-1.424	0.155	

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
R-sq.(adj) = 0.612    Deviance explained = 67.4%
UBRE score = 78.902  Scale est. = 1          n = 155
```



```
mod.gam2p<-gam(zinc~dist+elev+soil
+x+y+I(x^2)+I(y^2)+I(x*y), family="poisson", data=meuse)
summary(mod.gam2p)
```

Family: poisson Link function: log

Formula:

zinc ~ dist + elev + soil + x + y + I(x^2) + I(y^2) + I(x \* y)

Parametric coefficients:

	Estimate	Std. Error	z value	Pr(> z )	
(Intercept)	0.000e+00	0.000e+00	NA	NA	
dist	-1.584e+00	4.249e-02	-37.285	< 2e-16	***
elev	-2.666e-01	4.316e-03	-61.779	< 2e-16	***
soil2	-1.862e-01	1.401e-02	-13.297	< 2e-16	***
soil3	-3.000e-01	3.002e-02	-9.993	< 2e-16	***
x	1.329e-01	4.684e-03	28.368	< 2e-16	***
y	-7.221e-02	2.544e-03	-28.385	< 2e-16	***
I(x^2)	1.501e-07	2.133e-08	7.036	1.99e-12	***
I(y^2)	2.625e-07	9.306e-09	28.205	< 2e-16	***
I(x * y)	-5.644e-07	2.559e-08	-22.060	< 2e-16	***

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) = 0.632 Deviance explained = 70.6%

UBRE score = 71.18 Scale est. = 1 n = 155

```
mod.gam3p<-gam(zinc~dist+elev+soil
+s(x,y), family="poisson", data=meuse)
summary(mod.gam3p)
```

Family: poisson Link function: log
Formula: zinc ~ dist + elev + soil + s(x, y)
Parametric coefficients:

	Estimate	Std. Error	z value	Pr(> z )	
(Intercept)	8.629247	0.077135	111.872	< 2e-16	***
dist	-2.029306	0.240797	-8.427	< 2e-16	***
elev	-0.265391	0.005592	-47.458	< 2e-16	***
soil2	-0.126823	0.020675	-6.134	8.57e-10	***
soil3	-0.293970	0.041853	-7.024	2.16e-12	***

---
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:

	edf	Ref.df	Chi.sq	p-value	
s(x,y)	28.82	29	8240	<2e-16	***

---
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) = 0.848 Deviance explained = 88.7%
UBRE score = 27.05 Scale est. = 1 n = 155

SIR associés aux cofacteurs:

```
exp(mod.gam3p$coeff[2:5])
```

```
dist      elev      soil2      soil3  
0.1314267 0.7669060 0.8808892 0.7452989
```

# VI. Recherche de Cluster

## Méthodes

- autour d'un facteur de risque identifié, géo-localisé  
ex. centrales nucléaires / leucémies
- Méthodes générales  
recherche d'agrégats sans facteur de risque géo-localisé
  - Globales – « clustering »
  - Locales

- Détection globale

- Le pattern observé globalement est-il concordant avec l'hypothèse nulle ?

- corrélations



Moran

- comparaisons de distributions



Tango

1 test

- Détection locale

- Y a-t-il localement un excès de cas ?

- corrélations (voisinage)



LISA

- comparaisons de groupes de proximités



Satscan

k tests

# Hypothèse

## 3 questions possibles :

- Les distributions des cas de chaque zone sont-elles indépendantes?
- Quelles sont ces distributions ?
- Le risque est-il constant sur l'ensemble de la zone géographique?

# Hypothèses nulles possibles:

- Distribution des cas Uniforme sur le plan

***Complete Spatial Randomness***

- Distribution de Poisson hétérogène

***Constant Risk Hypothesis***

nombre de cas attendus dans la région  $i$  ←  $E_i = \lambda n_i$  → Effectif région  $i$

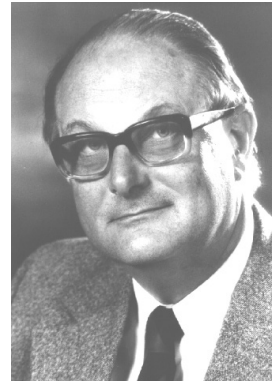
risque constant

$$\tilde{\lambda} = \frac{O_+}{n_+}$$

nombre total de cas observés :  $O_+ = \sum_{i=1}^k O_i$

effectif total :  $n_+ = \sum_{i=1}^k n_i$

# VI.1 Méthodes Globales



## Coefficient de Moran (1917-1988)

- Rappel Coefficient de corrélation de Pearson

$$r = \frac{\sum_i^n (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^n (X_i - \bar{X})^2 \sum_{i=1}^n (Y_i - \bar{Y})^2}}$$



- Coefficient de corrélation
- Pondéré par les distances
- **Similarités** entre régions:

écart à la moyenne de la région  $i$

$\Leftrightarrow$  écart à la moyenne de la région  $j$

$$I = \frac{K \times \sum_{i,j} w_{ij} (Y_i - \bar{Y})(Y_j - \bar{Y})}{w_+ \times \sum_{i=1}^K (Y_i - \bar{Y})^2}$$

*Plus les zones  $i$  et  $j$  sont éloignées, moins le poids est important*

## Préparation pour l'analyse

```
library(DCluster)
```

```
SPAT<-read.csv2("D:/Mon_dossier/DATA1.csv",header=TRUE)
attach(SPAT)
```

← charger les données

```
TAB<-data.frame(Observed=cas)
TAB<-cbind(TAB, Expected=n*sum(cas)/sum(n))
```

← construire le tableau de données adapté

```
SPAT.xy = SPAT[c("x", "y")]
coordinates(SPAT.xy) <- c("x", "y")
class(SPAT.xy)
```

← construire le tableau des coordonnées

```
coords<-coordinates(SPAT.xy)
```

```
dlist<-dnearneigh(coords, 0, Inf)
dlist<-include.self(dlist)
dlist.d<-nbdists(dlist, coords)
col.W<-nb2listw(
  dlist,glist=lapply(dlist.d,function(x){exp(-x)}),style="C")
```

← construire le tableau des distances et la matrice des poids

## Estimation du coefficient de Moran

```
I<-
```

```
moranI.stat(TAB, listw=col.W, n=length(dlist), S0=Szero(col.W) )
```

```
[1] 0.356332
```

$$S0 = w_+ = \sum_{i=1}^k w_i$$

## Test:

H0:  $I=0$ ; malades spatialement indépendants

H1:  $I>0$

– Sous H0: 
$$z = \frac{I - E(I)}{\sqrt{\text{var}(I)}} \sim \mathcal{N}(0;1)$$

– Condition : distribution des  $Y$  normale

- insoutenable
- unité statistique: région, avec  $k$  petit

– Loi de  $I$  inconnue  $\Rightarrow$  Monte-Carlo ou Bootstrap

## test du coefficient de Moran

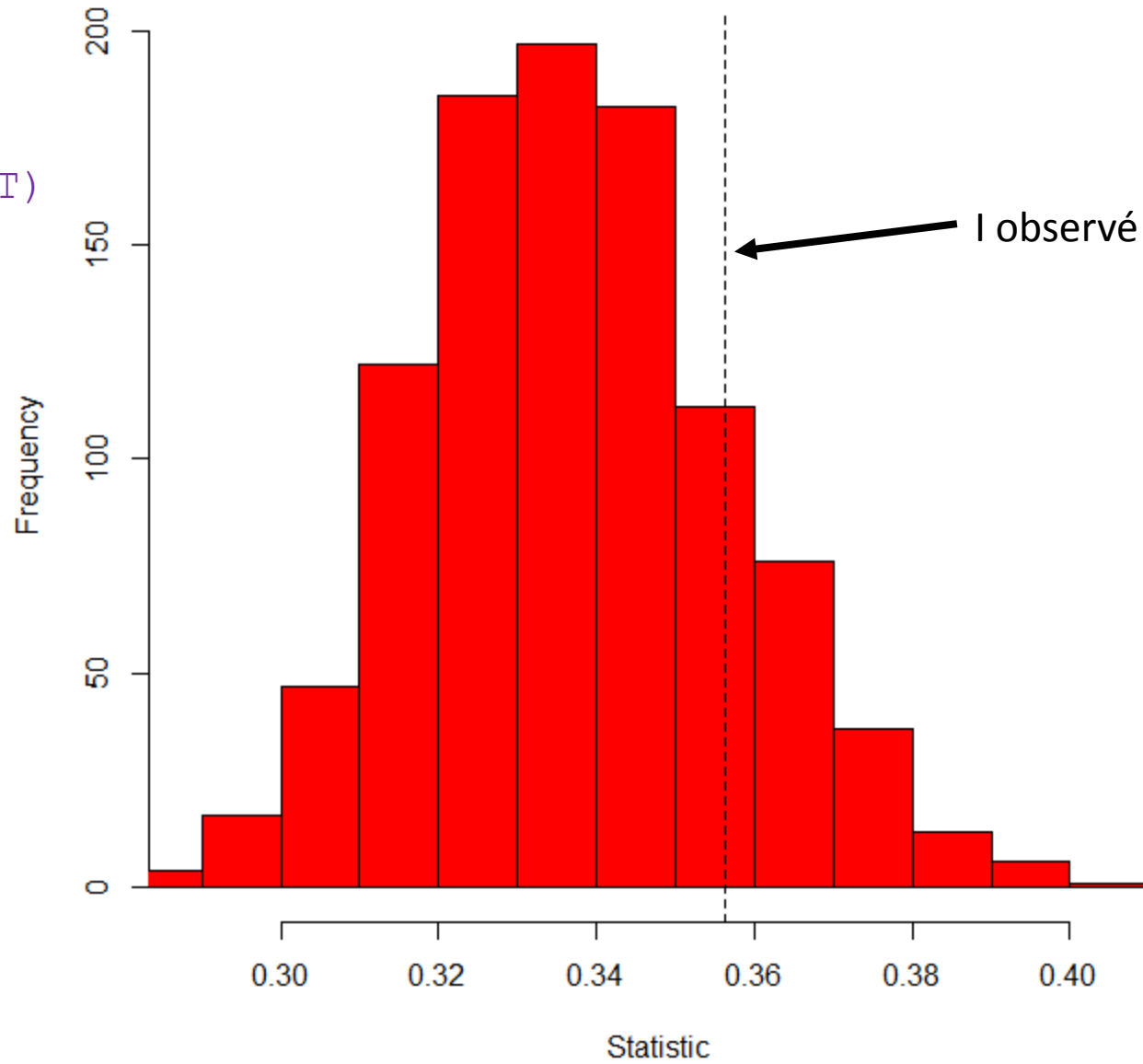
```
IT<-moranI.test(Observed~offset(log(Expected)), TAB,  
model="poisson", R=999, listw=col.W, n=length(dlist),  
S0=Szero(col.W))
```

Moran's I test of spatial autocorrelation

```
Type of boots.: parametric  
Model used when sampling: Poisson  
Number of simulations: 999  
Statistic: 0.356332  
p-value : 0.324
```

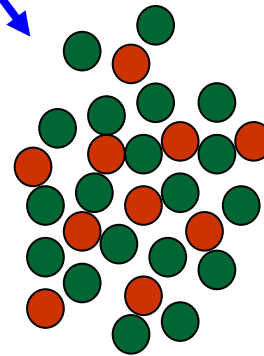
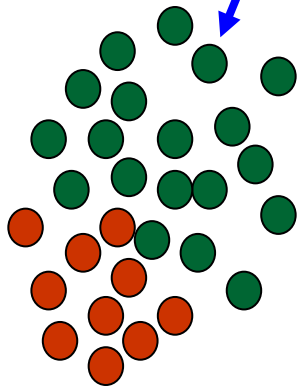
Histogram of the simulated values

plot(IT)



## Interprétation :

- $I > 0$ : régions voisines: **mêmes** écarts à la moyenne = pattern sous forme de clusters
- $I < 0$ : régions voisines: **≠** écarts à la moyenne, = pattern régulier.
- $I = 0$ : **aucune** corrélation spatiale.



- Mesure de l'écart à la moyenne générale:  
*pas d'interprétation locale possible.*

# Distances et Poids

- Distances entre localisations (ou centres des régions)

$$d_{ij} = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$$

- Différents poids = différents résultats !!

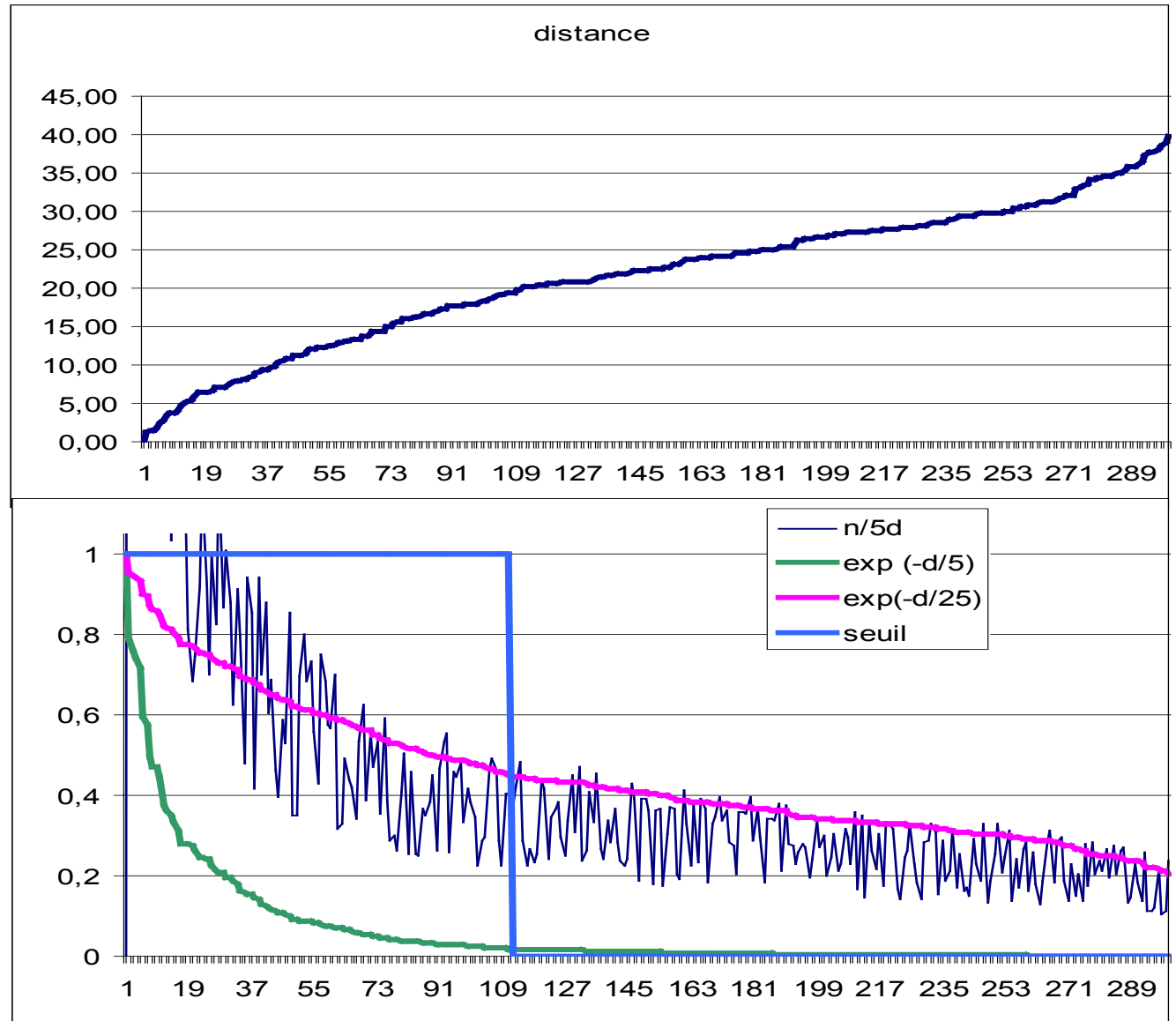
$$w_{ij} = \begin{cases} 1, & \text{si } d_{ij} < \delta \\ 0, & \text{si non} \end{cases}$$

$$w_{ij} = e^{\left(-\frac{d_{ij}}{\tau}\right)}$$

$$w_{ij} = d_{ij} / n_v$$



*Exemple:*  
Pour le point le plus à l'ouest



# Méthodes Locales

## LISA de Anselin

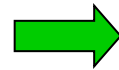
Local Indicator of Spatial Autocorrelation

Application locale du coefficient de Moran

*A proximité d'un cas observé, les cas sont-ils regroupés?*

*Global*

$$I = \frac{K \times \sum_{i,j} w_{ij} (Y_i - \bar{Y})(Y_j - \bar{Y})}{w_+ \times \sum_{i=1}^K (Y_i - \bar{Y})^2}$$



*Local: pour la région i*

$$I_i = (Y_i - \bar{Y}) \times \sum_{j=1}^K w_{ij} (Y_j - \bar{Y})$$

Test: Pour chaque région

H0:  $I_i = 0$  indépendance des régions/ voisins

H1:  $I_i > 0$

$$z = \frac{I_i - E(I_i)}{\sqrt{\text{var}(I_i)}} \sim \mathcal{N}(0;1)$$

– Sous H0 et condition  $\mathcal{N}$ :

Condition insoutenable:

=> Loi de  $I_i$  inconnue => Monte-Carlo ou Bootstrap

– **Tests multiples et corrélés**

=> *correction de Bonferroni*

$$\alpha_i = \frac{\alpha}{n_v}$$

```
plot(IT)
```

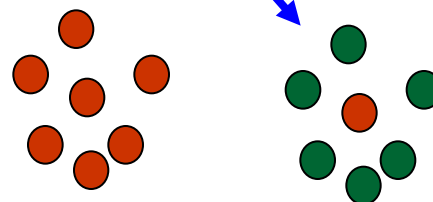
```
dlist2<-dnearneigh(coords, 0, Inf)  
dlist2.d<-nbdists(dlist2, coords)
```

```
LM<-localmoran(TAB$Observed, nb2listw(dlist2),  
p.adjust.method="bonferroni")
```

	Ii	E.Ii	Var.Ii	Z.Ii	Pr (z>0)
1	-4.328844e-03	-0.003344482	1.720577e-05	-0.2373111	1
2	-5.323609e-03	-0.003344482	1.720577e-05	-0.4771299	1
3	-1.009687e-02	-0.003344482	1.720577e-05	-1.6278736	1
4	-1.284904e-05	-0.003344482	1.720577e-05	0.8031933	1

## Interprétation

- $I_i > 0$ : régions voisines **similaires** à région  $i$  =  
cluster local
- $I_i < 0$ : régions voisines:  $\neq$  à région  $i$ , =  
région  $i$  particulière.
- $I_i = 0$ : **aucune** corrélation spatiale entre la région  $i$   
et ses voisins.



# Sites

- RS. Bivand, EJ. Pebesma, V. Gomez-Rubio, Applied Spatial Data Analysis with R, eds. Springer, Use R!
- L. Waller, C. Gotway, Applied Spatial Statistics for Public Health Data, eds. Wiley
- Open Source Geospatial Fondation [www.osgeo.org](http://www.osgeo.org)
- Geospatial Data Abstraction Library [www.gdal.org](http://www.gdal.org)
- Cartographic Projections library [trac.osgeo.org/proj/](http://trac.osgeo.org/proj/)
- <http://www.asdar-book.org/>
- <http://trac.osgeo.org/>