Happy R - Investigations Spatiales et Cartographiques

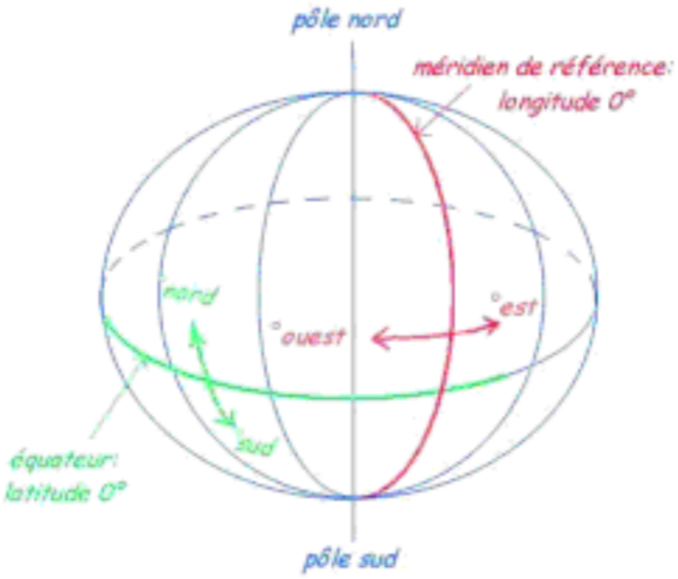
Suite et Poursuite de la présentation de Jessica du 17/11/17

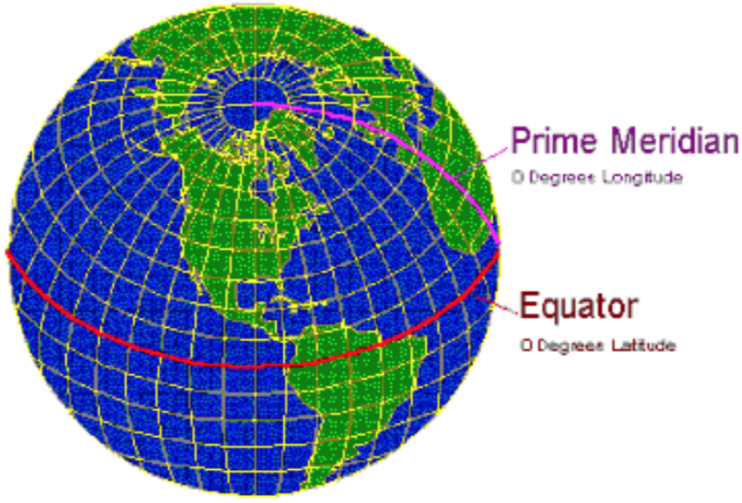
Eric & Isabelle

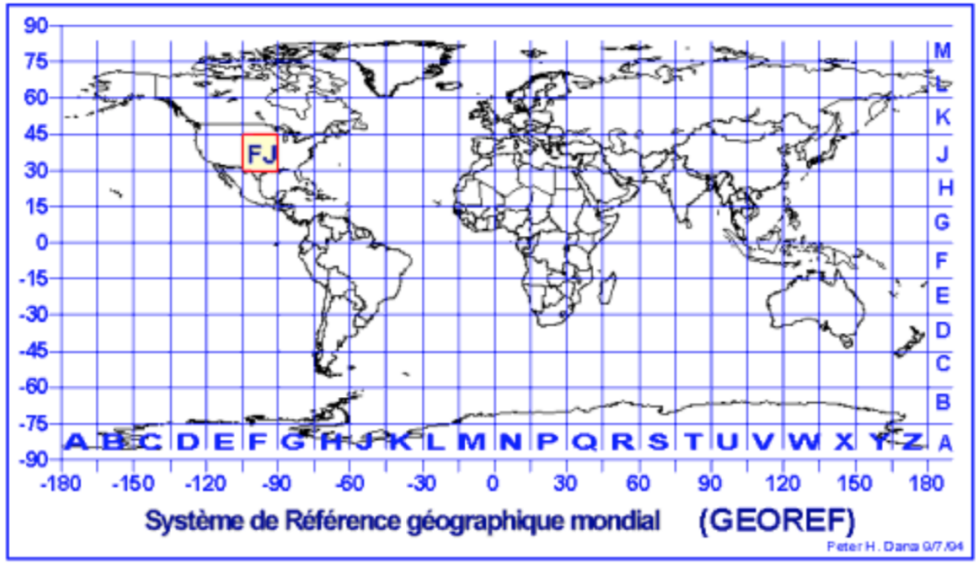
23 mars 2018

CRS (Merci à Nicolas SABY, Unité InfoSol, Orléans, cf. sa présentation atelier RESSTE)

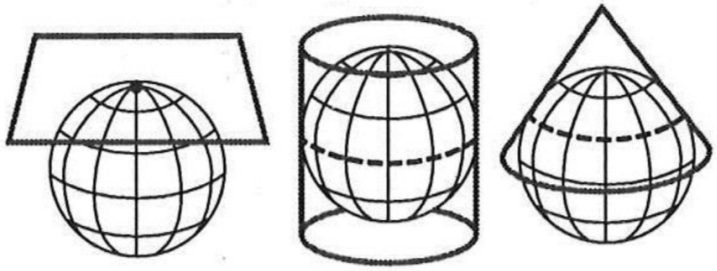
Un mille à l'équateur vaut une minute



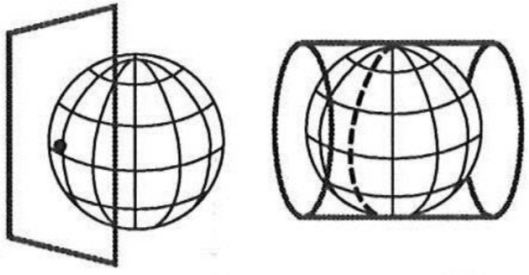




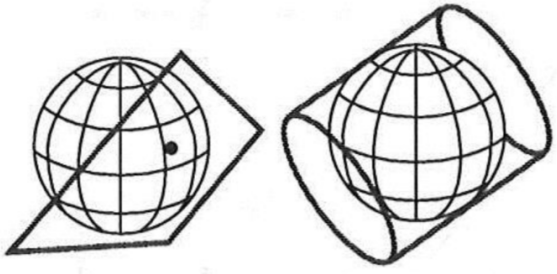
De multiples systèmes de projection



La projection est ***directe :***
la surface de projection est centrée
sur un pôle (projection azimutale),
sur l'équateur (projection cylindrique)
ou sur un parallèle (projection conique).

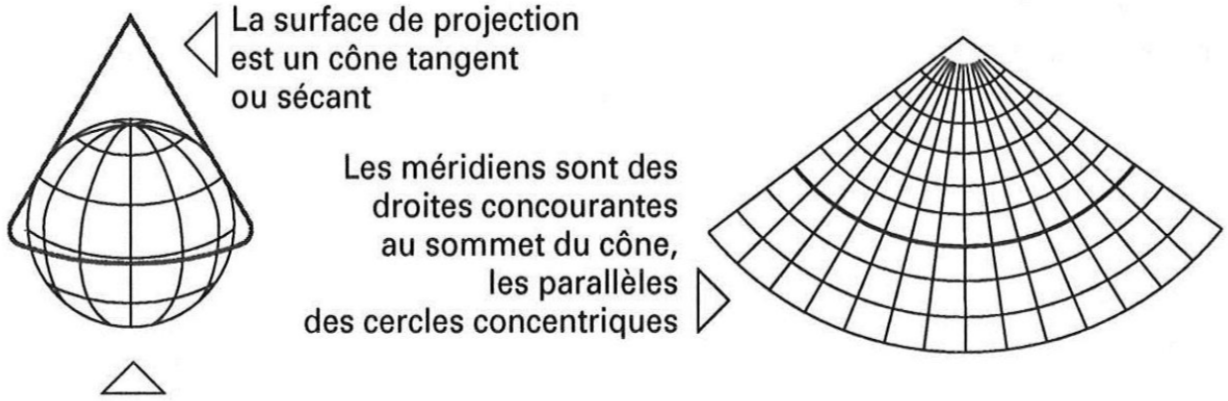


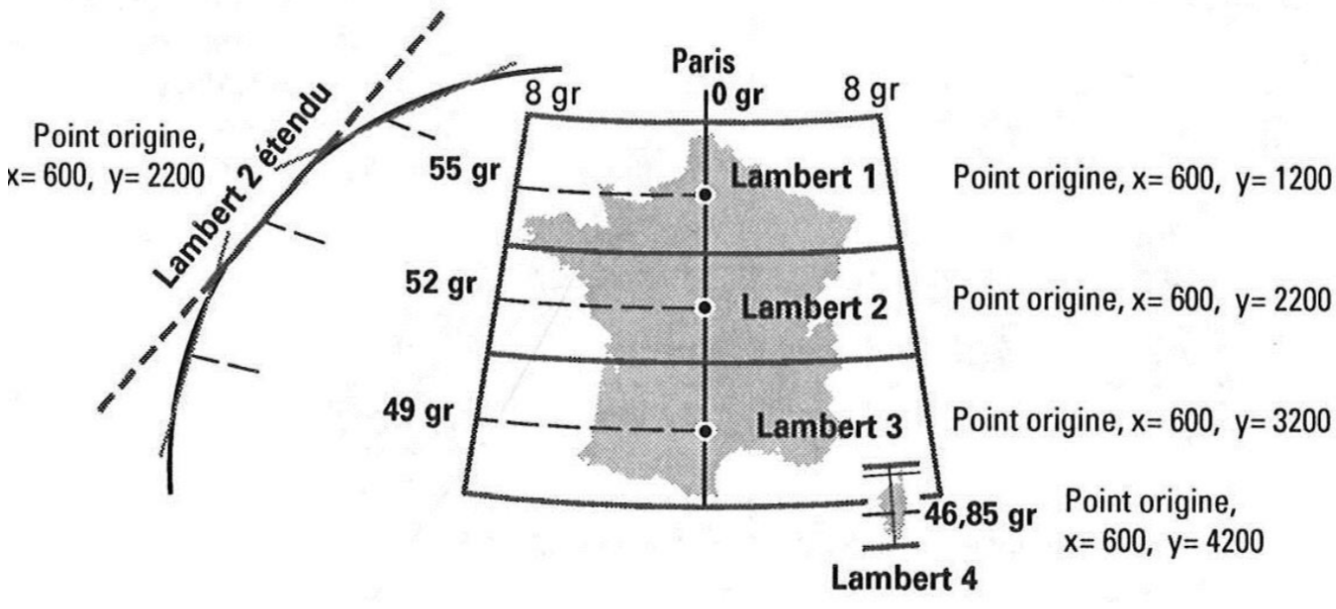
La projection est ***transverse :***
la surface de projection est centrée sur un
point de l'équateur (projection azimutale)
ou sur un méridien (projection cylindrique).



La projection est ***oblique :***
la surface de projection est centrée sur un
point quelconque de la sphère.

La France


La surface de projection
est un cône tangent
ou sécant
Les méridiens sont des
droites concourantes
au sommet du cône,
les parallèles
des cercles concentriques
Le centre de projection est un parallèle (2 parallèles lorsque le cône est sécant)


Paris
8 gr
0 gr
8 gr
Point origine,
x= 600, y= 2200
Lambert 2 étendu
55 gr
52 gr
49 gr
Lambert 1
Lambert 2
Lambert 3
Point origine, x= 600, y= 1200
Point origine, x= 600, y= 2200
Point origine, x= 600, y= 3200
46,85 gr
Point origine,
x= 600, y= 4200
Lambert 4

En R , deux façons théoriques de définir le système de projection

- par le nom du code EPSG, par exemple WGS84 = 4326 (google) ou 2154 (Lambert-93)
- par l'écriture directe du code à considérer dans le slot proj4string

```
CRSargs(CRS("+init=epsg:2154"))
## [1] "+init=epsg:2154 +proj=lcc +lat_1=49 +lat_2=44 +lat_0=46.5 +lon_0=3 +x_0
```

De fait, les objets spatiaux sont souvent dotés d'un CRS initial

```
CRSargs(france@proj4string)
## [1] "+proj=longlat +ellps=WGS84"
```

Si on ne sait, on cherche dans gdal et/ou on va sur le net:
http://spatialreference.org

```
EPSG <- make_EPSG()
EPSG_Lambert <- EPSG [grep("Lambert", EPSG$note), 1:2]
head(EPSG_Lambert)
##     code                                                    note
## 637 2138                          # NAD27(CGQ77) / Quebec Lambert
## 653 2154                                    # RGF93 / Lambert-93
## 654 2155 # American Samoa 1962 / American Samoa Lambert (deprecated)
## 684 2192                  # ED50 / France EuroLambert (deprecated)
## 686 2194 # American Samoa 1962 / American Samoa Lambert (deprecated)
## 809 2318                             # Ain el Abd / Aramco Lambert
```

En R , en pratique il faut **transformer** les coordonnées via le système de projection

- C'est le rôle de la fonction de reprojection **spTransform()**

```
bbox(france)
##          min       max
## x -4.790282  9.562218
## y 41.364927 51.091109
france_Lambert93 <- spTransform(france, CRS("+init=epsg:2154"))
bbox(france_Lambert93)
##          min     max
## x  124535.3 1242296
## y 6049526.6 7110717
```

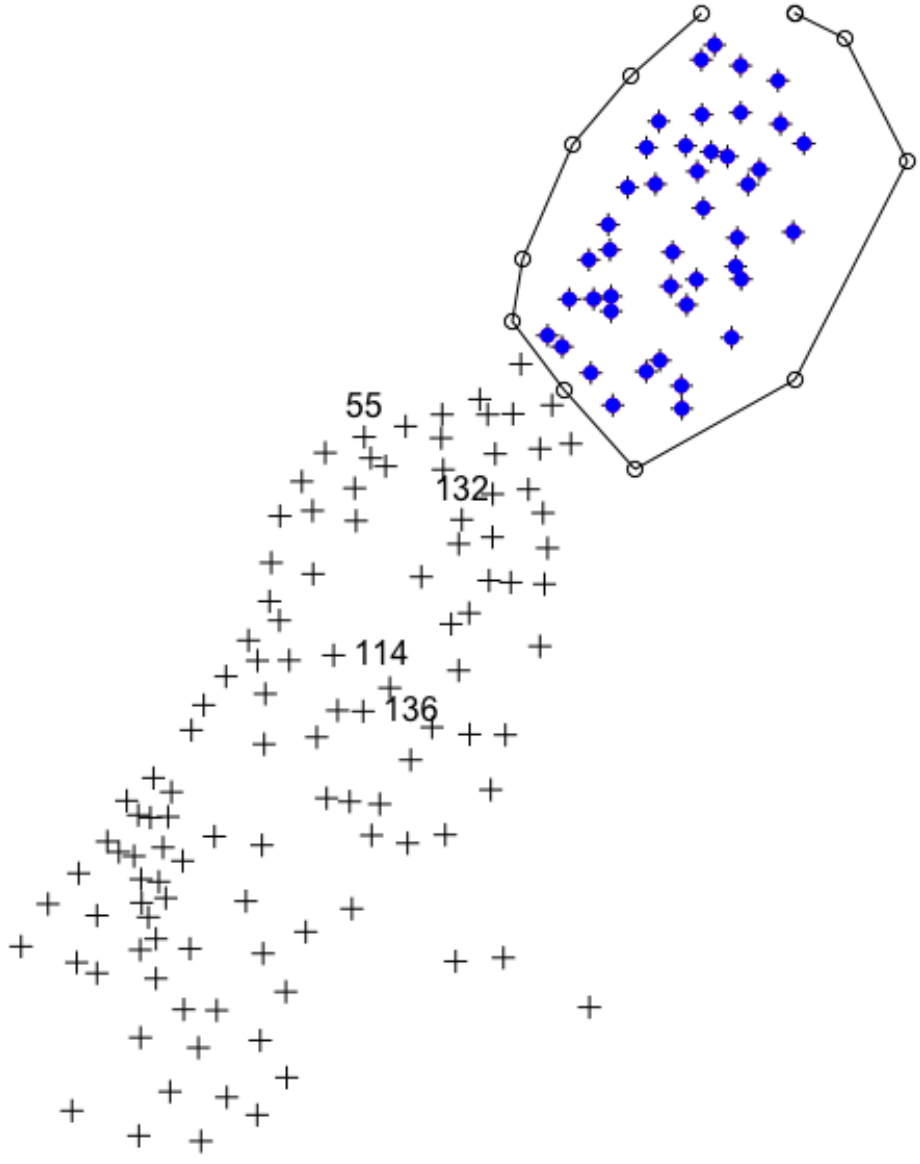
- Cette reprojection est importante pour les calculs de distances (Variogrammes!)
- Exercice: Paris a pour latitude : 48.866667 et pour longitude : 2.333333. Quelles sont ses coordonnées sur une carte Lambert93? Quelles sont ses coordonnées en degrés, minutes, secondes (cf. dd2dms). Celles de Bordeaux sont 48.866667 et 2.333333. Quelle est la distance entre ces deux villes (cf. spDistsN1 et gzAzimuth)
- Exercice: La grille des prévisions CHIMERE est donnée en coordonnées long/lat. Transformer l'objet spatial construit à l'exercice 3 en Lambert93 et visaliser les deux cartes obtenues pour voir l'effet de la projection conique.

Visualisation

Graphe interactif avec locator

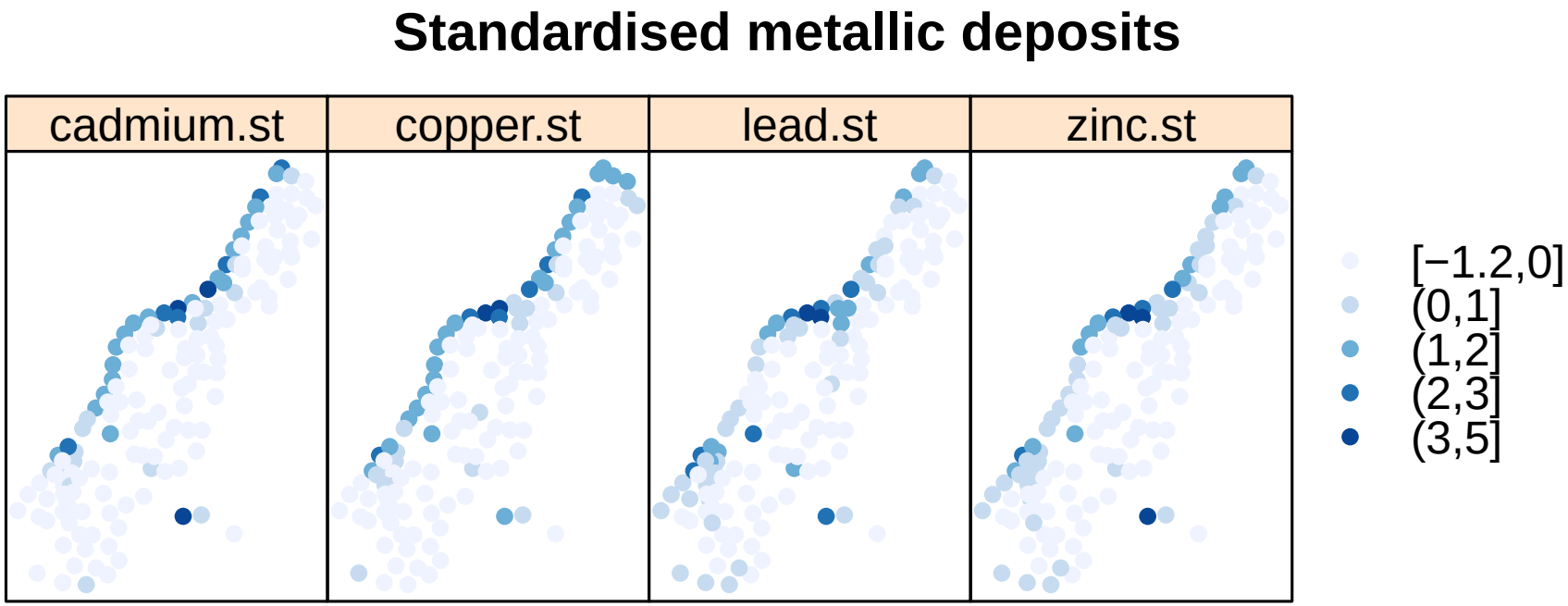
```
plot(meuse_sp)
region <- locator(type="o")
#finir avec Esc

n <- length(region$x)
p <- Polygon(cbind(region$x,region$y)[c(1:n,1),], hole=FALSE)
ps <- Polygons(list(p), ID = "region")
sps <- SpatialPolygons(list(ps))
plot(meuse_sp[sps,], pch=16, cex=1, add=TRUE, col="red")
# Voir ?over
plot(meuse_sp[!is.na(over(meuse_sp,sps)),], pch=16, cex=1, add=TRUE, col="blue"
#
pointschoisis<-identify(coordinates(meuse_sp))
print(pointschoisis)
```

Graph interactif avec locator



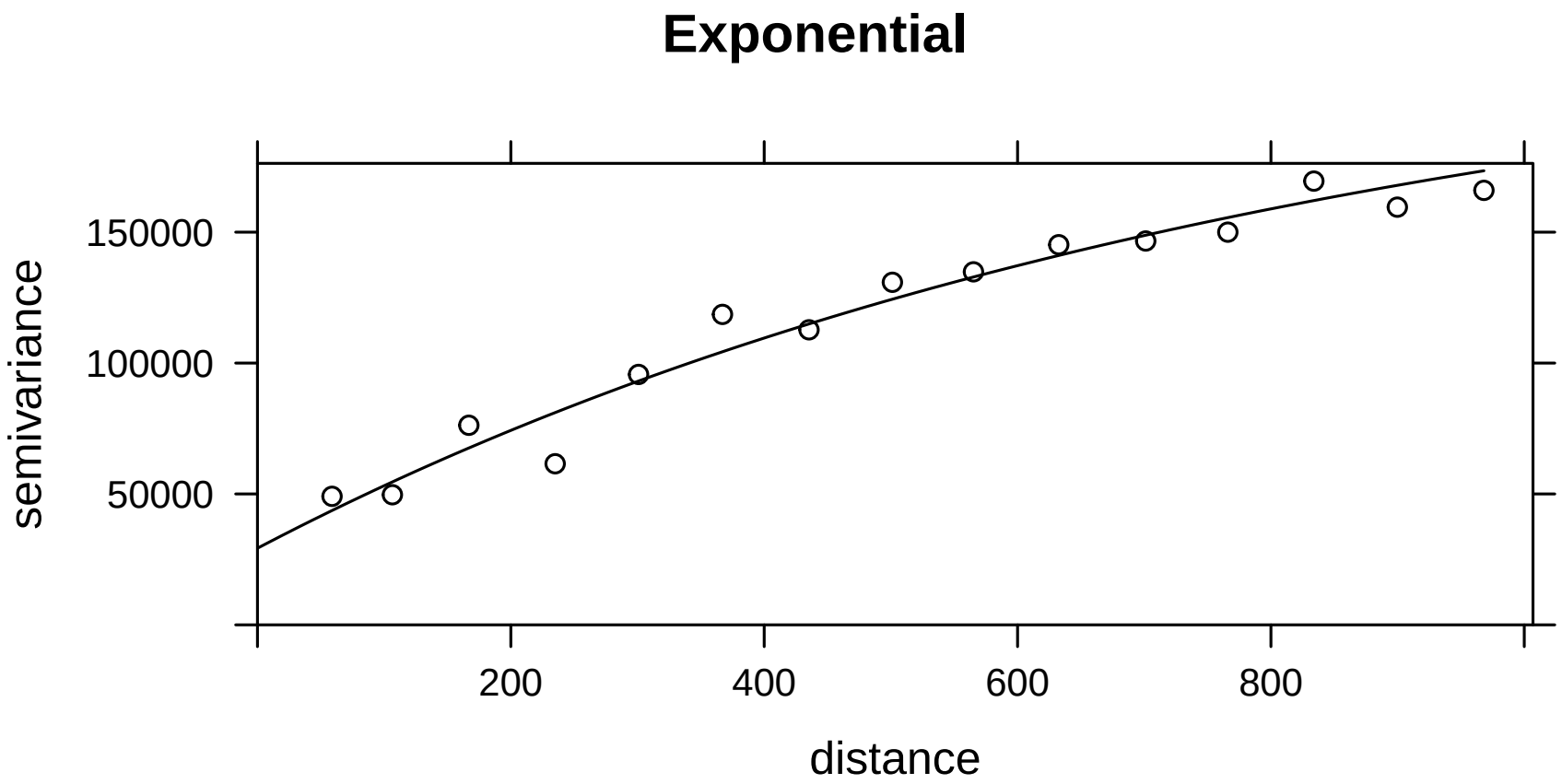
Visualisation treillis avec spplot

```
par(mfrow=c(1,1))
library(RColorBrewer)
bleus <- brewer.pal(7, "Blues")
lead.st <- as.vector(scale(meuse$lead)); zinc.st <- as.vector(scale(meuse$zinc))
copper.st <- as.vector(scale(meuse$copper)); cadmium.st <- as.vector(scale(meuse
meuse.st<-data.frame(x=meuse$x, y=meuse$y,lead.st,zinc.st,copper.st,cadmium.st)
coordinates(meuse.st) <- ~x+y
cuts=c(-1.2,0,1,2,3,5)
spplot(meuse.st, c("cadmium.st", "copper.st", "lead.st", "zinc.st"),
              key.space="right", main = "Standardised metallic deposits", cex =
              cuts = cuts, col.regions=bleus)
```



Visualisation spplot Kriege vs IDW

```
library(gstat)
zn <- krige(zinc~1,meuse_sp,meuse_sg)
## [inverse distance weighted interpolation]
meuse_sg$zincIDW <- zn$var1.pred

vgmMeuse <- variogram(zinc~1, meuse_sp, cutoff=1000)
vgmExpMeuse <- fit.variogram(vgmMeuse, vgm(150000, "Exp", 1000, 40000))
plot(vgmMeuse, vgmExpMeuse, main="Exponential", col='black')
```



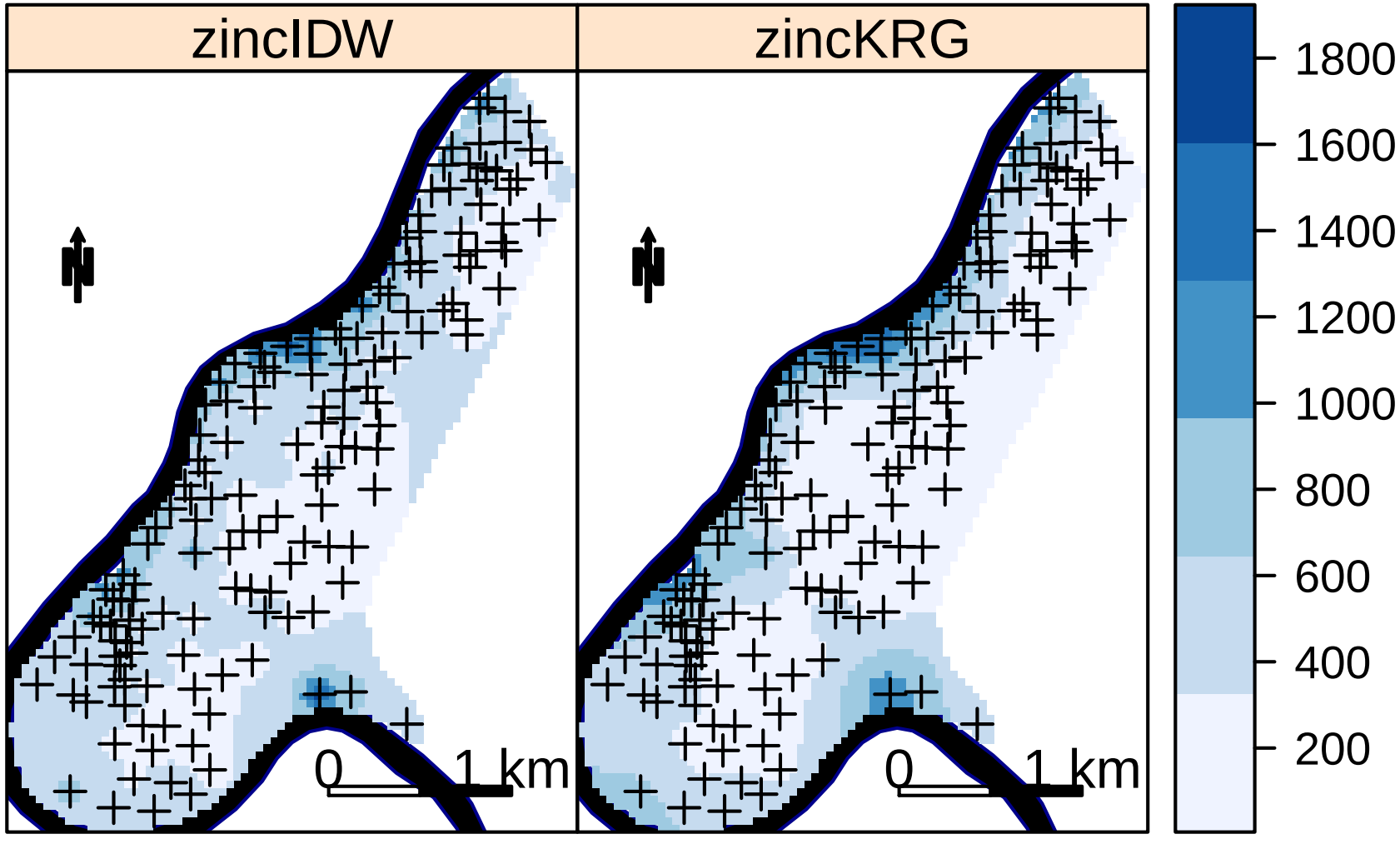

```
znkrige<-krige(zinc~1, meuse_sp,meuse_sg, model=vgmExpMeuse)
```

Construction de sp.layout

```
river <- list("sp.polygons", meuse.riv_poly, col='darkblue', fill=T)
north <- list("SpatialPolygonsRescale", layout.north.arrow(), offset =
                c(178750,332500),
              scale = 400)
scale <- list("SpatialPolygonsRescale", layout.scale.bar(), offset =
                c(180200, 329800), scale = 1000, fill=c("transparent","black"))
txt1 <- list("sp.text", c(180200, 329950), "0")
txt2 <- list("sp.text", c(181200, 329950), "1 km")
pts <- list("sp.points", meuse_sp, pch = 3, col = "black")
meuse.layout <- list(river, north, scale, txt1, txt2, pts)
```

Visualisation sp.layout+spplot

```
meuse.layout <- list(river, north, scale, txt1, txt2, pts)
spplot(meuse_sg, c("zincIDW", "zincKRG"), sp.layout = meuse.layout, cuts=5, col
```



Pistes pour le futur

1. Explorer le **package `sf`** qui associe les fonctionalités de `sp`, `rgdal`, et `rgeos` en un seul package *spatial features* fondé sur le principe `tidyverse`.
2. Les packages **outils** `rgeos`,`rgdal`,`maps`,`maptools`... Qui voudrait faire un tour d'horizon? La connection avec les outils SIG?
3. Continuer de comprendre la modélisation **geostatistique** et son inférence. L'étude des dépendances dans le modèle multinormal (d'observation ou latent) peut être approché sous l'angle de la:

- **matrice de variance** qui analyse les corrélations spatiales sous forme de variogrammes (packages: structuré sur `sp`: `gstat`, ou indépendants: `spatial`,`field`,`RandomFields`, `GeoR`, etc.)
- **matrice de précision** qui décrit les indépendances conditionnelles (voir par exemple le package structuré sur `sp`: `spdep`) ou son utilisation pour les modèles géostatistiques complexes `INLA`)

4. **Extensions de la géostatistique** qui intéresseront le réseau RESSTE

- **Le spatio-temporel** (packages: structuré sur `sp`: `spacetime`) sp+xts+data.frame=spacetime
- **Approche Bayesiennes** (réplicats MCMC de structure de type cartes)