

Special cases in geostatistics

Spatial Statistics

Madlene Nussbaum

m.nussbaum@uu.nl

Utrecht University, Department of Physical Geography

Geostatistical model: 2 special cases

There are many special cases for which the geostatistical model has been expanded. Let's have a look at two of them:

- 2 special cases to expand the geostatistical model
 - lognormal kriging
 - anisotropy

pro memoria: Model for Gaussian spatial data

Model for data: $Y_i = S(\mathbf{x}_i) + Z_i = \mu(\mathbf{x}_i) + E(\mathbf{x}_i) + Z_i$

- with Y_i : i^{th} datum; $S(\mathbf{x}_i)$: “signal” (true quantity) at location $\mathbf{x}_i$; $\mu(\mathbf{x}_i)$: trend
- $\{E(\mathbf{x}_i)\}$: a zero-mean Gaussian process, parametrized by covariance function $\gamma(\mathbf{h}; \theta)$ or variogram $V(\mathbf{h}; \theta)$
- Z_i : iid Gaussian measurement error with variance τ^2

Trend $\mu(\mathbf{x}_i)$ modeled by linear regression model with spatial covariates $d_k(\mathbf{x}_i)$

$$\mu(\mathbf{x}_i) = \sum_k d_k(\mathbf{x}_i) \beta_k = \mathbf{d}(\mathbf{x}_i)^T \boldsymbol{\beta}$$

1 Lognormal universal kriging

Gaussian model fitted to log-transformed response variable $Y(\mathbf{x}) = \log(U(\mathbf{x}))$
(e.g. often pollution datasets)

⇒ Computing UK predictions for log-transformed response

⇒ How should we back-transform to original scale of response?

Lognormal distribution

$$Y = \log(U) \sim \mathcal{N}(\mu_Y, \sigma_Y^2)$$

Expectation and variance of U

$$\begin{aligned}\mathbb{E}[U] &= \mu_U = \exp(\mu_Y + 0.5 \sigma_Y^2) \\ \text{Var}[U] &= \mu_U^2 (\exp(\sigma_Y^2) - 1)\end{aligned}$$

Backtransformation for lognormal universal kriging

$\exp(\hat{S}_k(\mathbf{x}'))$ is a biased predictor of $U(\mathbf{x}')$

Unbiased back-transformation

$$\hat{U}_{lk}(\mathbf{x}') = \exp\left(\hat{S}_k(\mathbf{x}') + 0.5 \left\{ \text{Var}[S(\mathbf{x}')] - \text{Var}[\hat{S}_k(\mathbf{x}')] \right\}\right)$$

Limits of prediction intervals can be back-transformed directly by $\exp()$.

Back-transformation implemented in function `lgnpp` of R package `georob`.

Example: Lognormal UK Meuse **zinc** data

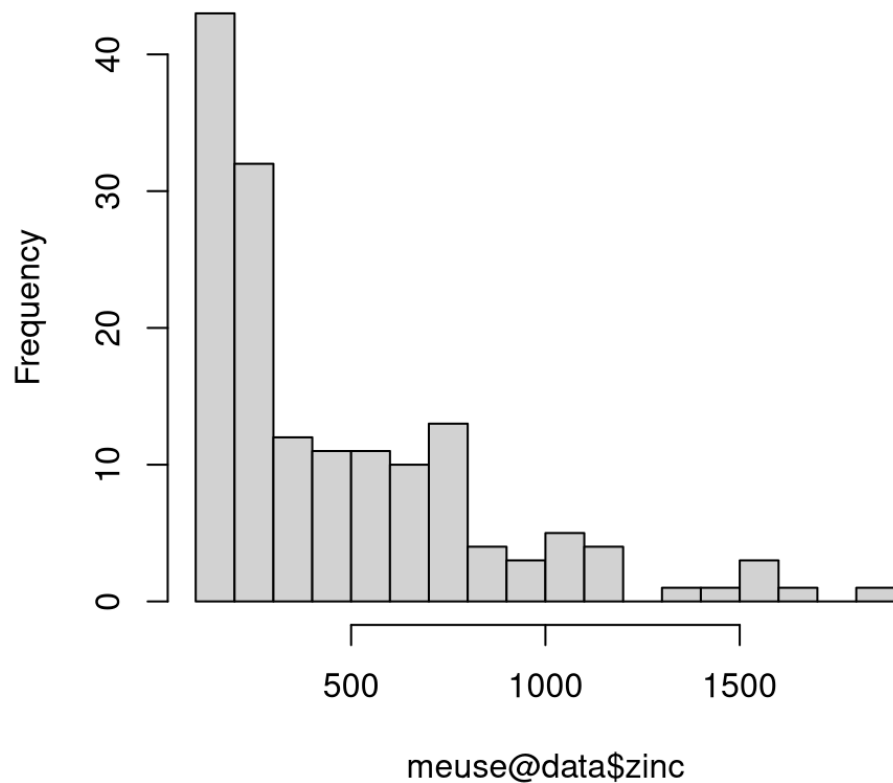
```
1 library(georob)
2 library(mapview)
3 data(meuse)
4 coordinates(meuse) <- ~x+y
5 proj4string(meuse) <- CRS("+init=epsg:28992")
6 mapview(meuse, zcol = "zinc", map.types = "Esri.WorldImagery")
```



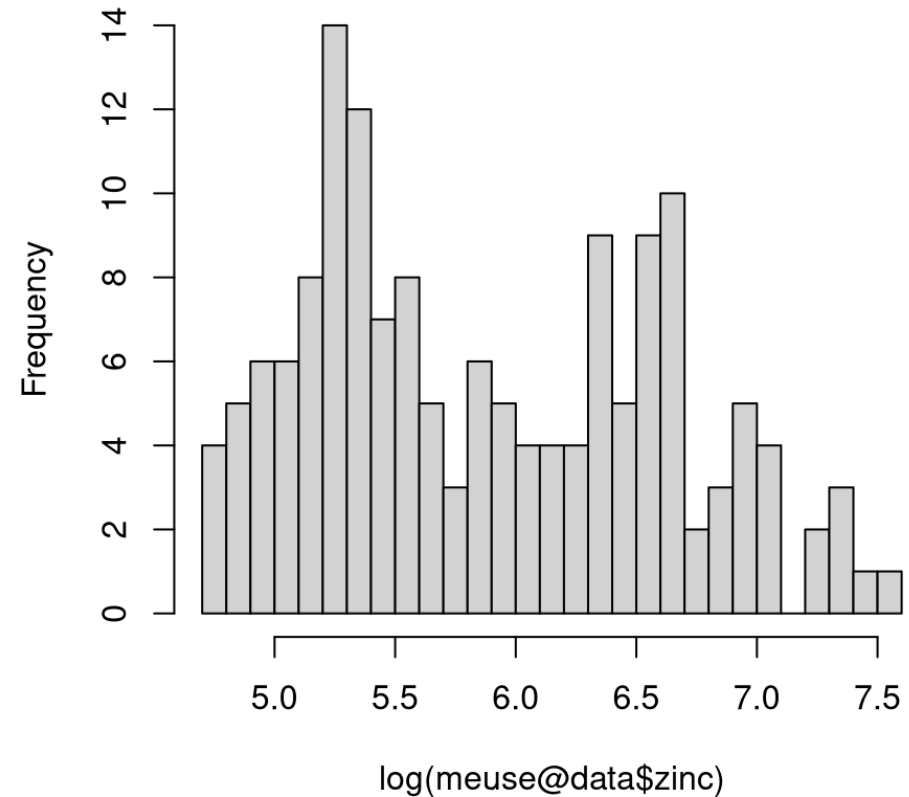
Example Meuse **zn**: Response distribution

```
1 par(mfrow=c(1,2))
2 hist(meuse@data$zinc, breaks = 20)
3 hist(log(meuse@data$zinc), breaks = 20)
```

Histogram of meuse@data\$zinc

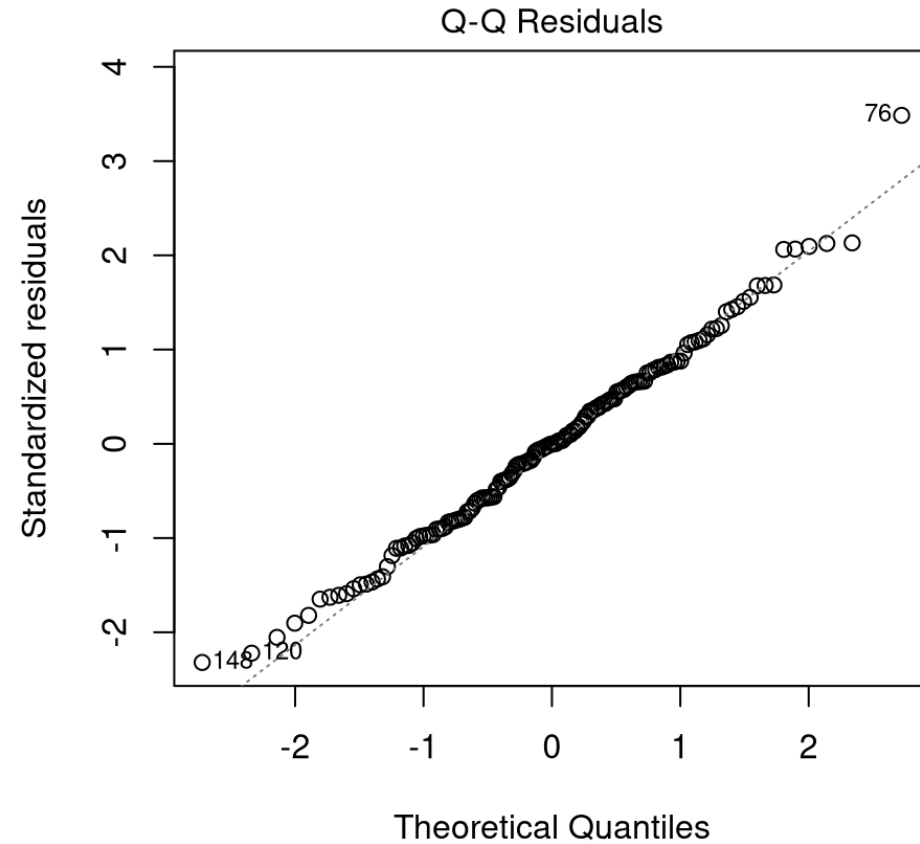
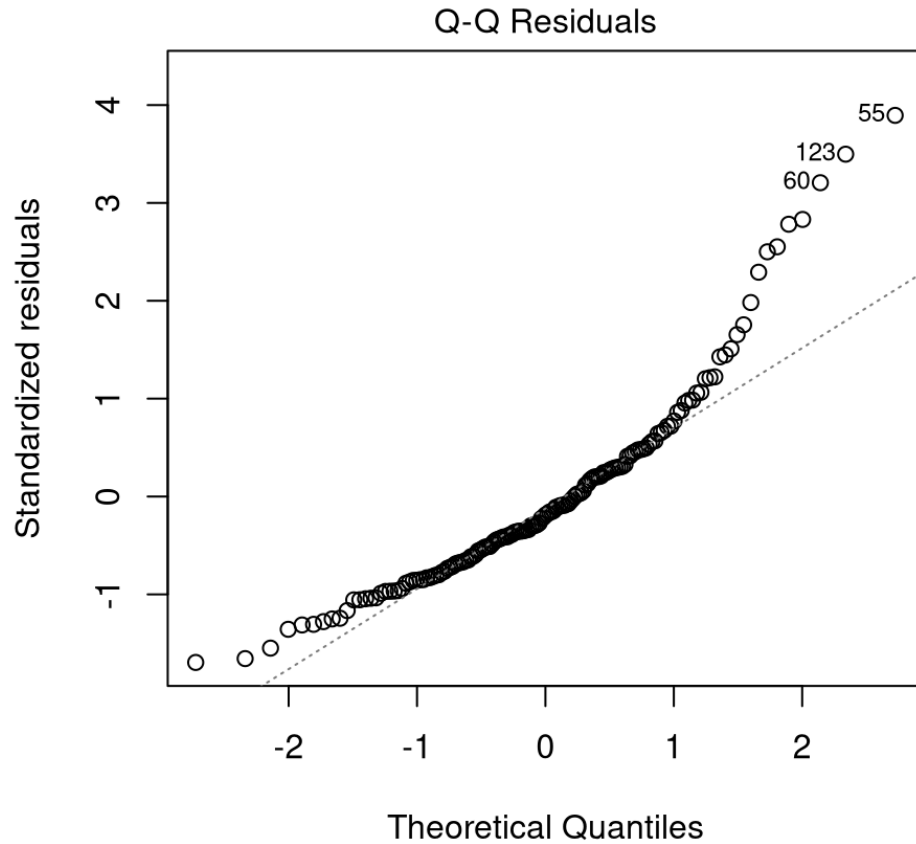


Histogram of log(meuse@data\$zinc)



Example Meuse **zn**: OLS residuals

```
1 par(mfrow = c(1,2))  
2 plot(lm(zinc ~ dist, meuse), 2)  
3 plot(lm(log(zinc) ~ dist, meuse), 2)
```



Example Meuse **zn**: REML fit

```
1 r.logzn <- georob(log(zinc)~sqrt(dist), meuse, locations=~x+y,  
2                 variogram.model="RMexp",  
3                 param=c(variance=0.15, nugget=0.05, scale=200),  
4                 tuning.psi=1000, control=control.georob(  
5                 cov.bhat=TRUE, cov.bhat.betahat=TRUE))  
6 r.logzn
```

Tuning constant: 1000

Fixed effects coefficients:

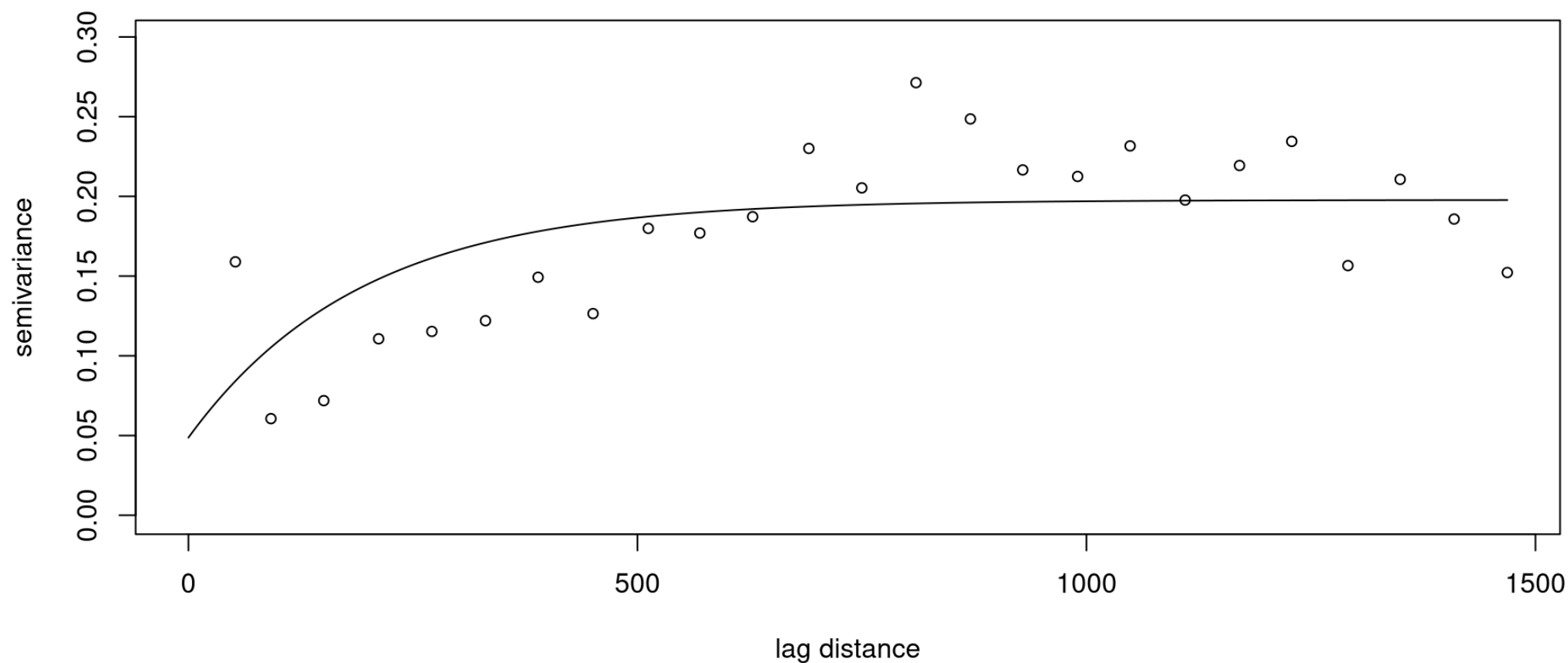
(Intercept)	sqrt(dist)
6.985	-2.567

Variogram: RMexp

variance	snugget(fixed)	nugget	scale
0.14910	0.00000	0.04867	192.52854

Example Meuse **zn**: REML variogram

```
1 plot(r.logzn, lag.dist.def=60, max.lag=1500)
```



Example Meuse **zn**: Universal kriging prediction

```
1 data(meuse.grid)
2 coordinates(meuse.grid) <- ~x+y
3 gridded(meuse.grid) <- TRUE
4 r.luk <- predict(r.logzn, newdata=meuse.grid,
5   control=control.predict.georob(extended.output=TRUE))
6 str(r.luk@data)
```

```
'data.frame':  3103 obs. of  8 variables:
 $ pred      : num  7.03 7.05 6.75 6.49 7.07 ...
 $ se        : num  0.362 0.335 0.342 0.349 0.288 ...
 $ lower     : num  6.32 6.39 6.08 5.8 6.5 ...
 $ upper     : num  7.73 7.7 7.42 7.17 7.63 ...
 $ trend     : num  6.99 6.99 6.7 6.45 6.99 ...
 $ var.pred  : num  0.0388 0.0539 0.0437 0.0357 0.0787 ...
 $ cov.pred.target: num  0.0285 0.0453 0.0379 0.0315 0.0724 ...
 $ var.target : num  0.149 0.149 0.149 0.149 0.149 ...
 - attr(*, "variogram.object")=List of 1
 ..$ :List of 9
 .. ..$ variogram.model: chr "RMexp"
 .. ..$ param          : Named num [1:4] 0.1491 0 0.0487 192.5285
 .. .. ..- attr(*, "names")= chr [1:4] "variance" "snugget" "nugget" "scale"
 .. ..$ fit.param      : Named logi [1:4] TRUE FALSE TRUE TRUE
 .. .. ..- attr(*, "names")= chr [1:4] "variance" "snugget" "nugget" "scale"
 .. ..$ isotropic      : logi TRUE
 .. ..$ aniso          : Named num [1:5] 1 1 90 90 0
 .. .. ..- attr(*, "names")= chr [1:5] "f1" "f2" "omega" "phi" ...
 .. ..$ fit.aniso      : Named logi [1:5] FALSE FALSE FALSE FALSE FALSE
 .. .. ..- attr(*, "names")= chr [1:5] "f1" "f2" "omega" "phi" ...
 .. ..$ sincos         :List of 6
 .. .. ..$ co: num 6.12e-17
 .. .. ..$ so: num 1
 .. .. ..$ cp: num 6.12e-17
 .. .. ..$ sp: num 1
 .. .. ..$ cz: num 1
 .. .. ..$ sz: num 0
 .. .. ..$ rotmat      : num [1:2, 1:2] 1 0 0 6.12e-17 6.12e-17 1 0 0
```

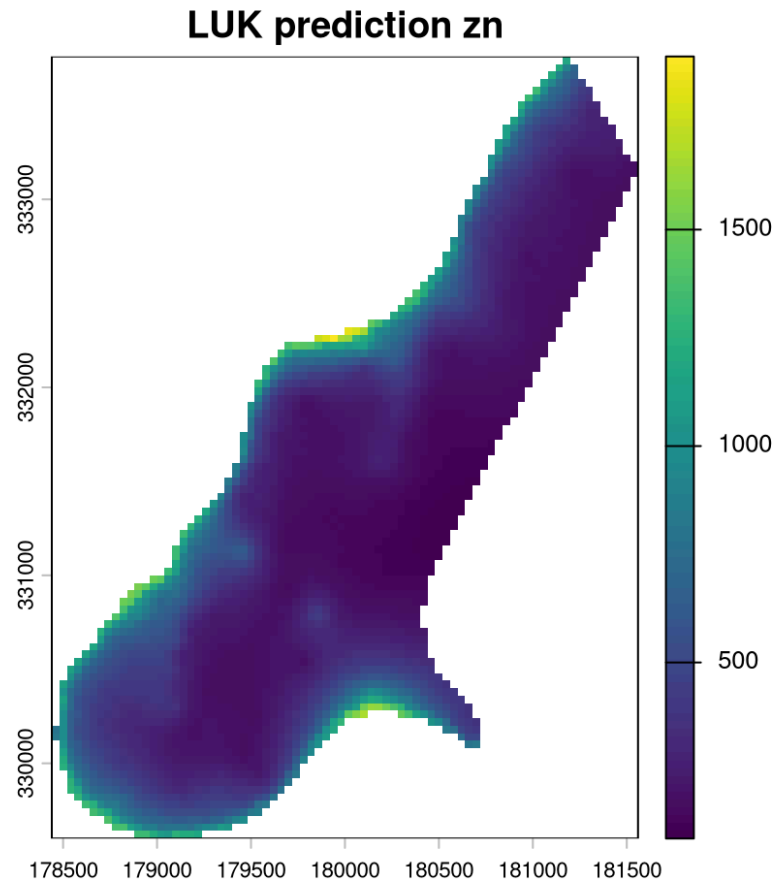
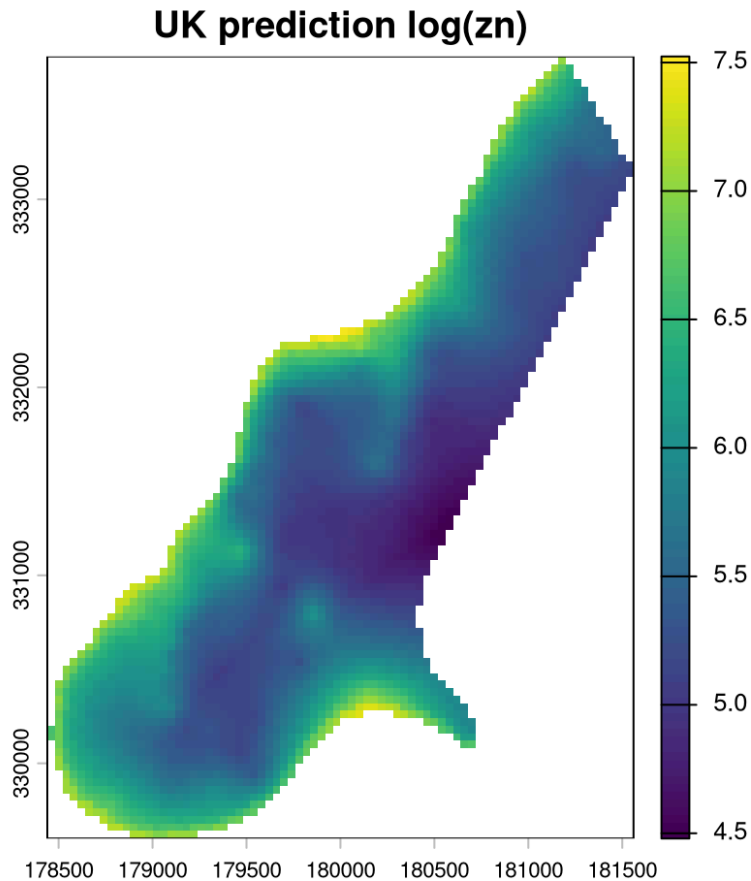
Example Meuse **zn**: Unbiased backtransformation

```
1 r.luk <- lgnpp(r.luk)
2 str(r.luk@data)
```

```
'data.frame': 3103 obs. of 12 variables:
 $ pred      : num  7.03 7.05 6.75 6.49 7.07 ...
 $ se        : num  0.362 0.335 0.342 0.349 0.288 ...
 $ lower     : num  6.32 6.39 6.08 5.8 6.5 ...
 $ upper     : num  7.73 7.7 7.42 7.17 7.63 ...
 $ trend     : num  6.99 6.99 6.7 6.45 6.99 ...
 $ var.pred  : num  0.0388 0.0539 0.0437 0.0357 0.0787 ...
 $ cov.pred.target: num  0.0285 0.0453 0.0379 0.0315 0.0724 ...
 $ var.target : num  0.149 0.149 0.149 0.149 0.149 ...
 $ lgn.pred   : num  1189 1204 902 694 1215 ...
 $ lgn.se     : num  440 409 314 249 354 ...
 $ lgn.lower  : num  554 595 438 331 667 ...
 $ lgn.upper  : num  2286 2215 1673 1299 2064 ...
- attr(*, "variogram.object")=List of 1
 ..$ :List of 9
 .. ..$ variogram.model: chr "RMexp"
 .. ..$ param          : Named num [1:4] 0.1491 0 0.0487 192.5285
 .. .. ..- attr(*, "names")= chr [1:4] "variance" "snugget" "nugget" "scale"
 .. ..$ fit.param      : Named logi [1:4] TRUE FALSE TRUE TRUE
 .. .. ..- attr(*, "names")= chr [1:4] "variance" "snugget" "nugget" "scale"
 .. ..$ isotropic      : logi TRUE
 .. ..$ aniso          : Named num [1:5] 1 1 90 90 0
 .. .. ..- attr(*, "names")= chr [1:5] "f1" "f2" "omega" "phi" ...
 .. ..$ fit.aniso      : Named logi [1:5] FALSE FALSE FALSE FALSE FALSE
 .. .. ..- attr(*, "names")= chr [1:5] "f1" "f2" "omega" "phi" ...
 .. ..$ sincos         :List of 6
 .. .. ..$ co: num 6.12e-17
 .. .. ..$ so: num 1
```

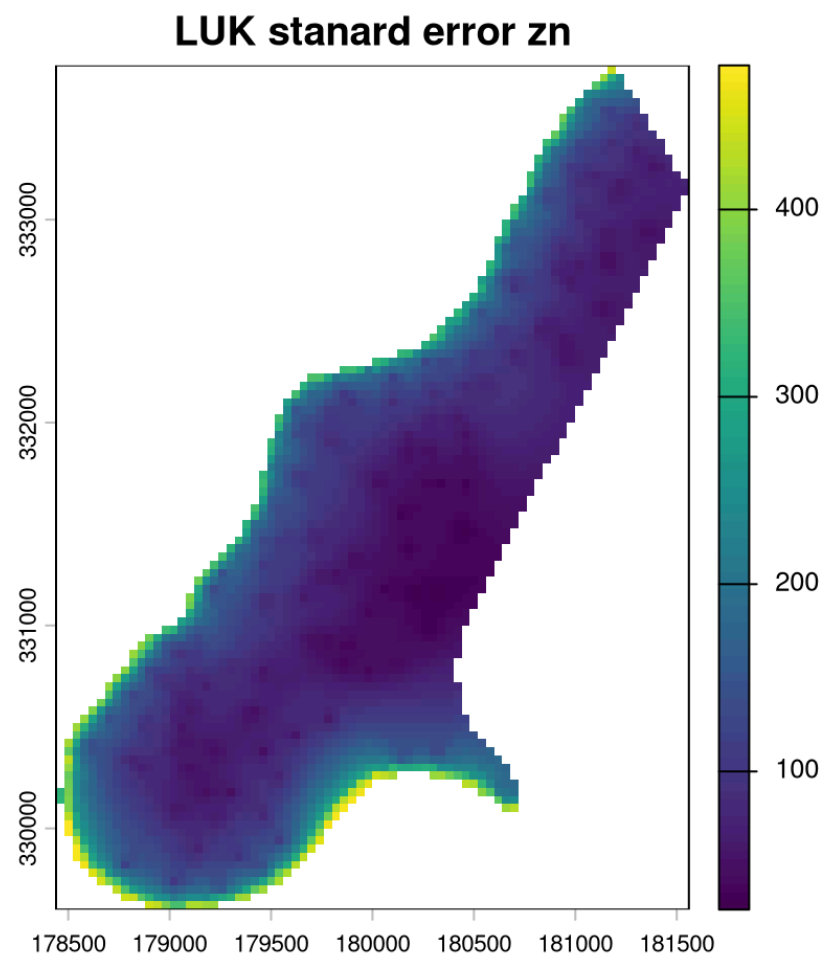
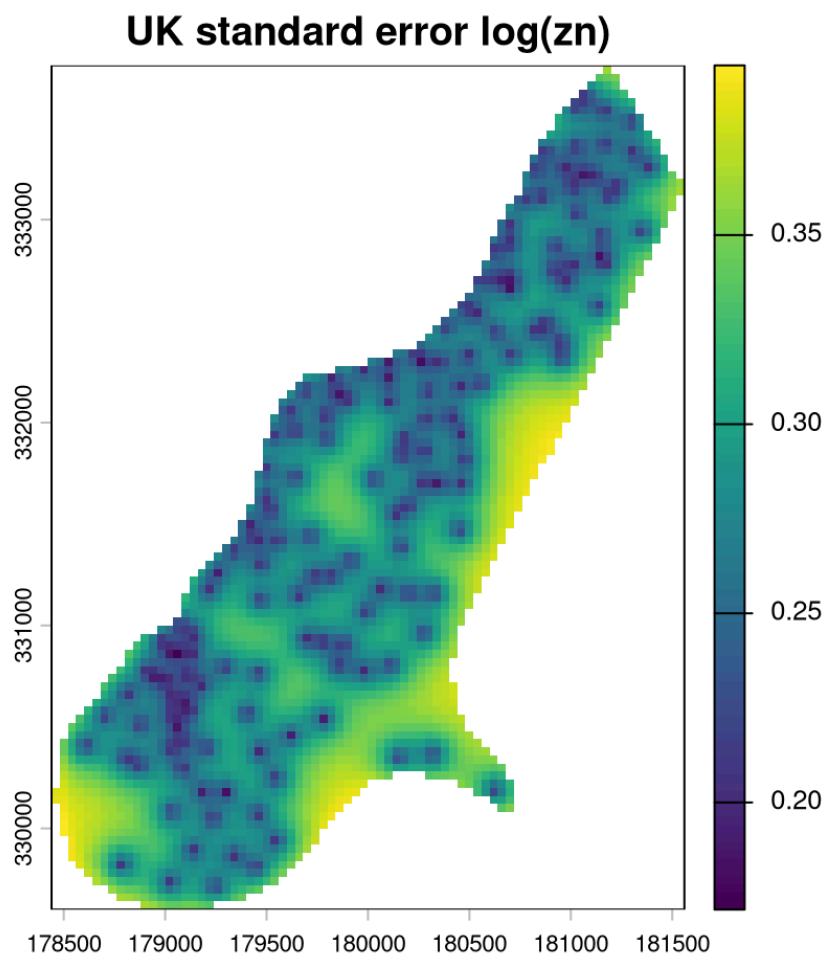
Example Meuse **zn**: Predicted maps

```
1 library(terra)
2 r.luk <- rast(r.luk)
3 par(mfrow=c(1,2))
4 plot(r.luk["^pred$"], main = "UK prediction log(zn)")
5 plot(r.luk["lgn.pred"], main = "LUK prediction zn")
```



Example Meuse **zn**: Predicted standard errors

```
1 par(mfrow=c(1,2))
2 plot(r.luk["^se$"], main = "UK standard error log(zn)")
3 plot(r.luk["lgn.se"], main = "LUK stanard error zn")
```



2 Anisotropic variograms

Spatial auto-correlation can vary from one direction to the other, i.e. it can be anisotropic.

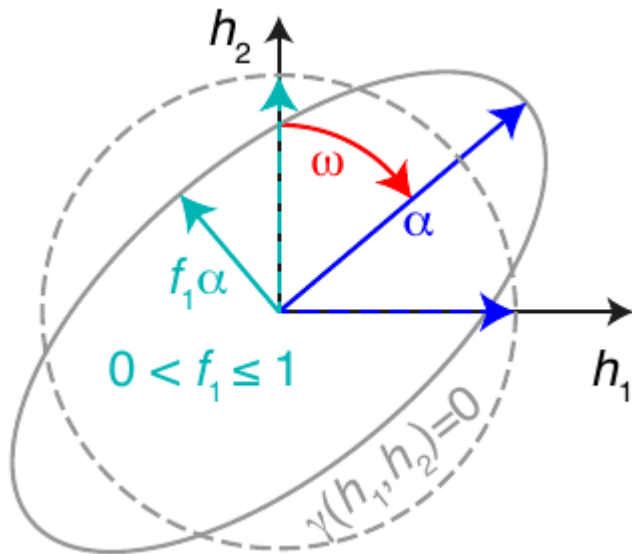
In many instances the anisotropy is such that it could be made isotropic by a simple linear transformation of the spatial coordinates. $\Rightarrow$ geometrically anisotropic variogram function

Idea: rotate and stretch/shrink components of $\mathbf{x}$ such that the stochastic process is isotropic in the transformed coordinate system

$$V(h^*) = V\left(\sqrt{(\mathbf{h}\mathbf{A})^T \mathbf{A}\mathbf{h}}\right)$$

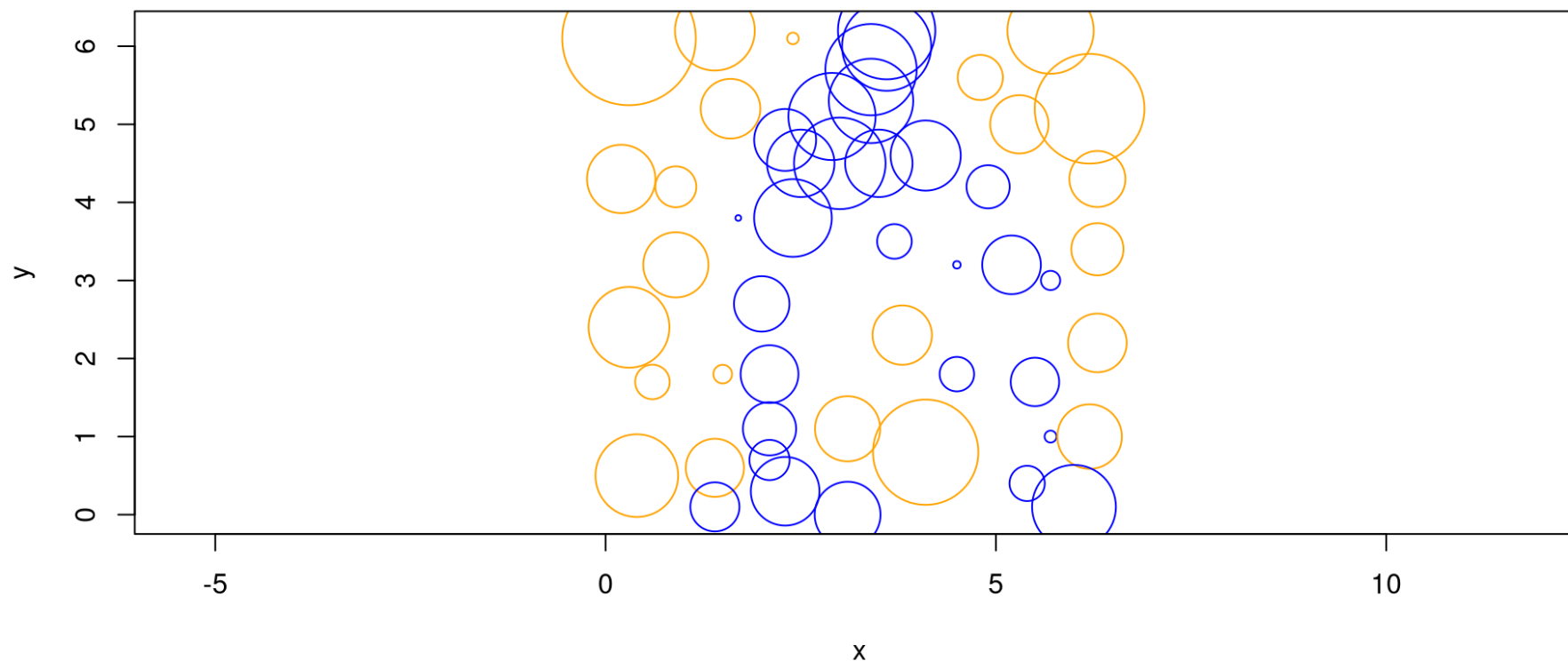
Iso-semivariances contours are ellipsoids in space of non-transformed coordinates and are mapped by transformation:

$$\mathbf{A} = \begin{bmatrix} 1/\alpha & 0 \\ 0 & 1/(f_1\alpha) \end{bmatrix} \begin{bmatrix} \cos(\omega) & \sin(\omega) \\ -\sin(\omega) & \cos(\omega) \end{bmatrix}$$



Example: elevation data

```
1 data(elevation, package="georob")
2 d.elevation <- elevation
3 d.elevation$res <- residuals(lm(height~y, d.elevation))
4 plot(y~x, d.elevation, cex=sqrt(abs(res)), asp=1,
5      col=c("blue", NA, "orange")[sign(res)+2])
```



Example: elevation data – directional sample variogram

```
1 r.sv <- sample.variogram(d.elevation$res,  
2   locations=as.matrix(d.elevation[, c("x","y")]),  
3   lag.dist.def=0.5, xy.angle.def=c(0, 45, 135, 180))  
4 summary(r.sv)
```

Sample variogram estimator: qn

Summary of lag distances

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
0.3812	2.2244	4.2278	4.1295	6.1756	8.2759

Summary of number of pairs per lag and distance classes

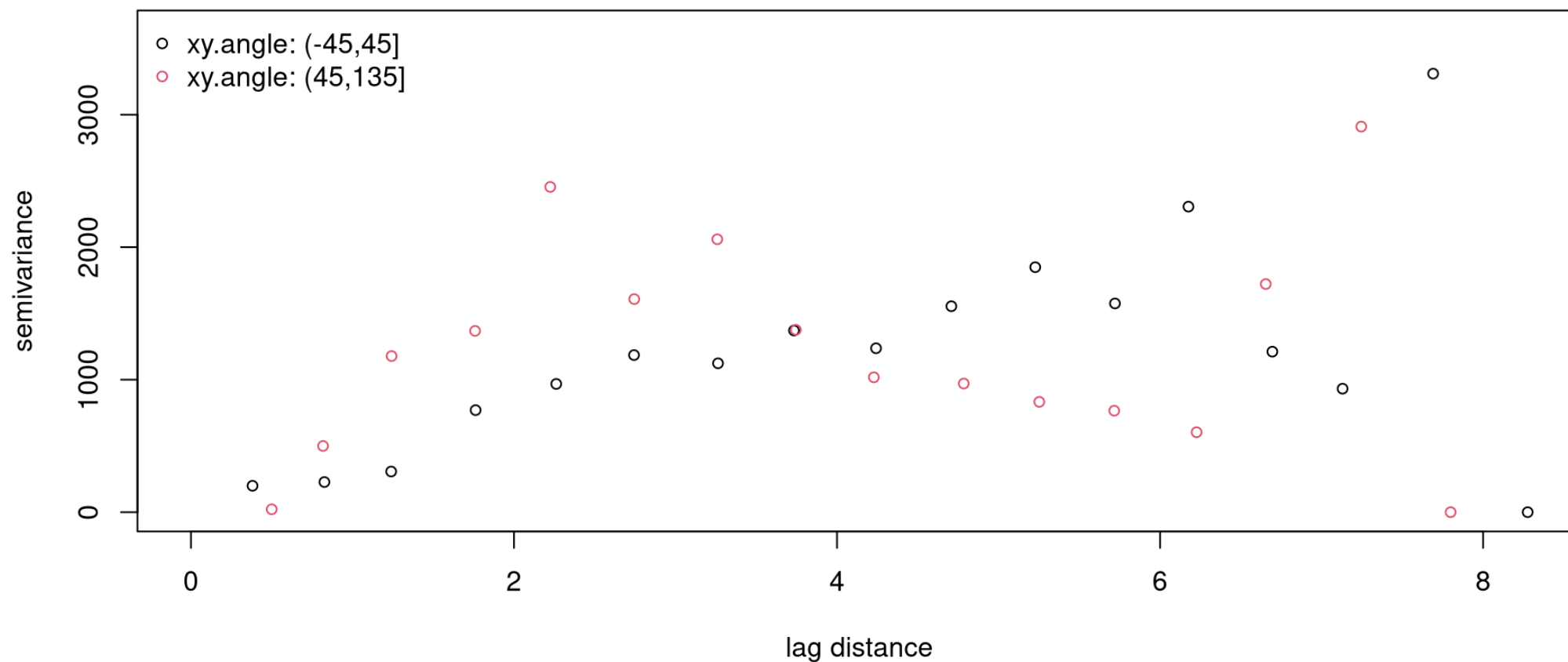
Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
1.00	11.00	45.00	40.18	63.00	73.00

Angle classes in xy-plane: (-45,45] (45,135]

Angle classes in xz-plane: [0,180]

Example: elevation data - directional sample variogram

```
1 plot(r.sv)
```



Example: elevation data – fitted directional variogram

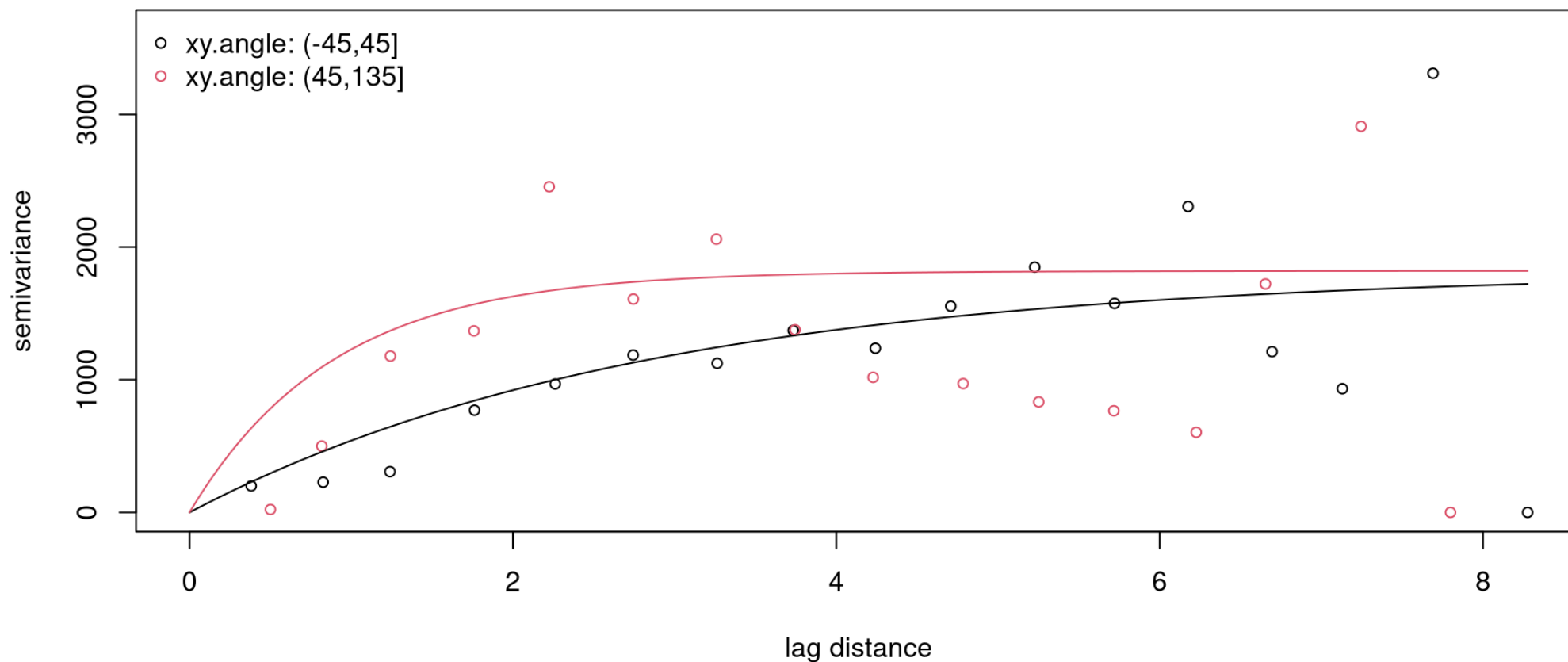
```
1 r.exp <- fit.variogram.model(r.sv,  
2   variogram.model="RMexp",  
3   param=c(variance=1500, nugget=10, scale=4),  
4   aniso = default.aniso(f1 = 0.4, omega = 0),  
5   fit.aniso = default.fit.aniso(f1 = TRUE, omega = TRUE))  
6 r.exp
```

Variogram: RMexp

variance	snugget(fixed)	nugget	scale
1.820e+03	0.000e+00	1.289e-07	1.256e+02
f1	f2(fixed)	omega	phi(fixed)
0.006757	1.000000	17.415945	90.000000
zeta(fixed)			
0.000000			

Example: elevation data - fitted directional variogram

```
1 plot(r.sv)  
2 lines(r.exp, xy.angle=c(0, 90))
```





**Utrecht
University**

Sharing science,
shaping tomorrow