Semivariogram example - Meuse river sediments

- This illustration uses the Meuse dataset included in gstat which comprises of heavy metals measured in the top soil in a flood plain along the river Meuse.
- Data set gives locations and topsoil heavy metal concentrations, along with a number of soil and landscape variables at the observation locations, collected in a flood plain of the river Meuse, near the village of Stein (NL).
- Heavy metal concentrations are from composite samples of an area of approximately 15 m x 15 m. (recorded sample location is a single point, yet each actual sample may cover approx. 15x15 m).
- Total area covered by sampling fits into a rectangle 2.8 km (x-axis) by 3.9 km (y-axis) approximately.
- Apparently, polluted sediment is carried by the river and mostly deposited close to the river bank.

```
str(meuse)
```

```
## 'data.frame':    155 obs. of  14 variables:
##  $ x       : num  181072 181025 181165 181298 181307 ...
##  $ y       : num  333611 333558 333537 333484 333330 ...
##  $ cadmium: num  11.7 8.6 6.5 2.6 2.8 3 3.2 2.8 2.4 1.6 ...
##  $ copper : num  85 81 68 81 48 61 31 29 37 24 ...
##  $ lead    : num  299 277 199 116 117 137 132 150 133 80 ...
##  $ zinc    : num  1022 1141 640 257 269 ...
##  $ elev    : num  7.91 6.98 7.8 7.66 7.48 ...
##  $ dist    : num  0.00136 0.01222 0.10303 0.19009 0.27709 ...
##  $ om      : num  13.6 14 13 8 8.7 7.8 9.2 9.5 10.6 6.3 ...
##  $ ffreq   : Factor w/ 3 levels "1","2","3": 1 1 1 1 1 1 1 1 1 1 ...
##  $ soil    : Factor w/ 3 levels "1","2","3": 1 1 1 2 2 2 2 1 1 2 ...
##  $ lime    : Factor w/ 2 levels "0","1": 2 2 2 1 1 1 1 1 1 1 ...
##  $ landuse: Factor w/ 15 levels "Aa","Ab","Ag",..: 4 4 4 11 4 11 4 2 2 15 ...
##  $ dist.m : num  50 30 150 270 380 470 240 120 240 420 ...
```

Coordinates

Function `coordinates()`: promotes the data.frame meuse into a SpatialPointsDataFrame which knows about its spatial coordinates.

- `x` and `y` coordinates are in meters.
  - `x` a numeric vector; Easting (m) in Rijksdriehoek (RDH) (Netherlands topographical) map coordinates
  - `y` a numeric vector; Northing (m) in RDH coordinates

```
coordinates(meuse) = ~x+y
class(meuse)
```

```
## [1] "SpatialPointsDataFrame"
## attr(,"package")
## [1] "sp"
```

```
str(meuse)
```

```
## Formal class 'SpatialPointsDataFrame' [package "sp"] with 5 slots
##   ..@ data       :'data.frame':  155 obs. of  12 variables:
##   .. ..$ cadmium: num [1:155] 11.7 8.6 6.5 2.6 2.8 3 3.2 2.8 2.4 1.6 ...
##   .. ..$ copper : num [1:155] 85 81 68 81 48 61 31 29 37 24 ...
##   .. ..$ lead   : num [1:155] 299 277 199 116 117 137 132 150 133 80 ...
##   .. ..$ zinc   : num [1:155] 1022 1141 640 257 269 ...
##   .. ..$ elev   : num [1:155] 7.91 6.98 7.8 7.66 7.48 ...
##   .. ..$ dist   : num [1:155] 0.00136 0.01222 0.10303 0.19009 0.27709 ...
##   .. ..$ om     : num [1:155] 13.6 14 13 8 8.7 7.8 9.2 9.5 10.6 6.3 ...
##   .. ..$ ffreq  : Factor w/ 3 levels "1","2","3": 1 1 1 1 1 1 1 1 1 1 ...
##   .. ..$ soil   : Factor w/ 3 levels "1","2","3": 1 1 1 2 2 2 2 1 1 2 ...
##   .. ..$ lime   : Factor w/ 2 levels "0","1": 2 2 2 1 1 1 1 1 1 1 ...
##   .. ..$ landuse: Factor w/ 15 levels "Aa","Ab","Ag",..: 4 4 4 11 4 11 4 2 2 15 ...
##   .. ..$ dist.m : num [1:155] 50 30 150 270 380 470 240 120 240 420 ...
##   ..@ coords.nrs : int [1:2] 1 2
##   ..@ coords     : num [1:155, 1:2] 181072 181025 181165 181298 181307 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:155] "1" "2" "3" "4" ...
##   .. .. ..$ : chr [1:2] "x" "y"
##   ..@ bbox       : num [1:2, 1:2] 178605 329714 181390 333611
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:2] "x" "y"
##   .. .. ..$ : chr [1:2] "min" "max"
##   ..@ proj4string:Formal class 'CRS' [package "sp"] with 1 slot
##   .. .. ..@ projargs: chr NA
```

```
head(meuse)
```

```
##        coordinates cadmium copper lead zinc  elev       dist   om ffreq
## 1 (181072, 333611)    11.7     85  299 1022 7.909 0.00135803 13.6     1
## 2 (181025, 333558)     8.6     81  277 1141 6.983 0.01222430 14.0     1
## 3 (181165, 333537)     6.5     68  199  640 7.800 0.10302900 13.0     1
## 4 (181298, 333484)     2.6     81  116  257 7.655 0.19009400  8.0     1
## 5 (181307, 333330)     2.8     48  117  269 7.480 0.27709000  8.7     1
## 6 (181390, 333260)     3.0     61  137  281 7.791 0.36406700  7.8     1
##   soil lime landuse dist.m
## 1    1    1      Ah     50
## 2    1    1      Ah     30
## 3    1    1      Ah    150
## 4    2    0      Ga    270
## 5    2    0      Ah    380
## 6    2    0      Ga    470
```

The function `coordinates()` can retrieve spatial coordinates from a SpatialPointsDataFrame:

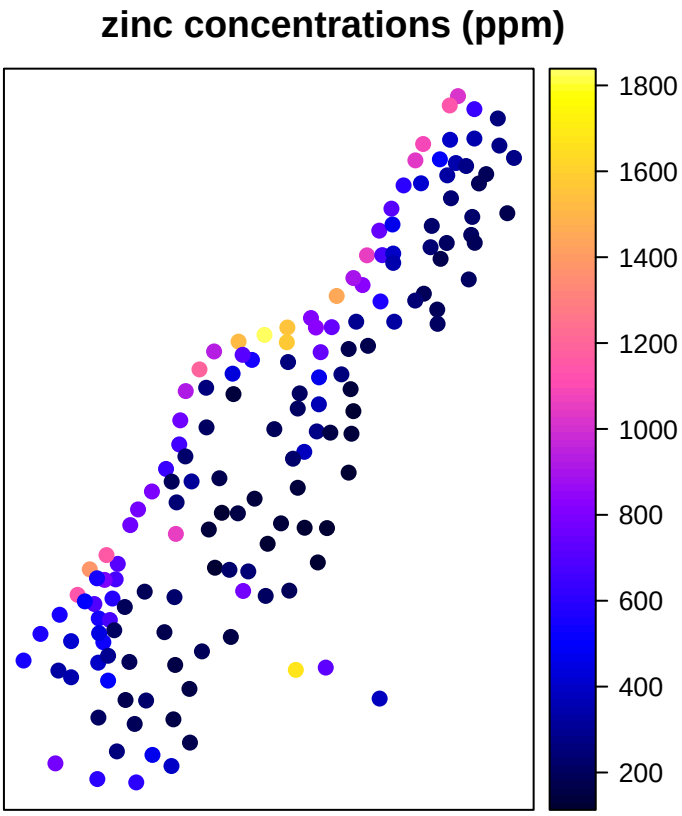
```
coordinates(meuse)[5:15,]
```

```
##         x      y
## 5  181307 333330
## 6  181390 333260
## 7  181165 333370
## 8  181027 333363
## 9  181060 333231
## 10 181232 333168
## 11 181191 333115
## 12 181032 333031
```

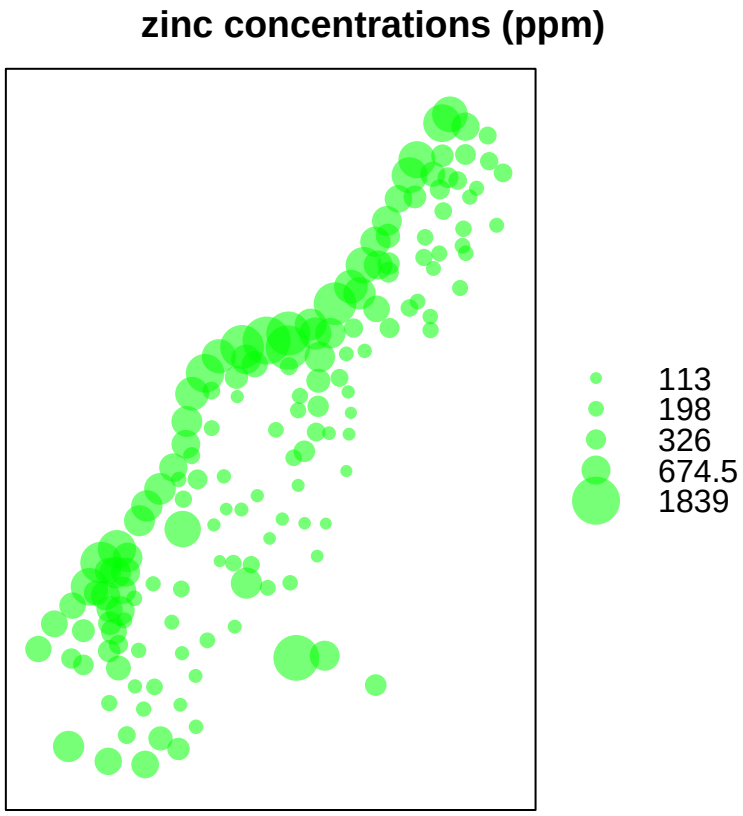
```
## 13 180874 333339
## 14 180969 333252
## 15 181011 333161
```

Now, we can plot the data using `spplot` and `bubble` as illustrated below (note: the x- and y-axis are the spatial coordinates)

```
spplot(meuse, "zinc",  colorkey = TRUE, main = "zinc concentrations (ppm)")
```




```
bubble(meuse, "zinc", col=c("#00ff0088"), main = "zinc concentrations (ppm)")
```



Concentrations seem to follow (inversely) from distances to the river

```
data(meuse.grid) # we read-in a separately provided sample grid (map)
coordinates(meuse.grid) = ~x+y
gridded(meuse.grid) = TRUE
class(meuse.grid)
```

```
## [1] "SpatialPixelsDataFrame"
## attr(,"package")
## [1] "sp"
```

```
# plot distances to river
image(meuse.grid["dist"], main="distance to river (red = 0)")
```

Semivariogram

- Calculate the sample variogram. This is done with the `variogram()` function.
- Fit a model to the sample variogram using `fit.variogram()` function

The `variogram()` function takes two arguments: the first denotes how one or more variables interact spatially, and the second is an SPDF where those variables reside.

- In our example, we assume that there is a constant mean (no trend) for the variable `log(zinc)`

```
lzn.vgm <- variogram(log(zinc)~1, meuse) # calculates sample variogram values
lzn.vgm # 15 groups | # obs per group | repres distances | gamma | ...isotropy assumed
```

```
##     np       dist     gamma dir.hor dir.ver   id
## 1   57   79.29244 0.1234479       0       0 var1
## 2  299  163.97367 0.2162185       0       0 var1
## 3  419  267.36483 0.3027859       0       0 var1
## 4  457  372.73542 0.4121448       0       0 var1
## 5  547  478.47670 0.4634128       0       0 var1
## 6  533  585.34058 0.5646933       0       0 var1
## 7  574  693.14526 0.5689683       0       0 var1
## 8  564  796.18365 0.6186769       0       0 var1
## 9  589  903.14650 0.6471479       0       0 var1
## 10 543 1011.29177 0.6915705       0       0 var1
## 11 500 1117.86235 0.7033984       0       0 var1
```

```
## 12 477 1221.32810 0.6038770       0       0 var1
## 13 452 1329.16407 0.6517158       0       0 var1
## 14 457 1437.25620 0.5665318       0       0 var1
## 15 415 1543.20248 0.5748227       0       0 var1
```

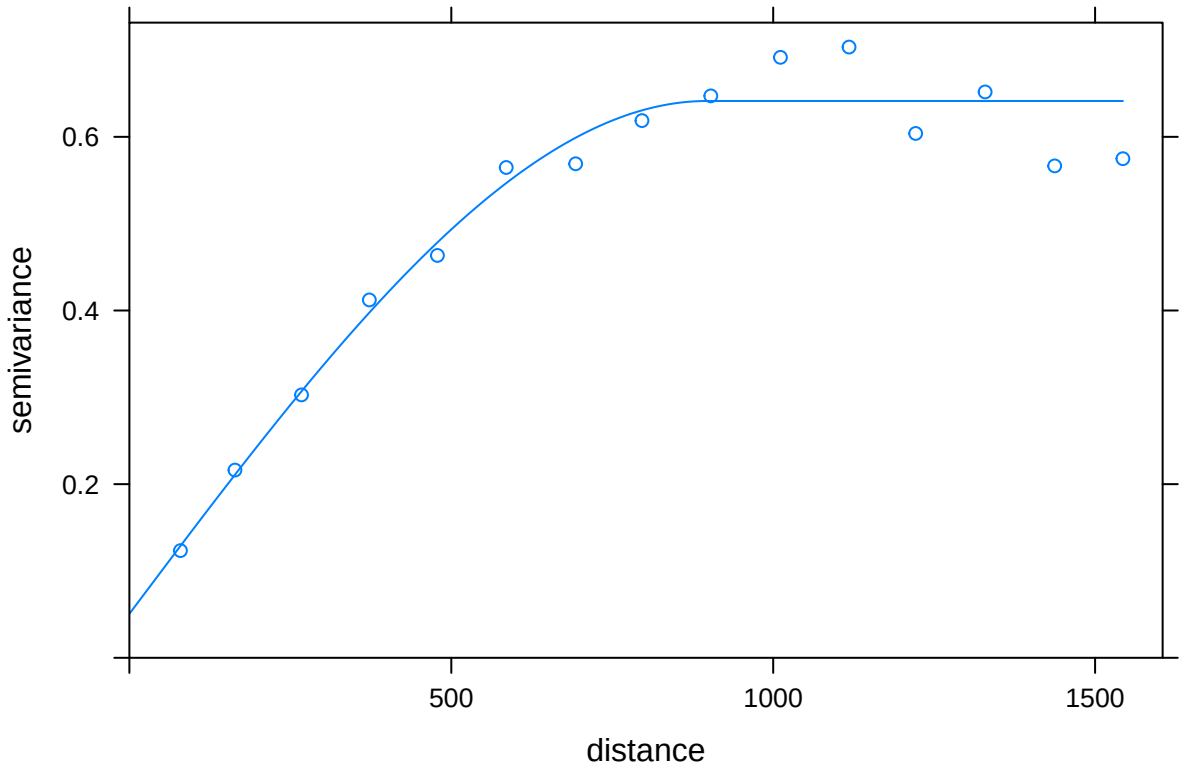
For the `fit.variogram()` function, a sample variogram is the first argument. The second is the model, with parameters, to be fit to the sample variogram. For a list of all possible variograms that can be used, call `?vgm`, and to see graphical properties/characteristics of these models, call `?show.vgms`.

- `vgm` arguments (quick summary)
  - `psill=NA` - starting value for the sill argument is not provided (generated internally for the estimation algorithm)
  - `Sph` - model type, e.g. "Exp", "Sph", "Gau"
  - `range=900` range in meters
  - `nugget=1` nonzero nugget added (included in variogram calculation)

```
lzn.fit <- fit.variogram(lzn.vgm, model=vgm(psill= NA, "Sph", range=900, nugget=1)) # fit model
lzn.fit
```

```
##   model      psill    range
## 1   Nug 0.05066642   0.0000
## 2   Sph 0.59061172 897.0499
```

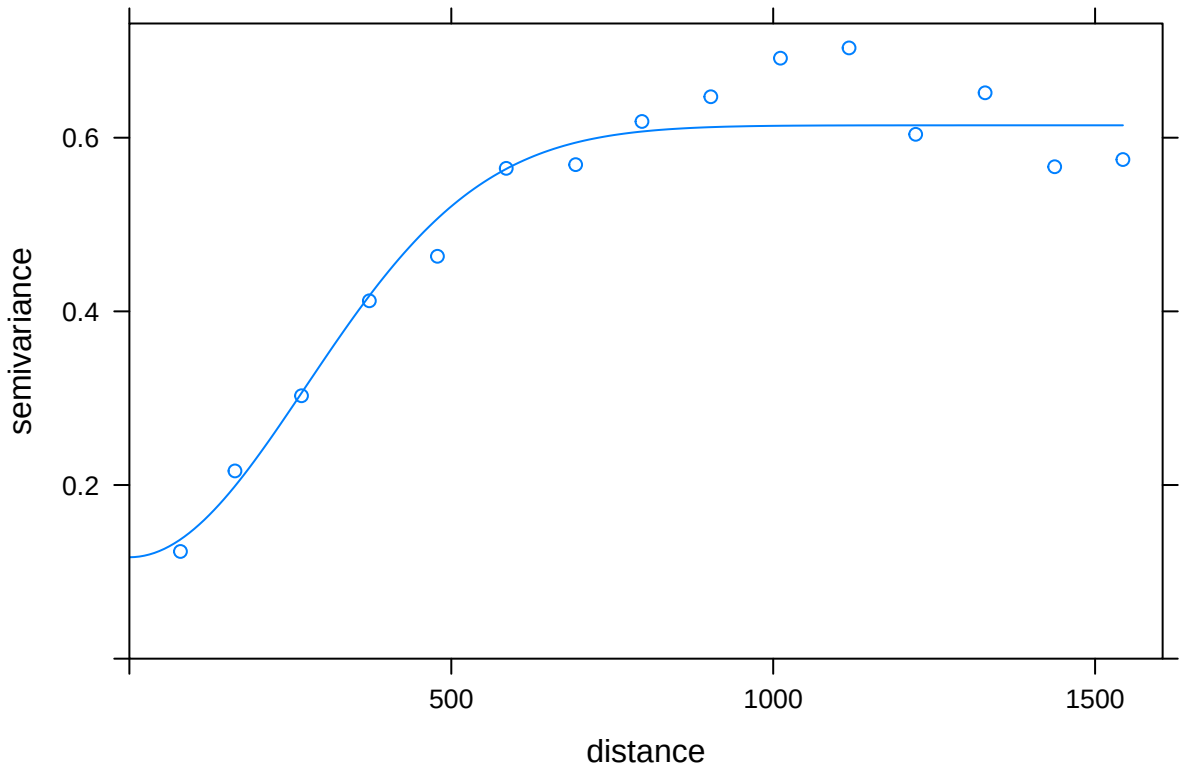
```
plot(lzn.vgm, lzn.fit) # plot the fitted variogram and the observed variogram on the same graph
```



Fitted semivariograms depend on assumptions used:

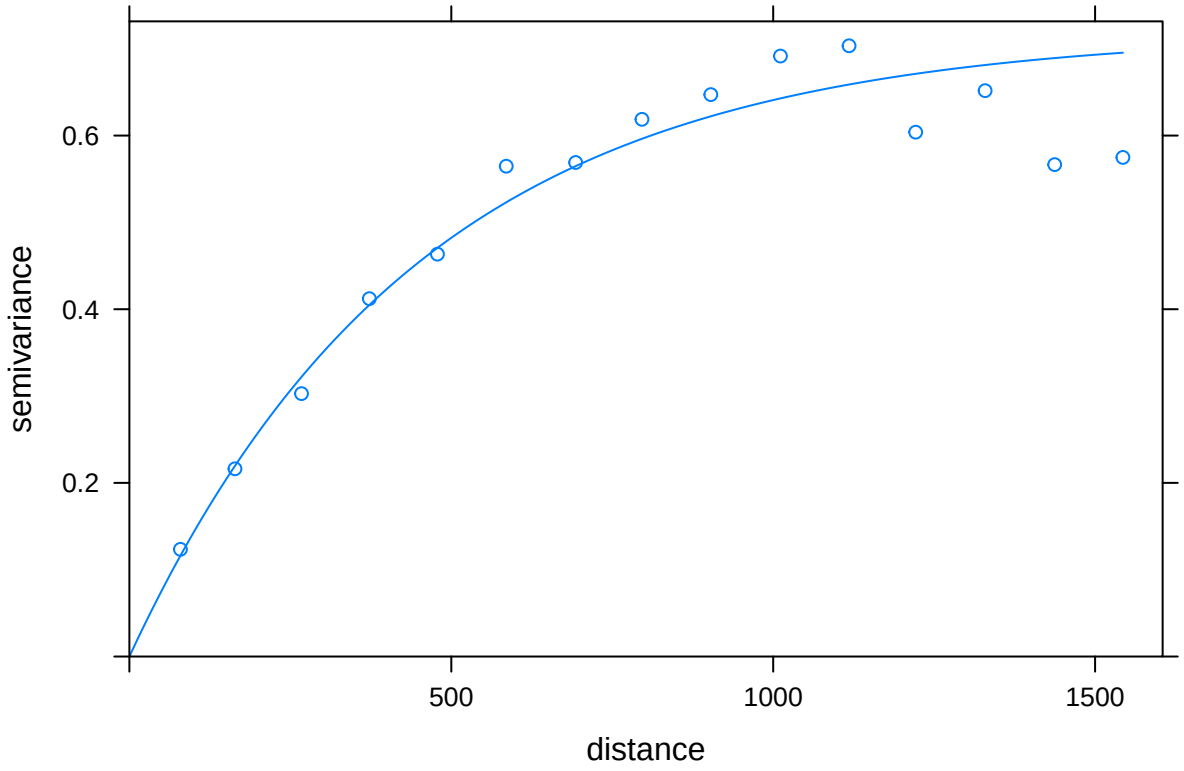
Gaussian semivariance wrt distance:

```
lzn.gau <- fit.variogram(lzn.vgm, model=vgm(psill= NA, "Gau", range=900, nugget=1)) # fit model
#lzn.gau
plot(lzn.vgm, lzn.gau)
```



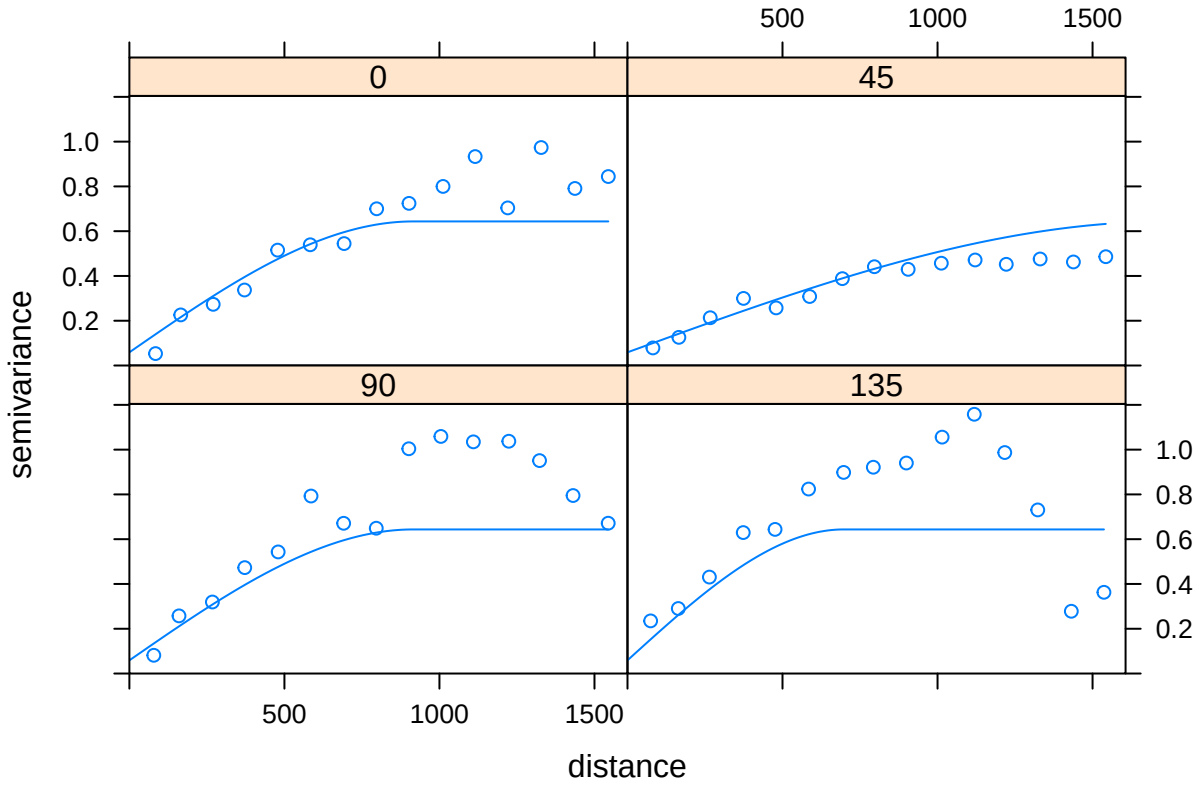
Exponential semivariance wrt distance:

```
lzn.exp <- fit.variogram(lzn.vgm, model=vgm(psill= NA, "Exp", range=900, nugget=1)) # fit model
#lzn.exp
plot(lzn.vgm, lzn.exp)
```



Directional (anisotropic) semivariance wrt distance:

```
vgm3.dir <- variogram(log(zinc)~1, meuse, alpha = c(0, 45, 90, 135))
vgm3.fit.d <- fit.variogram(vgm3.dir, model = vgm(.59, "Sph", 1200, .05, anis = c(45, .4)))
#vgm3.fit.d
plot(vgm3.dir, vgm3.fit.d, as.table = T)
```



For spatio-temporal semivariogram, see e.g. tutorial here