

Spatial analysis with the R Project for Statistical Computing

D G Rossiter

International Institute for Geo-information Science & Earth Observation (ITC)

October 16, 2008

Copyright © 2008 ITC.

All rights reserved. Reproduction and dissemination of the work as a whole (not parts) freely permitted if this original copyright notice is included. Sale or placement on a web site where payment must be made to access this document is strictly prohibited. To adapt or translate please contact the author (<http://www.itc.nl/personal/rossiter>).



Topics for this lecture

1. Spatial analysis in R
2. The `sp` package: spatial classes
3. Interfaces with other spatial analysis
4. The `gstat` package: geostatistics

Spatial analysis in R

1. Spatial data
2. R approaches to spatial data
3. Analysis sequence: R in combination with other programs

Spatial data

- ... are data with **coördinates**, i.e. known locations
- These are **absolute locations** in one, two or three dimensions
 - * in some defined **coördinate system**
 - * possibly with some defined **projection** and **datum**
- Points are implicitly related by **distance** and **direction** of **separation**
- Polygons are implicitly related by adjacency, containment
- Such data requires **different analysis** than non-spatial data
 - * e.g. can't assume independence of observations
- And they need special **data structures** which recognize the special status of coördinates

R approaches

- No native S classes for these
- S is **extensible** with new classes (S3 or S4 systems), methods and packages
- So, several **add-in packages** have been developed
- Add-in packages which define **spatial classes and methods**:
 - * sp (Bivand, Pebesma): generic S4 spatial classes
- Add-in packages for **spatial analysis**; statisticians can implement their own ideas; including:
 - * spatial (Ripley, based on 1981 book)
 - * geoR, geoRglm (Ribeiro & Diggle, basis for 2007 book)
 - * gstat (Pebesma)
 - * spatstat (Baddeley & Turner): point patterns
 - * circular: directional statistics
 - * RandomFields (Schlather)

Interface to GIS

- Add-in packages:
 - * `rgdal`: interface to Geospatial Data Abstraction Library (GDAL)
 - * `maptools`: interface to external spatial data structures e.g. shapefiles

(See later topic)

CRAN Task View: Analysis of Spatial Data

<http://cran.r-project.org/web/views/Spatial.html> explains what packages are available for:

- Classes for spatial data
- Handling spatial data
- Reading and writing spatial data
- Point pattern analysis
- Geostatistics
- Disease mapping and areal data analysis
- Spatial regression
- Ecological analysis

Analysis sequence

Use the most **suitable** tool for each phase of the analysis:

1. Prepare spatial data in GIS or image processing program
 - Can prepare matrices, dataframes (e.g. coördinates and data values), even topology directly in R but usually it's easier to use a specialized program.
2. Import to R
3. Perform analysis in R
4. Display spatial results in R using R graphics
5. Export results to GIS or mapping program
6. Use for further GIS analysis or include in map layout

The sp package

- Motivation: “The advantage of having multiple R packages for spatial statistics seemed to be hindered by a lack of a **uniform interface** for handling spatial data.”
- This package provides **classes** and **methods** for dealing with **spatial data** in S
- By itself it does not provide geostatistical analysis
- Various analytical packages can all use these classes
- Data does not have to be restructured for different sorts of analysis

sp Spatial data structures

- Spatial **data structures** (S4 classes):
 - * **points**
 - * **lines**
 - * **polygons**
 - * **grids** (rasters)
- These may all have **attributes** (dataframes)
- S4 class names like SpatialPointsDataFrame
- Generic methods with appropriate behaviour for each class

sp Methods for handling spatial data

Some standard methods:

- `bbox`: bounding box
- `dimensions`: number of spatial dimensions
- `coordinates`: set or extract coordinates
- `overlay`: combine two spatial layers of different type
 - * e.g. retrieve the polygon or grid values on a set of points
 - * e.g. retrieve the points or their attributes within (sets of) polygons
- `spsample`: point sampling schemes within a geographic context (grids or polygons)
- `spplot`: visualize spatial data

Converting data to sp classes

Simple rule: data is spatial if it has **coordinates**.

The most common way to assign coordinates is to use the `coordinates` method as the LHS of an expression.

First we see how spatial data looks when it is not treated as spatial data:

```
> library(gstat) # meuse is an example dataset in this package
> data(meuse); str(meuse); spsample(meuse, 5, "random")
```

```
'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 ...
```

```
Error in function (classes, fdef, mtable) :
  unable to find an inherited method for function "spsample", for signature "data.frame"
```

Fields `x` and `y` are coordinates but they have no special status in the dataframe. The method `spsample` expects a spatial object but does not find it.

Converting to spatial data

We explicitly identify the fields that are coordinates:

```
> coordinates(meuse) <- ~ x + y; 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 ...
..@ coords.nrs : int [1:2] 1 2
..@ coords     : num [1:155, 1:2] 181072 181025 181165 181298 181307 ...
.. ..- attr(*, "dimnames")=List of 2
.. .. ..$ : NULL
.. .. ..$ : 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 slots
.. .. ..@ projargs: chr NA
```

The S4 class has **slots** (@) for different kinds of information (coordinates, dimensions, data, bounding box, projection)

- Now the `spsample` method works
- It returns an object of class `SpatialPoints` (i.e. just the locations, no attributes)

```
> spsample(meuse, 5, "random")
```

```
SpatialPoints:
```

```
          x      y  
[1,] 179246 330766  
[2,] 180329 330055  
[3,] 181280 329798  
[4,] 180391 330627  
[5,] 178725 330202
```

Interfaces with other spatial analysis

Most spatial data is prepared outside of R; results of R analyses are often presented outside of R. How is information exchanged?

- `maptools`
- `rgdal`
- Projections

The maptools package

These are tools for reading and handling spatial objects, developed as part of the sp initiative. For the most part superseded by rgdal.

- `read.shape`, `readShapePoints`, `readShapeLines`, `readShapePoly`: read ESRI **shapefiles**
- `writeShapePoints`, `writeShapeLines`, `writeShapePoly`: write ESRI shapefiles
- `read.AsciiGrid`, `write.AsciiGrid`: **ESRI Ascii GRID** interchange format

These are restricted to just the named formats; a more generic solution was needed, which is provided by ...

The rgdal package

This package provides **bindings** for the **Geospatial Data Abstraction Library** (GDAL)¹, which is an open-source **translator** library for geospatial data formats.

rgdal uses the sp classes.

- readGDAL; writeGDAL: Read/write between GDAL grid maps and Spatial objects
- readOGR, writeOGR: Read/write spatial vector data using OGR (including **KML** for Google Earth)
 - * OGR: C++ open source library providing read (and sometimes write) access to a variety of **vector file formats** including ESRI Shapefiles, S-57, SDTS, PostGIS, Oracle Spatial, and Mapinfo mid/mif and TAB formats

¹<http://www.gdal.org/>

Projections

Until a projection is defined, the coördinates are just numbers; they are not related to the Earth's surface. This can be useful for “spatial” analysis of non-Earth objects, e.g. an image from a microscope.

For true geographic analysis, the object must be related to the Earth's figure.

Then true **distances** and **azimuths** can be computed.

The CRS method

sp uses the **CRS** (Coördinate Reference System) method of the **rgdal** package to interface with the **proj.4** cartographic projection library from the USGS².

For example, to specify the datum, elipsoid, projection, and coördinate system of the Dutch Rijksdriehoek (RDH) system:

```
> proj4string(meuse) <- CRS("+proj=stere
+   +lat_0=52.15616055555555 +lon_0=5.387638888888889
+   +k=0.999908 +x_0=155000 +y_0=463000
+   +ellps=bessel +units=m +no_defs
+   +towgs84=565.2369,50.0087,465.658,
+   -0.406857330322398,0.350732676542563,-1.8703473836068,
+   4.0812")
```

Most systems are included in the European Petroleum Survey Group (**EPSG**) database³, in which case just the system's number in that database is enough to specify it:

```
> proj4string(meuse) <- CRS("+init=epsg:28992")
```

²<http://trac.osgeo.org/proj/>

³<http://www.epsg.org/>

The gstat package

- R implementation of the stand-alone **gstat** package for geostatistics
- Author and maintainer Edzer Pebesma
 - * mostly developed at Physical Geography, University of Utrecht (NL)
 - * since Oct 2007 at Institute for Geoinformatics, University of Münster (D)
- 1992 – 2008 and still going strong
- Purpose: “**modelling**, **prediction** and **simulation** of geostatistical data in one, two or three dimensions”
- Highly-developed suite of tools with rich analytical possibilities; uses the **sp** spatial data structures

Note: there are other R packages with largely overlapping aims but with different philosophies and interfaces (notably, `geoR` and `spatial`).

Modelling

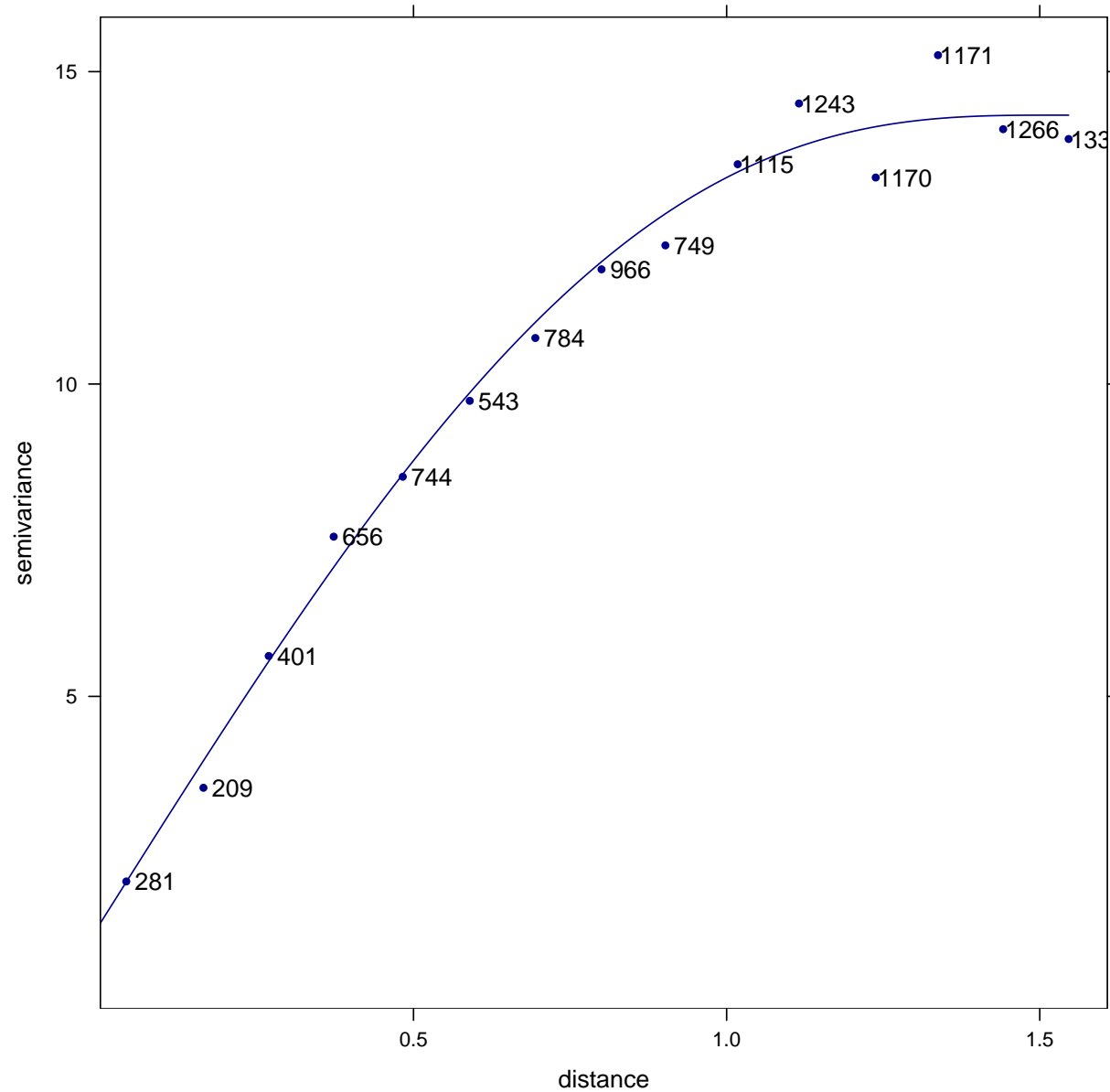
- variogram: Compute **experimental variograms** (also directional, residual)
 - * User-specifiable cutoff, bins, anisotropy angles and tolerances
 - * Can use Matheron or robust estimators
 - * Optional argument to produce a **variogram cloud** (all point-pairs)
 - * Optional argument to produce a **directional variogram surface** (“map”)
- vgm: specify a **theoretical variogram model** for an empirical variogram
 - * Many authorized models
 - * Can specify models with **multiple structures**
- `fit.variogram`: least-squares adjustment of a variogram model to the empirical model
 - * User-selectable fitting criteria
 - * Can also use restricted maximum likelihood `fit.variogram.reml`
- `fit.lmc`: fit a linear model of **coregionalization** (for cokriging)

The gstat method

More complicated procedures must be specified with the gstat method.

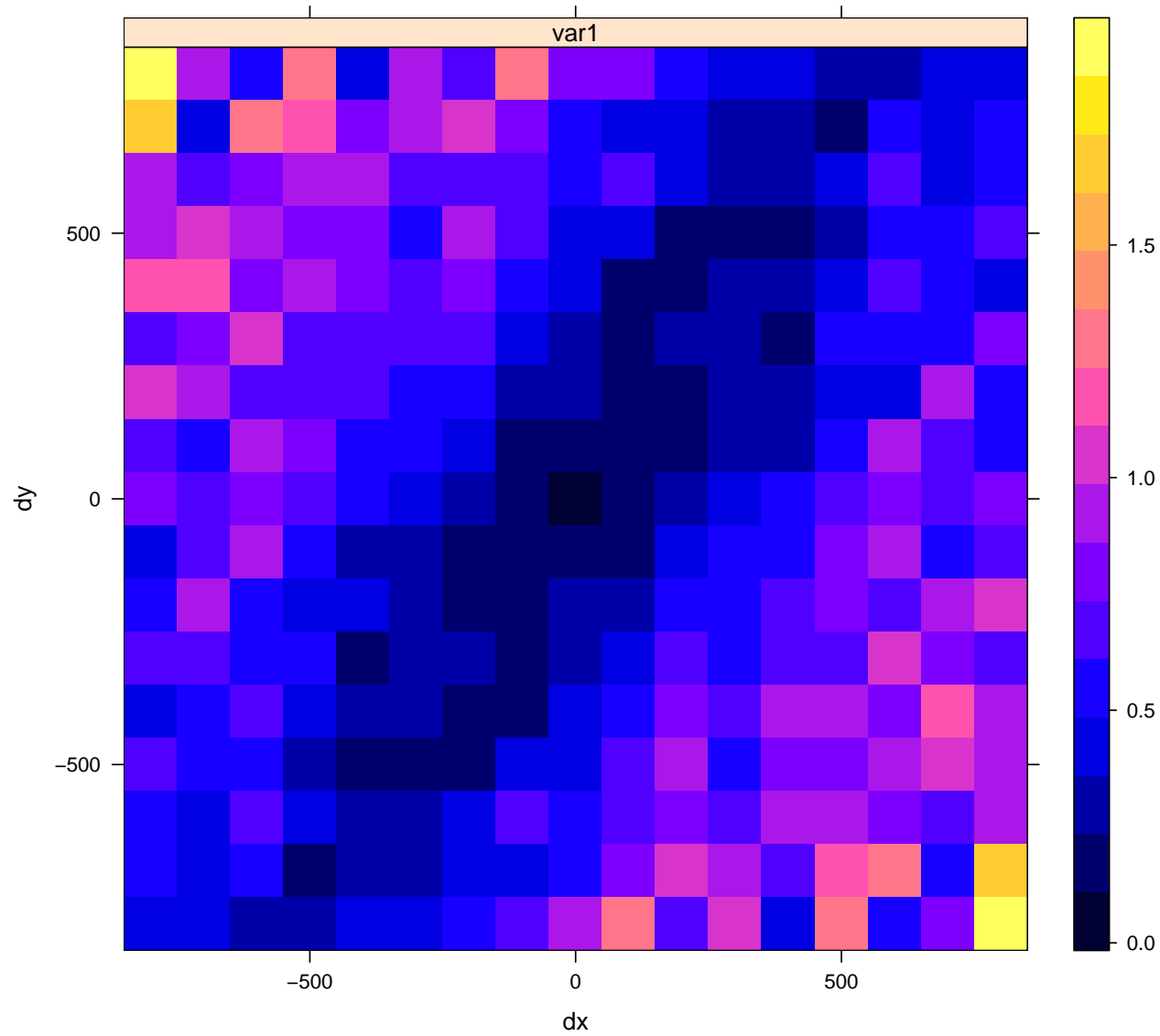
Example: [cokriging](#)

Experimental variogram and fitted model

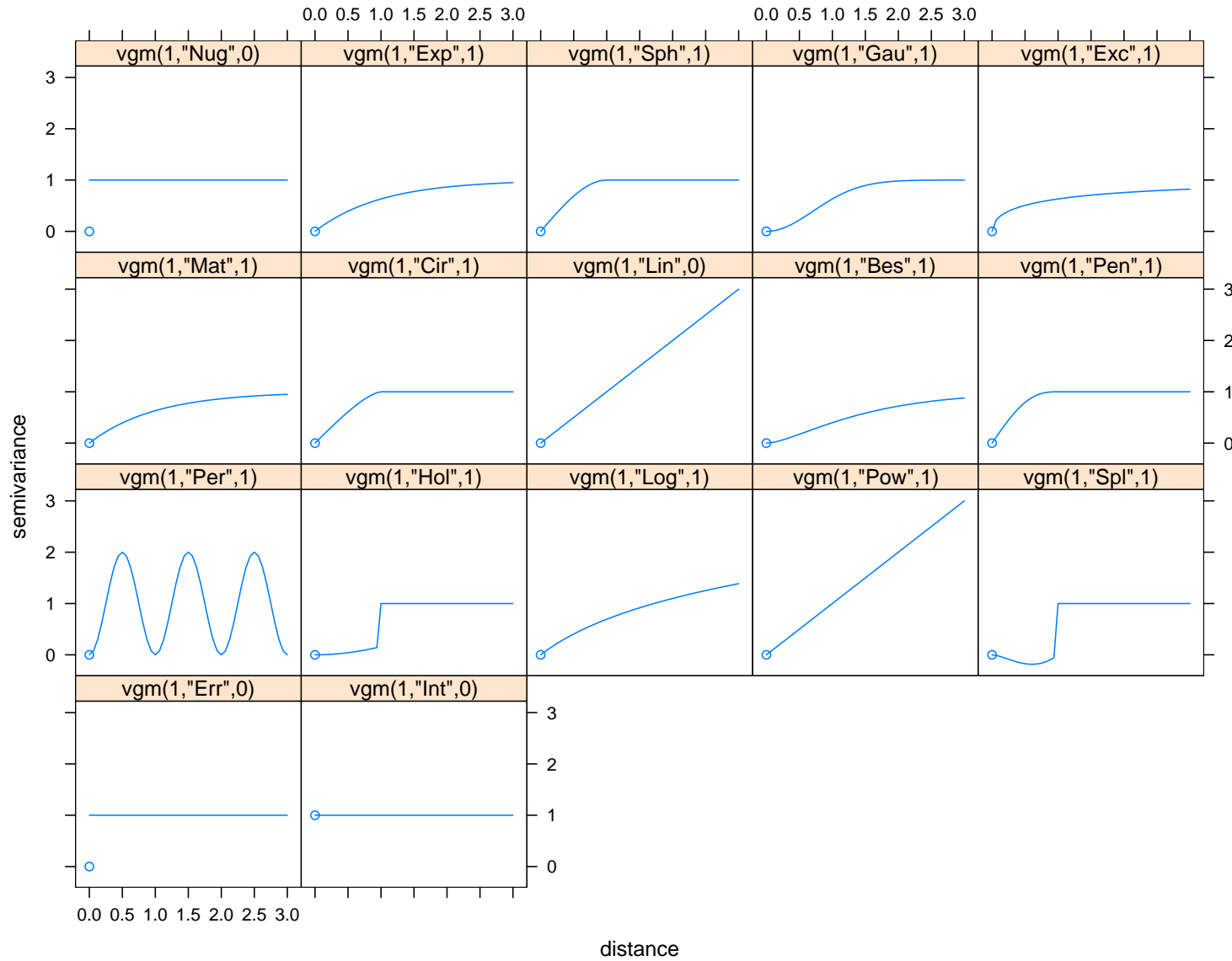


Variogram map

Variogram map, Meuse River, log(zinc)



Authorized models

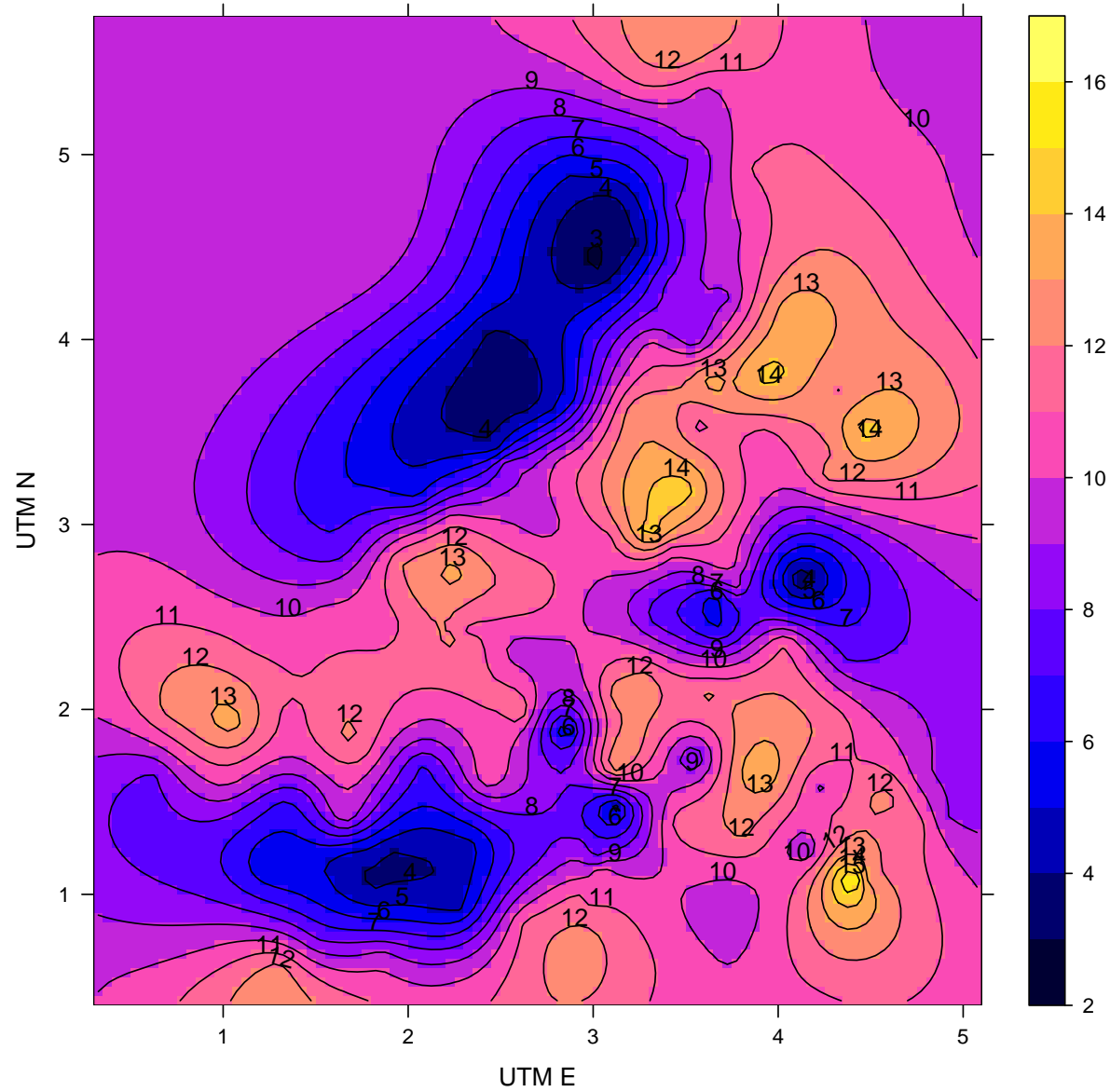


Prediction

- `krige`: Kriging prediction
 - * Simple (known spatial mean)
 - * Ordinary (spatial mean must be estimated also)
 - * Universal, KED (geographic trend or feature-space model)
 - * global or in local neighbourhood
 - * at points or integrated over blocks
- `idw`: Inverse distance prediction (user-specified power)
- `predict.gstat`: trend surfaces, cokriging

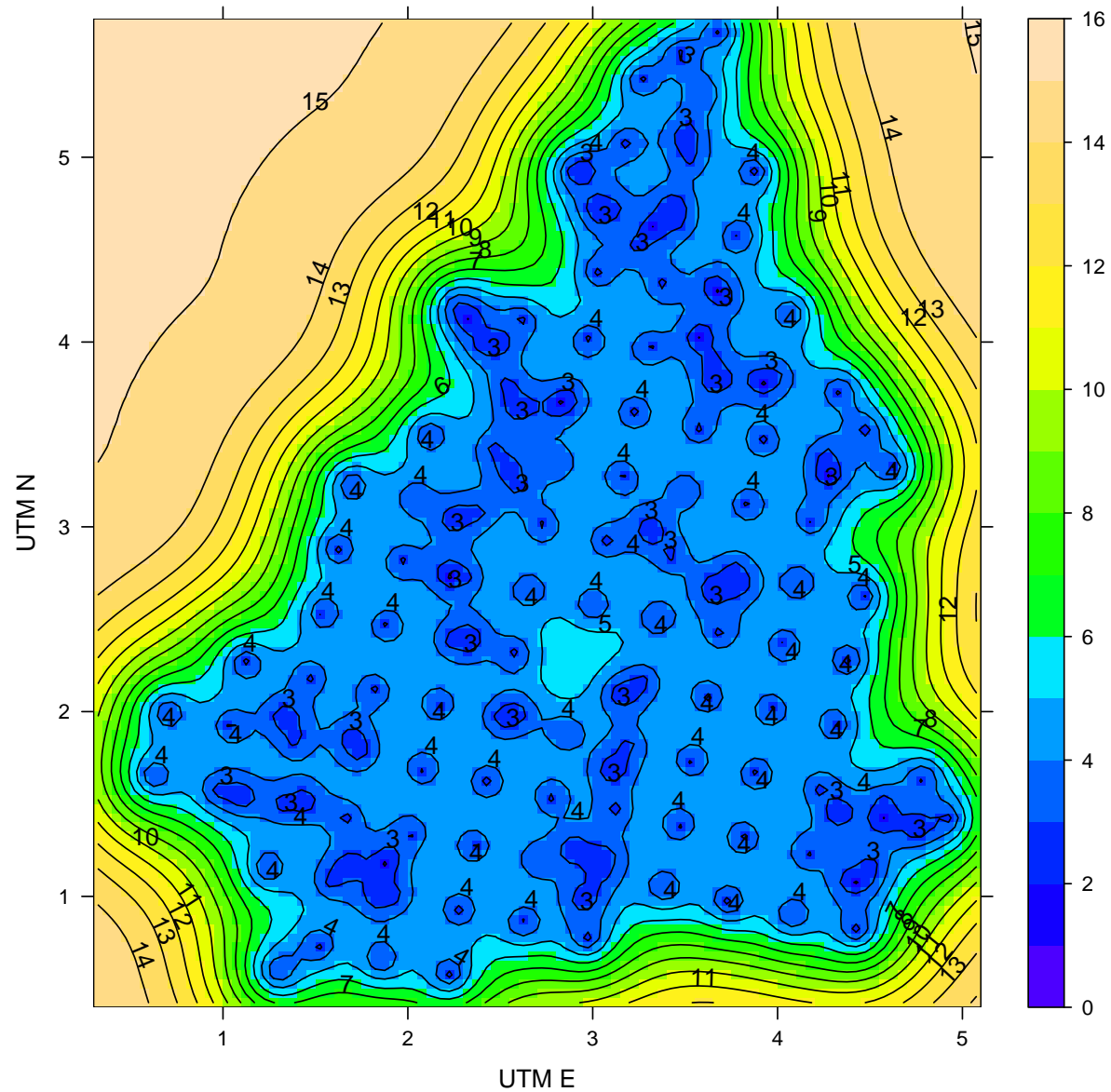
Kriging prediction

Predicted values, Co (ppm)



Kriging prediction variance

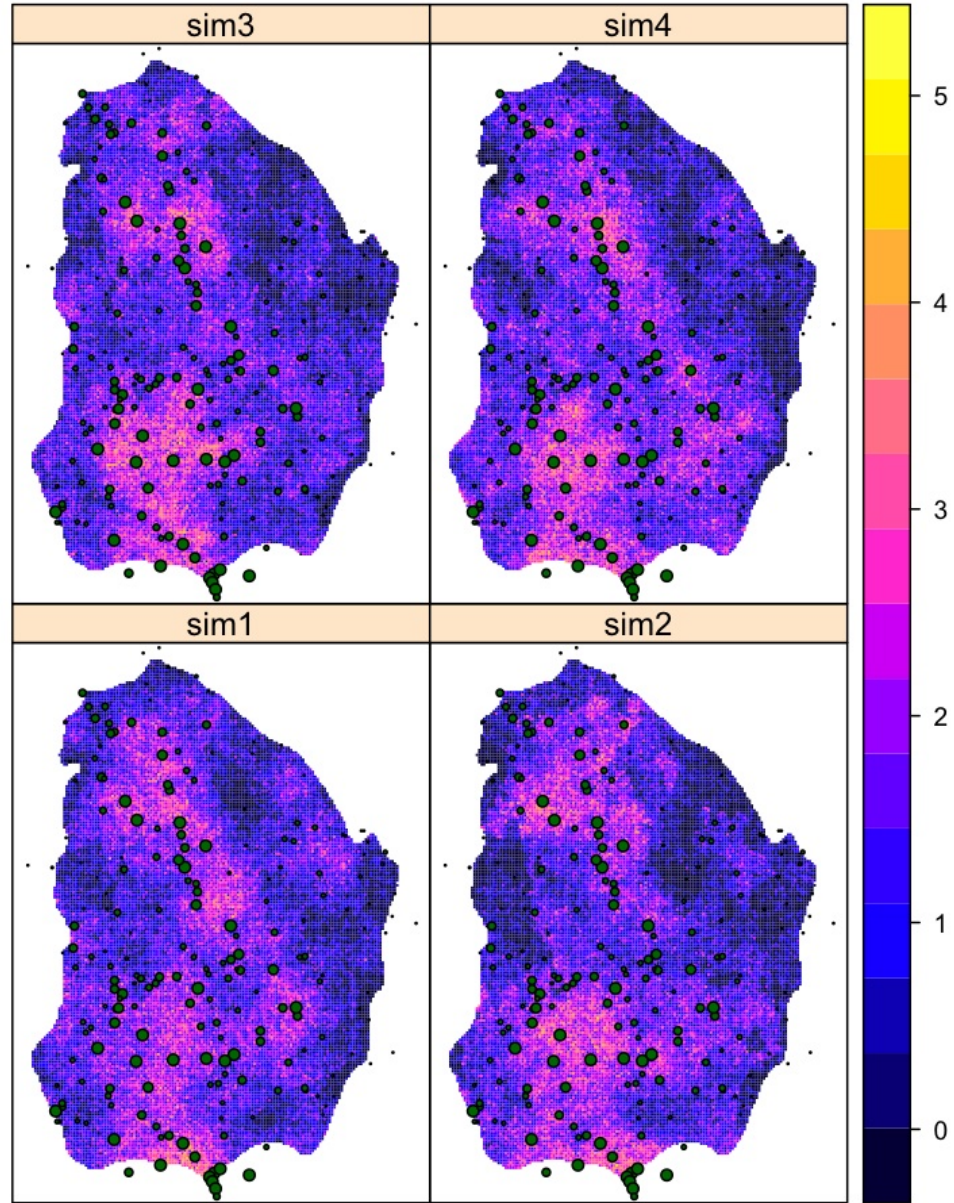
Kriging variance, Co (ppm²)



Simulation

- `krige`: optional arguments to specify **conditional** or **unconditional** simulation with a known variogram model
- Shows possible states of nature, given the known data and the fitted model
- More realistic spatial picture than the (smooth) kriged result
- Often used in distributed models

Conditional simulations, KED on land use



regolith depth (m)

Validation

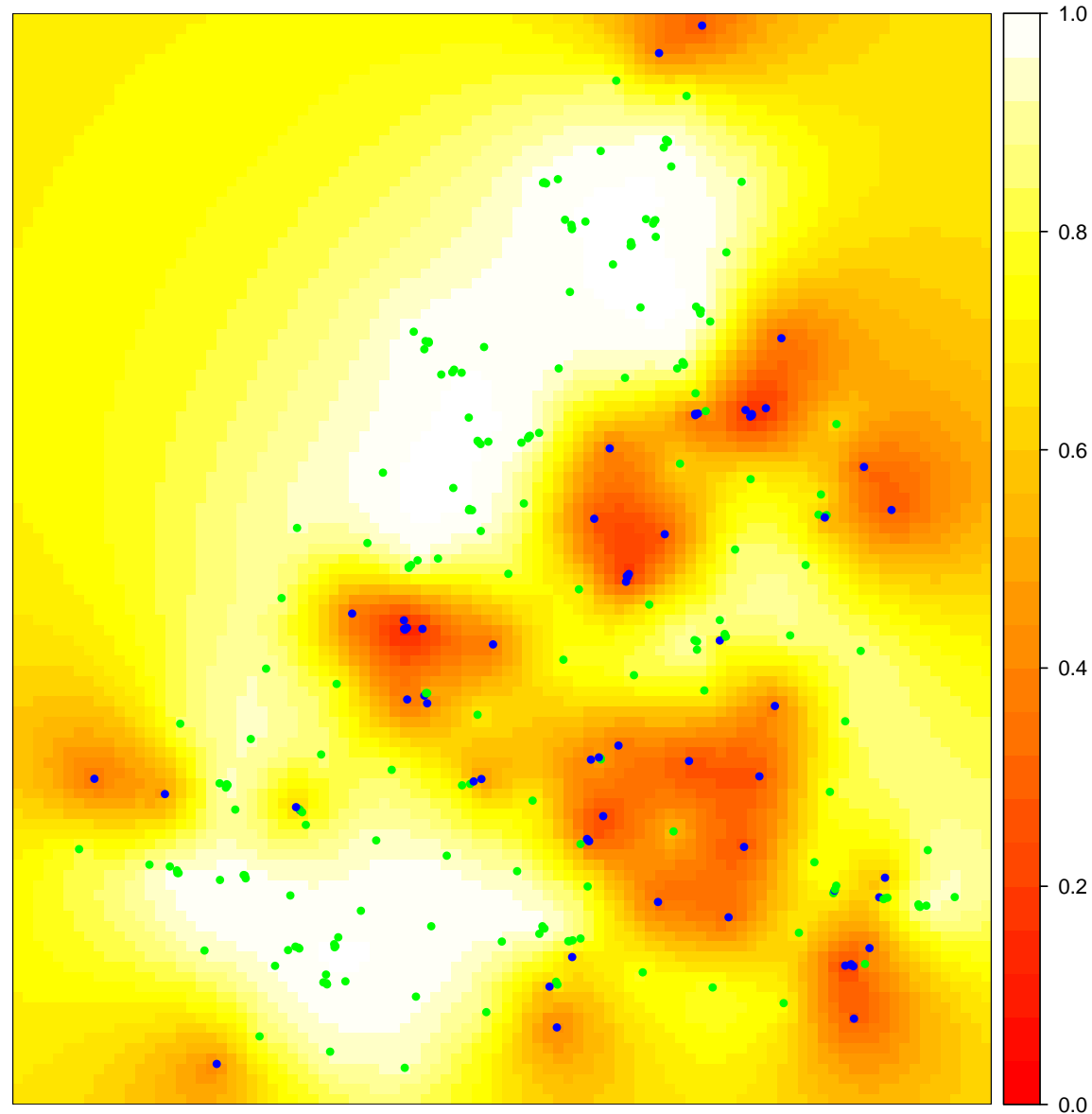
- `krige.cv`, `gstat.cv`: n-fold **cross-validation**, including leave-one-one cross-validation (LOOCV).

Non-parametric methods

- All methods can be applied to **indicators** (0/1, T/F, Y/N)
- It is possible to treat the indicators as **coregionalized** variables

Indicator Kriging prediction

Probability Co < 12 ppm, Jura soils



Sample points: <12 ppm green, >= 12 ppm blue

A typical analysis

```
> # load an example dataset into the workspace
> data(meuse)
> # convert it to a spatial object
> coordinates(meuse) <- ~ x + y
> # compute the default experimental variogram
> v <- variogram(log(lead) ~ 1, meuse)
> # plot the variogram and estimate the model by eye
> plot(v)
> vm <- vgm(0.5, "Sph", 1000, 0.1)
> plot(v, model=vm)
> # fit the model with the default automatic fit
> (vmf <- fit.variogram(v, vm))
> # load a prediction grid
> data(meuse.grid)
> # convert it to a spatial object
> coordinates(meuse.grid) <- ~ 1
> # predict on the grid by Ordinary Kriging
> ko <- krige(log(lead) ~ 1, meuse, newdata=meuse.grid, model=vmf)
> summary(ko)
> # plot the map
> spplot(ko, zcol="var1.pred")
> # leave-one-out cross-validation
> kcv <- krige.cv(log(lead) ~ 1, meuse, model=vmf)
> summary(kcv)
```

Output from modelling and prediction

```
model  psill  range
1  Nug 0.051563  0.00
2  Sph 0.515307 965.15
```

[using ordinary kriging]

Object of class SpatialPointsDataFrame

Coordinates:

```
      min      max
x 178460 181540
y 329620 333740
```

Number of points: 3103

var1.pred		var1.var	
Min.	:3.67	Min.	:0.083
1st Qu.:	4.23	1st Qu.:	0.125
Median	:4.56	Median	:0.145
Mean	:4.65	Mean	:0.164
3rd Qu.:	5.08	3rd Qu.:	0.185
Max.	:6.30	Max.	:0.422

Output from cross-validation

Object of class SpatialPointsDataFrame

Coordinates:

```
      min      max  
x 178605 181390  
y 329714 333611
```

Number of points: 155

var1.pred	var1.var	observed	residual	zscore
Min. :3.73	Min. :0.107	Min. :3.61	Min. :-1.036486	Min. :-2.555439
1st Qu.:4.37	1st Qu.:0.140	1st Qu.:4.28	1st Qu.: -0.213458	1st Qu.: -0.527381
Median :4.82	Median :0.157	Median :4.81	Median :-0.011006	Median :-0.027213
Mean :4.81	Mean :0.166	Mean :4.81	Mean :-0.000381	Mean :-0.000285
3rd Qu.:5.20	3rd Qu.:0.178	3rd Qu.:5.33	3rd Qu.: 0.186346	3rd Qu.: 0.468856
Max. :6.10	Max. :0.467	Max. :6.48	Max. : 1.619113	Max. : 4.028651

Conclusion

- A **disadvantage** of working in R is the lack of interactive graphical analysis (e.g. in ArcGIS Geostatistical Analyst)
- The main **advantage** of doing spatial analysis in R is that the full power of the R environment (data manipulation, non-spatial modelling, user-defined functions, graphics ...) can be brought to bear on spatial analyses
- The advantages of R (**open-source**, **open environment**, packages contributed and vetted by statisticians) apply also to spatial analysis