6

Heavy metal concentrations (NGS)

6.1 Introduction

Now that you have become familiar with basic geostatistical operations in gstat and SAGA, we can proceed with running a mapping exercises with a more complex data set, i.e. a case study that is much closer to real applications. In this exercise we will produce maps of heavy metal concentrations (**HMCs**) for a large area (almost an entire continent), by using an extensive point data set, and a large quantity of auxiliary maps.

Heavy metals occur naturally in rocks and soils, but increasingly higher quantities of them are being released into the environment by anthropogenic activities. Before any solution to the problem of soil heavy metal pollution can be suggested, a distinction needs to be made between natural anomalies and those resulting from human activities. Auxiliary maps such as the ones used in this exercise can be used to show that HMCs are due to industrial activities, toxic releases, or due to the background geology. Such investigations permit one to do an in-depth analysis of the processes that cause the distribution of HMCs, so that also appropriate remediation policies can be selected.

We use the US National Geochemical Survey database (**NGS**), which contains 74,408 samples of 53(+286) attributes sampled over the period from (1979[1]) 2003 to 2008. The original goal of the NGS project was to analyze at least one stream-sediment sample in every 289 km^2 area by a single set of analytical methods across the entire USA. This is one of the most complete and largest geochemical databases in the World. Nevertheless, the most recent version of NGS (v.5) still contains some gaps[2], mostly in western states, although the overall coverage is already >80% of the country. The original data is provided as point maps or as county-level averages. The database is explained in detail in Grossman et al. (2008) and is publicly accessible[3]. Our objective is to map the areas of overall pollution for eight critical heavy metals: arsenic, cadmium, chromium, copper, mercury, nickel, lead and zinc. For practical reasons, we will focus only on the contiguous 48-state area.

The National Geochemical Survey Team has not yet analyzed these data using geostatistical techniques; so far, only maps of individual heavy metal parameters (interpolated using inverse distance interpolation) can be obtained from the NGS website. The maps presented in this exercise were created for demonstration purposes only. The map shown in Fig. 6.11 should be taken with a caution. Its accuracy needs to be assessed using objective criteria.

The advantages of NGS, as compared to e.g. similar European geochemical surveys[4] is that it is: (1) produced using consistent survey techniques; (2) it is spatially representative for all parts of USA; (3) it is extensive; and (4) it is of high quality. The biggest disadvantage of the NGS is that the samples from different media (water, soil and stream sediments) are not equally spread around the whole country — in fact in most cases the two sampling projects do not overlap spatially. Soil samples are only available for about 30% of the whole territory; stream/lake sediment samples have a slightly better coverage. It is unfortunate that different

[1] The NURE samples were collected around 1980.

[2] Mainly Indian reservations and military-controlled areas (see further Fig. 6.2).

[3] http://tin.er.usgs.gov/geochem/

[4] European Geological Surveys Geochemical database contains only 1588 georeferenced samples for 26 European countries.

media are not equally represented in different parts of the USA. Mixing of media in this analysis is certainly a serious problem, which we will ignore in this exercise. For more info about NGS data, please contact the USGS National Geochemical Survey Team.

This exercise is largely based on papers previously published in Geoderma (Rodriguez Lado et al., 2009), and in Computers and Geosciences journals (Romić et al., 2007). A more comprehensive introduction to geochemical mapping can be followed in Reimann et al. (2008). A more in-depth discussion about the statistical aspects of dealing with hot spots, skewed distributions and measurement errors can be followed in the work of Moyeed and Papritz (2002) and Papritz et al. (2005). Note also that this is an exercise with intensive computations using a large data set. It is not recommended to run this exercises using a PC without at least 2 GB of RAM and at least 1 GB of free space. Also make sure you regularly save your R script and the R workspace, using the `save.image` method so you do not lose any work.

6.2 Download and preliminary exploration of data

6.2.1 Point-sampled values of HMCs

Open the R script (`NGS.R`) and run line by line. The NGS shapefile can be directly obtained from:

```
> download.file("http://tin.er.usgs.gov/geochem/ngs.zip",
+       destfile=paste(getwd(),"ngs.zip",sep="/"))
> for(j in list(".shp", ".shx", ".dbf")){
>   fname <- zip.file.extract(file=paste("ngs", j, sep=""), zipname="ngs.zip")
>   file.copy(fname, paste("./ngs", j, sep=""), overwrite=TRUE)
> }
```

To get some info about this map, we can use the `ogrInfo` method:

```
> ogrInfo("ngs.shp", "ngs")
```

```
 Driver: ESRI Shapefile number of rows 74408
 Feature type: wkbPoint with 2 dimensions
 Number of fields: 53
         name type length typeName
 1      LABNO    4     10   String
 2   CATEGORY    4     11   String
 3    DATASET    4     19   String
 4   TYPEDESC    4     12   String
 5      COUNT    0      6  Integer
 6  ICP40_JOB    4      9   String
 7   AL_ICP40    2     14     Real
 8   CA_ICP40    2     14     Real
 ...
 52     SE_AA    2     12     Real
 53     HG_AA    2     14     Real
```

which shows the number of points in the map, and the content of the attribute table: `AL_ICP40` are the measurements of aluminum using the ICP40 method (Inductively Coupled Plasma-atomic emission spectrometry[5]) with values in wt%; `HG_AA` are values of mercury estimated using the AA method (Activation Analysis) in ppm and so on. The file is rather large, which will pose some limitations on the type of analysis we can do. We can now import the shapefile to R:

```
> ngs <- readOGR("ngs.shp", "ngs")
```

The samples that compose the NGS come from five main media (`CATEGORY`):

```
> summary(ngs$CATEGORY)
```

[5] http://tin.er.usgs.gov/geochem/doc/analysis.htm

```
        NURE NURE SIEVED       PLUTO
       45611        4754        1453
        RASS       STATE   SW-ALASKA
        1755       20126         709
```

The majority of points belongs to the National Uranium Resource Evaluation (NURE) and the STATE program, both were collected in the period (1979) 2003–2008. Following the NGS documentation[6], these points are in geographical coordinates with the NAD27 datum and Clarke 1866 ellipsoid, which we can set in R as:

```
> proj4string(ngs) <- CRS("+proj=longlat +ellps=clrk66 +datum=NAD27 +no_defs")
```

We are interested in mapping the following nine[7] variables:

```
> HMC.list <- c("AS_ICP40", "CD_ICP40", "CR_ICP40", "CU_ICP40",
+     "NI_ICP40", "ZN_ICP40", "AS_AA", "HG_AA", "PB_ICP40")
# short names:
> HM.list <- c("As","Cd","Cr","Cu","Ni","Zn","As2","Hg","Pb")
```

Let us first take a look at the properties of the data, i.e. what the range of values is, and how skewed the variables are. We can look at the first variable on the list:

```
> summary(ngs@data[,HMC.list[1]])
```

```
     Min. 1st Qu.  Median    Mean 3rd Qu.  Max.    NA's
   -40.00  -10.00  -10.00   -1.82   10.00  5870.00 3164.00
```

which shows that there are also negative values in the table — obviously artifacts. If we go back to the original documentation[8], we can notice that negative values mean "*measured, but below detection limit*". Masking the negative values with NA will not be correct because these are really zero or close to zero measurements. This would make a huge difference for further analysis because the proportion of such measurements can be significant. The solution is to automatically replace all negative (and values below the detection limit) values with half the detection limit (Reimann et al., 2008, p.23–24):

```
# "AS_ICP40"
> ngs@data[,HMC.list[1]] <- ifelse(ngs@data[,HMC.list[1]]<2,
+     abs(ngs@data[,HMC.list[1]])/2, ngs@data[,HMC.list[1]])
# "CD_ICP40"
> ngs@data[,HMC.list[2]] <- ifelse(ngs@data[,HMC.list[2]]<1, 0.2,
+     ngs@data[,HMC.list[2]])
> for(hmc in HMC.list[-c(1,2)]){
>   ngs@data[,hmc] <- ifelse(ngs@data[,hmc]<0, abs(ngs@data[,hmc])/2, ngs@data[,hmc])
> }
# check the summary statistics now:
> summary(ngs@data[,HMC.list[1]])
```

```
      Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
     2.500    5.000    5.000    8.843   10.000 5870.000 3164.000
```

Now that we have filtered negative values, we can take a look at the histograms and cross-correlations between HMCs. We assume that all distributions are skewed and hence use in further analysis the log-transformed values (Fig. 6.1):

```
> HMC.formula <- as.formula(paste(" ~ ", paste("log1p(", HMC.list, ")",
+      collapse="+"), sep=""))
> HMC.formula ~ log1p(AS_ICP40) + log1p(CD_ICP40) + log1p(CR_ICP40) +
+     log1p(CU_ICP40) + log1p(NI_ICP40) + log1p(ZN_ICP40) + log1p(AS_AA) +
+     log1p(HG_AA) + log1p(PB_ICP40)
> pc.HMC <- prcomp(HMC.formula, scale=TRUE, ngs@data)
> biplot(pc.HMC, arrow.len=0.1, xlabs=rep(".", length(pc.HMC$x[,1])),
+      main="PCA biplot", xlim=c(-0.04,0.02), ylim=c(-0.03,0.05), ylabs=HM.list)
```

[6] See the complete metadata available in the attached document: ofr-2004-1001.met.

[7] We will make a total of eight maps in fact. Metal Arsenic is measured by using two laboratory methods.

[8] see also http://tin.er.usgs.gov/geochem/faq.shtml

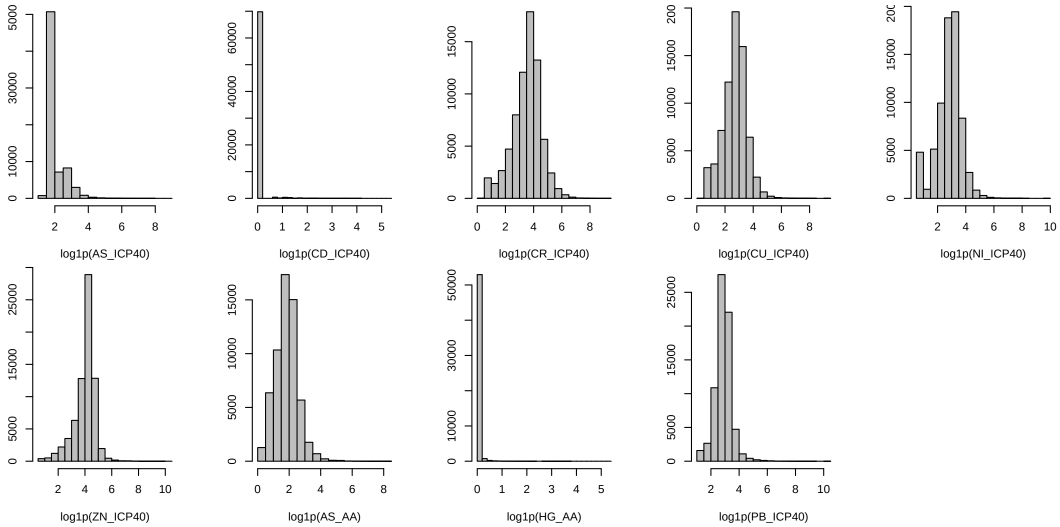

Fig. 6.1: Histograms for log-transformed target variables (HMCs). Note that variables Cd and Hg show skewness even after the transformation.

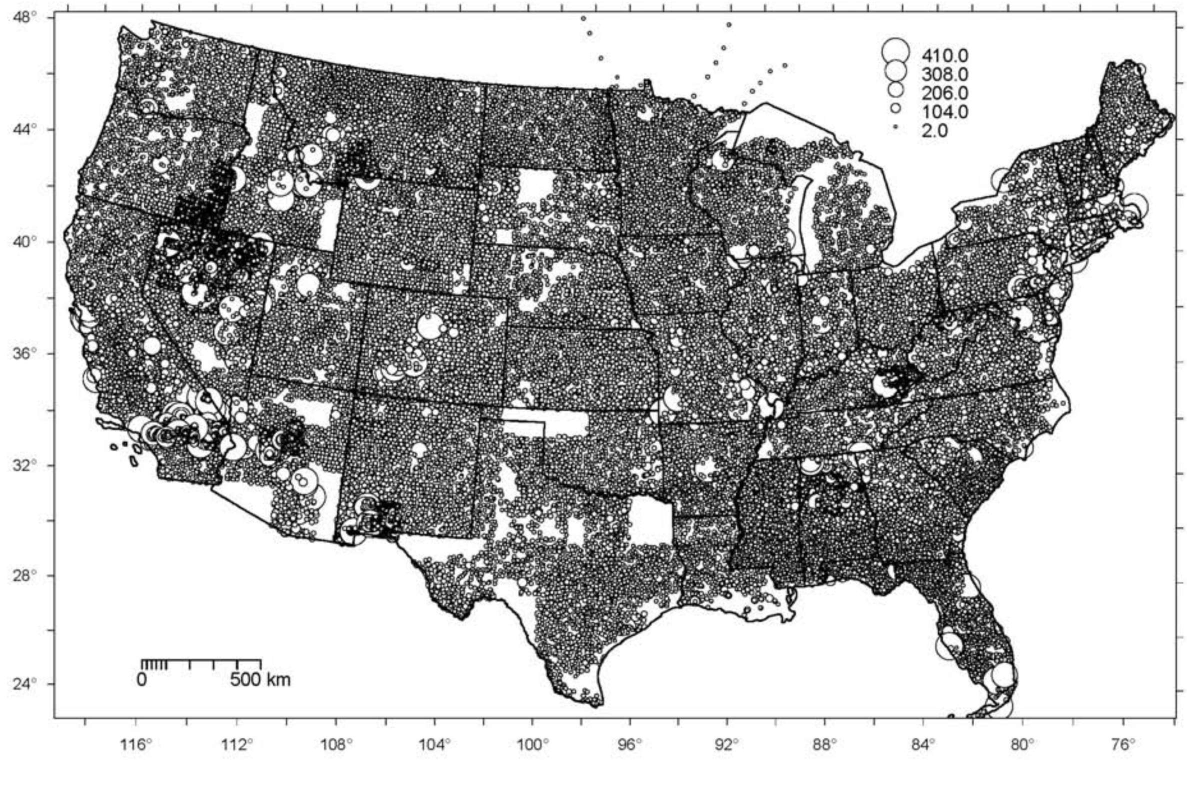

Fig. 6.2: Sampling locations and values of Pb (ppm) based on the NGS data set. Notice the large areas completely unsampled (Indian reservations and military controlled areas), while on the other hand, many states and/or regions have been really densely sampled. Visualized in ILWIS GIS.

The biplot (Fig. 6.3) shows two interesting things: (1) it appears that there are two main clusters of values — Zn, Cu, Ni, Cr, and As, Pb, Hg, Cd; (2) HMCs in both clusters are positively correlated, which means that if e.g. values of Zn increase, so will the values of Cu, Ni, Cr. Similar properties can be noticed with the HMCs in Europe (Rodriguez Lado et al., 2009).

If we look at individual correlations we can see that the most correlated heavy metals are: Cu, Ni and Zn (r=0.76), Cr and Zn (r=0.62), As and Zn (r=0.56). Note also that, at this stage, because the density of points is so high, and because distributions of the target variables are skewed, it is hard to see any spatial pattern in the HMC values (see Fig. 6.2).

6.2.2 Gridded predictors

A sound approach to geostatistical mapping is to first consider all factors that can possibly control the spatial distribution of the feature of interest, and then try to prepare maps that can represent that expert knowledge (Koptsik et al., 2003; Pebesma, 2006). For example, we can conceptualize that the distribution of HMCs is controlled by the a number of environmental and anthropogenic factors: (a) geology, (b) continuous industrial activities — traffic, heating plants, chemical industry and airports, (c) historic land use and mining activities, and (d) external factors and random events — spills and accidents, transported and wind-blown materials.

Because for USA a large quantity of GIS layers[9] is available from the USGS[10] or the National Atlas[11], we have at our disposal a wide variety of predictors:

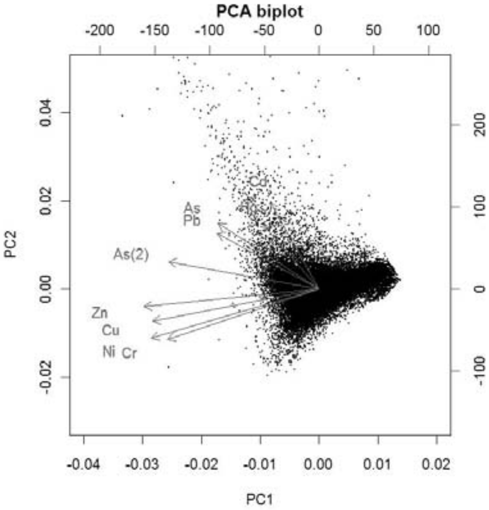

Fig. 6.3: HMCs shown using a PCA biplot.

Urbanization level — Urbanization level can be represented by using images of lights at night (Fig. 6.4; `nlights03.asc`). These are typically highly correlated with industrial activity. The assumption is that HMCs will increase in areas of high urbanization.

Density of traffic — Maps of main roads and railroads can be used to derive density[12] of transportation infrastructure (`sdroads.asc`). In addition, we can use the kernel density of airport traffic (`dairp.asc`), derived using the total enplanements (ranging from few hundreds to >30 million) as weights.

Density of mineral operations — The National Atlas contains an inventory of all major mineral operations (ferrous and nonferrous metal mines). These layers can be merged to produce a kernel density map (`dmino.asc`) showing overall intensity of mineral exploration. In addition, we can also consider the type of mineral exploration (`minotype.asc`; 67 classes), which should help us explain the local hot spots for different heavy metals.

Density of Earthquakes — The magnitude of significant United States Earthquakes (1568–2004) can be used to derive the overall intensity of earthquakes (`dquksig.asc`). We are not certain if this feature controls the distribution of HMCs, but it quantifies geological activities, *ergo* it could also help us explain background concentrations.

Industrial pollutants — The pan-American Environmental Atlas of pollutants (35,000 industrial facilities in North America that reported on releases or transfers of pollutants in 2005) can be used to derive the density of toxic releases (`dTRI.asc`, Fig. 6.4). This feature should likewise be able to explain local hot-spots for heavy metals.

Geological stratification — Geological map at scale 1:1,000,000 (`geomap.asc`; 39 classes) is available from USGS (Fig. 6.4). We can assume that, within some geological units, concentrations will be more homogenous.

[9] See also section 7.2.1.

[10] `http://www.usgs.gov/pubprod/data.html`

[11] `http://nationalatlas.gov/`

[12] See function `Segment density` in ILWIS GIS.

Geomorphological characteristics — From the global Digital Elevation Model (globedem.asc), a number of DEM parameters of interest can be derived: Topographic Wetness Index (twi.asc), visible sky (vsky.asc) and wind effect index (winde.asc). This can help us model HMCs carried by wind or water.

Green biomass — Green biomass can be represented with the Global Biomass Carbon Map that shows carbon density in tons of C ha^{-1} (gcarb.asc, Fig. 6.4). We assume that areas of high biomass amortize pollution by HMCs, and are inversely correlated with HMCs.

Wetlands areas — Because the HMCs have been sampled in various mediums (streams, rivers, lakes, soils, rocks etc.), we also need to use the map of lakes and wetlands (glwd31.asc) to distinguish eventual differences.

The maps listed above can be directly obtain from the data repository, and then extracted to a local directory:

```
> download.file("http://spatial-analyst.net/book/system/files/usgrids5km.zip",
+     destfile=paste(getwd(), "usgrids5km.zip", sep="/"))
> grid.list <- c("dairp.asc", "dmino.asc", "dquksig.asc", "dTRI.asc", "gcarb.asc",
+     "geomap.asc", "globedem.asc", "minotype.asc", "nlights03.asc", "sdroads.asc",
+     "twi.asc", "vsky.asc", "winde.asc", "glwd31.asc")
> for(j in grid.list){
>   fname <- zip.file.extract(file=j, zipname="usgrids5km.zip")
>   file.copy(fname, paste("./", j, sep=""), overwrite=TRUE)
> }
```

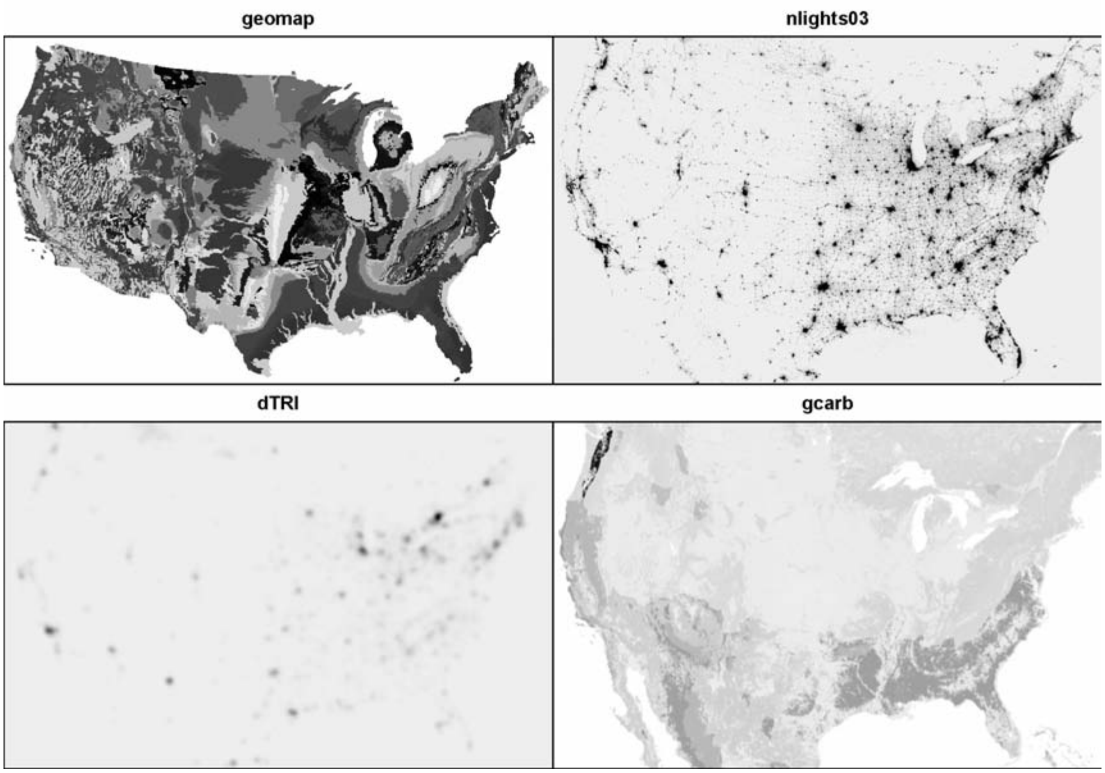

Fig. 6.4: Examples of environmental predictors used to interpolate HMCs: geomap — geological map of US; nlights03 — lights and night image for year 2003; dTRI — kernel density of reported toxic releases; gcarb — biomass carbon map. Visualized using the image method of the adehabitat package.

In total, we have at our disposal 14 gridded maps, possibly extendable to 130 grids if one also includes the indicators (geomap.asc, minotype.asc, and glwd31.asc maps are categories). Note also that, although all layers listed above are available in finer resolutions (1 km or better), for practical reasons we will work with the 5 km resolution maps.

The gridded maps are projected in the Albers equal-area projection system:

```
# read grids into R:
> gridmaps <- readGDAL(grid.list[1])
> names(gridmaps)[1] <- sub(".asc", "", grid.list[1])
> for(i in grid.list[-1]) {
>   gridmaps@data[sub(".asc", "", i[1])] <- readGDAL(paste(i))$band1
> }
> AEA <- "+proj=aea +lat_1=29.5 +lat_2=45.5 +lat_0=23 +lon_0=-96 +x_0=0 +y_0=0
+         +ellps=GRS80 +datum=NAD83 +units=m +no_defs"
> proj4string(gridmaps) <- CRS(AEA)
```

which is often used to display the whole North American continent.

In the same zip file (`usgrids5km.zip`), you will also find a number of ASCII files with the extension `*.rdc`[13]. The `*.rdc` file carries the complete layer metadata, which allows easy access and editing. This is an example of a description file for the Global Lakes and Wetlands map (categorical variable):

```
file format : Arc/Info ASCII Grid
file title  : glwd31.asc
last update : 12.07.2009
producer    : T. Hengl
lineage     : The GLWD31 1 km grid was resampled to 5 km grid.
data type   : byte
file type   : ASCII
columns     : 940
rows        : 592
meas. scale : categorical
description : Global Lakes and Wetlands
proj4string : +proj=aea +lat_1=29.5 +lat_2=45.5 +lat_0=23 +lon_0=-96
+x_0=0 +y_0=0 +ellps=GRS80 +datum=NAD83 +units=m +no_defs
ref. system : projected
ref. units  : meters
unit dist.  : 1
min. X      : -2405000
max. X      : 2295000
min. Y      : 260000
max. Y      : 3220000
pos'n error : 1000
resolution  : 5000
min. value  : 1
max. value  : 12
display min : 1
display max : 12
value units : factor
value error : unspecified
flag value  : -1
flag def'n  : unavailable data
src. English: Global Lakes and Wetlands v3.1
src. URL    : http://www.worldwildlife.org/science/data/item1877.html
src. scale  : 1000 m
src. refs   : Lehner, B., Doll, P., 2004. Development and validation of a global
 database of lakes, reservoirs and wetlands. Journal of Hydrology 296(1-4): 1-22.
src. date   : 2003
src. owner  : WWF; GLWD is available for non-commercial scientific, conservation,
 and educational purposes.
legend cats : 13
category   0: other classes
```

[13] Idrisi GIS format raster (image) documentation file.

```
category   1: Lake
category   2: Reservoir
category   3: River
category   4: Freshwater Marsh, Floodplain
category   5: Swamp Forest, Flooded Forest
category   6: Coastal Wetland
category   7: Pan, Brackish/Saline Wetland
category   8: Bog, Fen, Mire (Peatland)
category   9: Intermittent Wetland/Lake
category  10: 50-100% Wetland
category  11: 25-50% Wetland
category  12: Wetland Complex (0-25% Wetland)
```

such metadata will become important once we start doing interpretation of the results of model fitting — we might need to check the original document describing how was the map produced, what each legend category means etc.

6.3 Model fitting

6.3.1 Exploratory analysis

Before we run any geostatistical predictions, we can simply generate a raster map showing the general spatial pattern of the target variable by using a mechanical interpolator. Because this is a data set with a fairly dense sampling density, the easiest way to generate a complete map from points is to use the "*Close gap*" operation in SAGA. First, we need to convert point data to a raster map:

```
# prepare the mask map:
> rsaga.esri.to.sgrd(in.grids="geomap.asc",
+      out.sgrds="geomap.sgrd", in.path=getwd())
# reproject to the local coordinate system:
> ngs.aea <- spTransform(ngs, CRS(AEA))
# write each element and convert to a raster map:
> for(hmc in HMC.list){
>   writeOGR(subset(ngs.aea, !is.na(ngs.aea@data[,hmc]))[hmc],
+        paste(hmc, ".shp", sep=""), paste(hmc), "ESRI Shapefile")
>   rsaga.geoprocessor(lib="grid_gridding", module=0, param=list(GRID=paste(hmc,
+       ".sgrd",sep=""), INPUT=paste(hmc, ".shp",sep=""), FIELD=0, LINE_TYPE=0,
+       USER_CELL_SIZE=cell.size, USER_X_EXTENT_MIN=gridmaps@bbox[1,1]+cell.size/2,
+       USER_X_EXTENT_MAX=gridmaps@bbox[1,2]-cell.size/2,
+       USER_Y_EXTENT_MIN=gridmaps@bbox[2,1]+cell.size/2,
+       USER_Y_EXTENT_MAX=gridmaps@bbox[2,2]-cell.size/2))
# close gaps (linear interpolation):
>   rsaga.geoprocessor(lib="grid_tools", module=7, param=list(INPUT=paste(hmc,
+       ".sgrd", sep=""), MASK="geomap.sgrd", RESULT=paste(hmc, ".sgrd", sep="")))
> }
```

which will produce the maps shown below (Fig. 6.5). Although these maps seem to be rather complete, they can also be very misleading because we have produced them by completely ignoring landscape features, geology, anthropogenic sources, heterogeneity in the sampling density, spatial auto-correlation effects and similar.

6.3.2 Regression modeling using GLM

Before we can correlate HMCs with environmental predictors, we need to obtain values of predictor grids at sampling locations:

```
> ngs.ov <- overlay(gridmaps, ngs.aea)
> ngs.ov@data <- cbind(ngs.ov@data, ngs.aea@data)
```

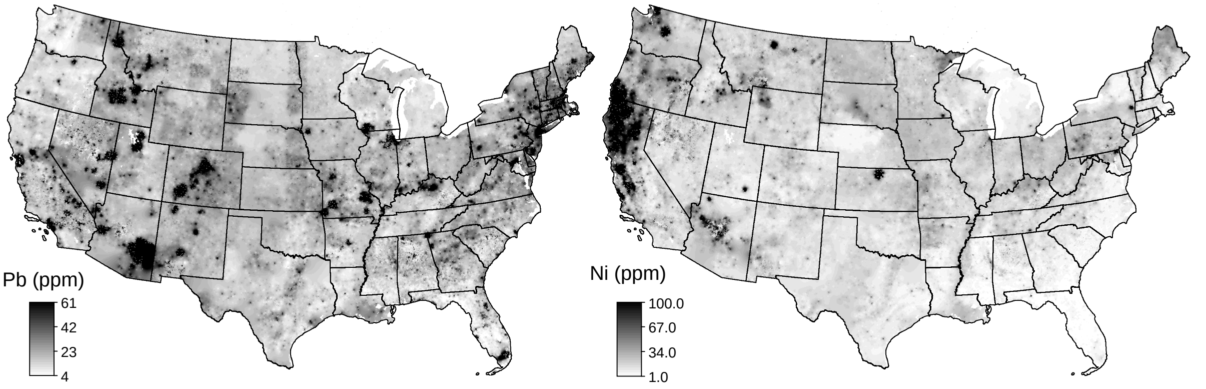

Fig. 6.5: General spatial pattern in values of Pb and Ni generated using interpolation from gridded data (close gap operation in SAGA). Compare further with the maps generated using regression-kriging (as shown in Fig. 6.9).

which creates a matrix with 9+14 variables. This allows us to do some preliminary exploration, e.g. to see for example how much the geological mapping units differ for some HMCs. Let us focus on Pb:

```
> boxplot(log1p(ngs.ov@data[,HMC.list[9]]) ~ ngs.ov$geomap.c,
+       col=grey(runif(levels(ngs.ov$geomap.c))), ylim=c(1,8))
```

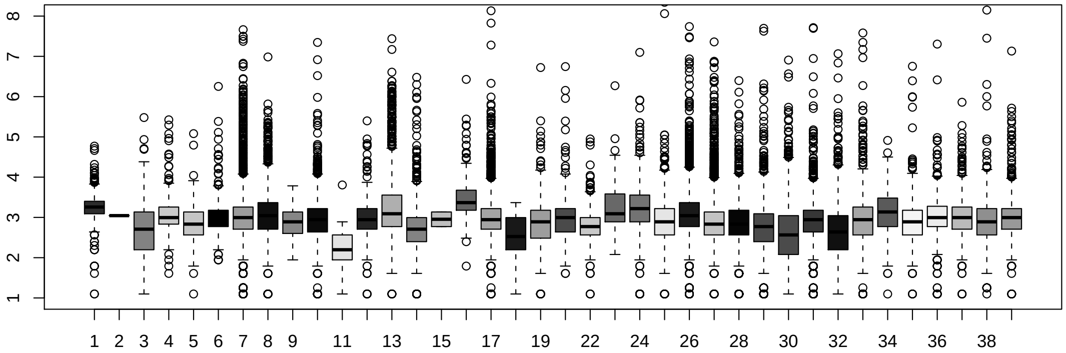

Fig. 6.6: Boxplot showing differences in values of Pb for different geological mapping units (39).

which shows that only a few units (e.g. "1", "11", "16") have distinctly different concentrations of Pb. Some units do not have enough points for statistical analysis, which poses a problem:

```
# how many units need to be masked:
> summary(summary(ngs.ov$geomap.c)<5)
```

```
   Mode   FALSE    TRUE    NA's
logical      37       2       0
```

We can run the same check for the other two categorical maps:

```
> summary(ngs.ov$minotype.c)
```

```
    0     1     2     3     4     5    65    66  NA's
67245     4     0     6    12    13    9     0  6749
```

```
> summary(summary(ngs.ov$minotype.c)<5)
```

```
   Mode   FALSE    TRUE    NA's
logical      27      36       0
```

which shows that there are many units that do not have enough observations for statistical analysis and need to be masked out[14]. We need to do that also with the original grids, because it is important that the regression matrix and the prediction locations contain the same range of classes. We can replace the classes without enough points with dominant classes in the map by using:

```
# determine inappropriate classes (geomap):
> geomap.c.fix <- as.numeric(attr(sort(summary(ngs.ov$geomap.c))
+       [1:sum(summary(ngs.ov$geomap.c)<5)], "names"))
> geomap.c.fix
```

```
[1]  2 15
```

```
> geomap.c.dom <- as.numeric(attr(sort(summary(gridmaps$geomap.c
+       [!is.na(gridmaps$geomap.c)]), decreasing=TRUE)[1], "names"))
# replace the values using the dominant class:
> for(j in geomap.c.fix){
>   gridmaps$geomap <- ifelse(gridmaps$geomap==j, geomap.c.dom, gridmaps$geomap)
> }
> gridmaps$geomap.c <- as.factor(gridmaps$geomap)
# repeat the same for minotype and glwd31...
```

and we can check that the classes with insufficient observations have been replaced:

```
# update the regression matrix:
> ngs.ov <- overlay(gridmaps, ngs.aea)
> ngs.ov@data <- cbind(ngs.ov@data, ngs.aea@data)
> summary(summary(ngs.ov$geomap.c)<5)
```

```
   Mode   FALSE    NA's
logical      37       0
```

Next we can fit a regression model for our sample variable and generate predictions at all grid nodes. Because the values of HMCs are skewed, we will be better off if we fit a GLM model to this data, i.e. using a *Poisson* family with a log link function:

```
> Pb.formula <- as.formula(paste(HMC.list[9], "~", paste(sub(".asc", "",
+      grid.list[-c(6,8,14)]), collapse="+"), "+geomap.c+glwd31.c"))
> Pb.formula
```

```
PB_ICP40 ~ dairp + dmino + dquksig + dTRI + gcarb + globedem +
    nlights03 + sdroads + twi + vsky + winde + geomap.c + glwd31.c
```

```
# fit the model using GLM:
> Pb.lm <- glm(Pb.formula, ngs.ov@data, family=poisson(link="log"))
# predict values at new locations:
> Pb.trend <- predict(Pb.lm, newdata=gridmaps, type="link", na.action=na.omit, se.fit=TRUE)
```

note that the result of prediction is just a data frame with two columns: (1) predicted values in the transformed scale[15] (controlled with `type="link"`, (2) model prediction error (set with `se.fit=TRUE`):

```
> str(Pb.trend)
```

```
List of 3
 $ fit           : Named num [1:314719] 3.55 3.75 3.84 3.71 3.62 ...
  ..- attr(*, "names")= chr [1:314719] "10429" "10430" "10431" "10432" ...
 $ se.fit        : Named num [1:314719] 0.0065 0.00557 0.00529 0.00509 0.00473 ...
  ..- attr(*, "names")= chr [1:314719] "10429" "10430" "10431" "10432" ...
 $ residual.scale: num 1
```

[14] Recall that, by a rule of thumb, we should have at least 5 observations per mapping unit.

[15] We use the transformed scale because we will further sum the interpolated residuals, which are also in the transformed scale.

which means that the coordinates of the grids nodes are not attached any more, and we cannot really visualize or export this data to a GIS. We can reconstruct a gridded map because the grid node names are still contained in the attribute field. This will take several processing steps:

```
# get the coordinates of the original grid:
> pointmaps <- as(gridmaps["globedem"], "SpatialPointsDataFrame")
> sel2 <- as.integer(attr(Pb.trend$fit, "names"))
> rk.Pb <- data.frame(X=pointmaps@coords[sel2,1],
+      Y=pointmaps@coords[as.integer(attr(Pb.trend$fit, "names")),2],
+      HMC=Pb.trend$fit, HMC.var=Pb.trend$se)
> coordinates(rk.Pb) <- ~ X+Y
> gridded(rk.Pb) <- TRUE
# resample to the original grid:
> write.asciigrid(rk.Pb["HMC"], "tmp.asc", na.value=-999)
> rsaga.esri.to.sgrd(in.grids="tmp.asc", out.sgrds="tmp.sgrd", in.path=getwd())
# create an empty grid:
> rsaga.geoprocessor(lib="grid_tools", module=23,
+     param=list(GRID="tmp2.sgrd", M_EXTENT=0,
+     XMIN=gridmaps@bbox[1,1]+cell.size/2, YMIN=gridmaps@bbox[2,1]+cell.size/2,
+     NX=gridmaps@grid@cells.dim[1], NY=gridmaps@grid@cells.dim[2],
+     CELLSIZE=cell.size))
# add decimal places:
> rsaga.geoprocessor("grid_calculus", module=1,
+     param=list(INPUT="tmp2.sgrd", RESULT="Pb_trend_GLM.sgrd", FORMUL="a/100"))
# resample the target grid:
> rsaga.geoprocessor(lib="grid_tools", module=0, param=list(INPUT="tmp.sgrd",
+     GRID="Pb_trend_GLM.sgrd", KEEP_TYPE=FALSE, METHOD=2, SCALE_DOWN_METHOD=0,
+     GRID_GRID="Pb_trend_GLM.sgrd"))
> rsaga.sgrd.to.esri(in.sgrds="Pb_trend_GLM.sgrd", out.grids="Pb_trend_GLM.asc",
+     out.path=getwd())
```

We can quickly check whether the prediction model is efficient in reflecting the original distribution of sampled Pb values:

```
# compare the distributions (95% range of values):
> round(quantile(expm1(Pb.trend$fit), c(.05,.95), na.rm=TRUE), 0)
```

```
 5% 95%
 12  46
```

```
# samples:
> quantile(ngs.ov@data[,HMC.list[9]], c(.05,.95), na.rm=TRUE)
```

```
 5% 95%
  6  41
```

```
# precision:
> sd(residuals(Pb.lm))
```

```
[1] 6.65979
```

If we zoom into the original data, we can notice that there are very few of the original point data that are extremely high (over 5000 times higher than the mean value), most of the values are in the range 6–41 ppm. If we plot the predicted and measured values next to each other, we can notice that the model will have serious problems in predicting both high and low values. There is noticable scatter around the regression line, which also means that the residuals will be significant.

At this stage, it also useful to explore some individual plots between the target variable and predictors. In the case of mapping Pb, it seems that only `dTRI.asc` shows a clear relationship with the target variable, all other correlation plots are less distinct (Fig. 6.7). The good news is that majority of correlations reflect our expectations in qualitative terms: higher concentrations of Pb are connected with higher density of toxic releases and higher industrial activity.

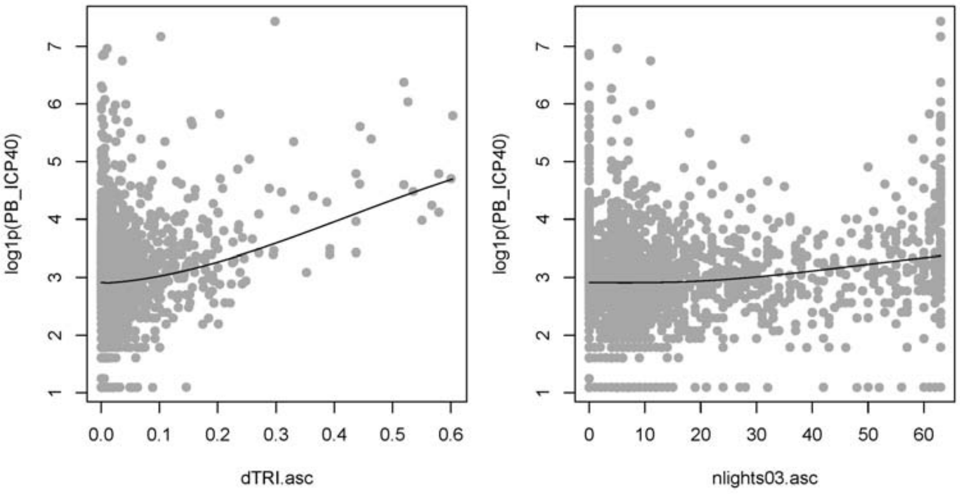

Fig. 6.7: Correlation plots Pb versus some significant predictors: density of toxic release accidents (`dTRI.asc`), and lights at night image (`nlights03.asc`).

6.3.3 Variogram modeling and kriging

Now that we have predicted the trend part of the model, we can proceed with interpolating the residuals. We will also interpolate the residuals in the transformed scale, and not in the response scale, which means that we need to derive them:

```
> residuals.Pb <- log1p(Pb.lm$model[,HMC.list[j]])-log1p(fitted.values(Pb.lm))
```

An important check we need to make is to see that the residuals are normally distributed:

```
> hist(residuals.Pb, breaks=25, col="grey")
# residuals are normally distributed!
```

which is a requirement to interpolate this variable using ordinary kriging.

Fitting a variogram model with >50,000 points in gstat is computationally intensive and would take significant time, especially if we would like to do it using global search radius. Instead, we can speed up the processing by: (a) limiting the search radius, (b) sub-setting the points randomly[16]. To estimate the mean shortest distance between points we can use spatstat package:

```
> library(spatstat)
> ngs.ppp <- as(ngs.aea[1], "ppp")
> boxplot(dist.ngs <- nndist(ngs.ppp), plot=F)$stats

          [,1]
 [1,]     0.000
 [2,]  1020.381
 [3,]  3021.972
 [4,]  7319.961
 [5,] 16766.662

> search.rad <- 2*boxplot(dist.ngs <- nndist(ngs.ppp), plot=F)$stats[5]
```

which shows that the mean shortest distance to the nearest point is about 3 km, none of the points is >17 km away from the first neighbor. To be on the safe side, we can limit the search radius to two times the highest `nndist` i.e. 34 km in this case.

Next, we can prepare a point map with residuals, and randomly sub-sample the data set to 20% of its original size:

[16] Assuming that a large part of variation has already been explained by the GLM model, we can be less accurate about fitting the variogram.

```
> sel <- as.integer(attr(Pb.lm$model, "na.action"))
> res.Pb <- data.frame(X=coordinates(ngs.ov[-sel,])[,1],
+        Y=coordinates(ngs.ov[-sel,])[,2], res=residuals.Pb)
# mask out NA values:
> res.Pb <- subset(res.Pb, !is.na(res.Pb$res))
> coordinates(res.Pb) <- ~ X+Y
> proj4string(res.Pb) <- CRS(AEA)
# sub-sample to 20%!
> res.Pb.s <- res.Pb[runif(length(res.Pb@data[[1]]))<0.2,]
```

so that fitting of the variogram will go much faster:

```
> var.Pb <- variogram(res ~ 1, data=res.Pb.s, cutoff=34000)
> rvgm.Pb <- fit.variogram(var.Pb, vgm(nugget=var(res.Pb$res, na.rm=TRUE)/5,
+      model="Exp", range=34000, psill=var(res.Pb$res, na.rm=TRUE)))
> plot(var.Pb, rvgm.Pb, plot.nu=F, pch="+", cex=2,
+      col="black", main="Vgm for Pb residuals")
```

The variogram shows that the feature is correlated up to the distance of about 10 km; about 50% of sill variation (nugget) we are not able to explain. Use of GLM *Poisson* model is beneficial for further geostatistical modeling — the residuals have a symmetrical distribution; the final predictions will also follow a similar distribution, i.e. they will maintain hot-spot locations, which might have been otherwise smoothed-out if a simple linear regression was used.

To speed up the interpolation[17], we use the SAGA geostatistics module:

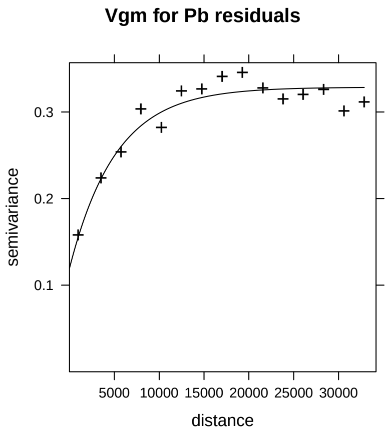

Fig. 6.8: Results of variogram fitting for the Pb GLM-residuals (log-transformed).

```
# export to a shapefile:
> writeOGR(res.Pb, "Pb_res.shp", "Pb_res",
+        "ESRI Shapefile")
# Ordinary kriging in SAGA:
> rsaga.geoprocessor(lib="geostatistics_kriging",
+   module=5, param=list(GRID="Pb_res_OK.sgrd",
+   SHAPES="Pb_res.shp", BVARIANCE=F, BLOCK=F,
+   FIELD=1, BLOG=F, MODEL=1, TARGET=0,
+   NPOINTS_MIN=10, NPOINTS_MAX=60,
+   NUGGET=rvgm.Pb$psill[1], SILL=rvgm.Pb$psill[2],
+   RANGE=rvgm.Pb$range[2],
+   MAXRADIUS=3*search.rad, USER_CELL_SIZE=cell.size,
+   USER_X_EXTENT_MIN=gridmaps@bbox[1,1]+cell.size/2,
+   USER_X_EXTENT_MAX=gridmaps@bbox[1,2]-cell.size/2,
+   USER_Y_EXTENT_MIN=gridmaps@bbox[2,1]+cell.size/2,
+   USER_Y_EXTENT_MAX=gridmaps@bbox[2,2]-cell.size/2))
```

Finally, we can combine the two maps (predicted trend and interpolated residuals) to produce the best estimate of the Pb values (Fig. 6.9):

```
# sum the regression and residual part:
> rsaga.sgrd.to.esri(in.sgrds="Pb_rk.sgrd", out.grids="Pb_rk.asc", out.path=getwd())
> gridmaps@data[,"Pb_rk"] <- exp(readGDAL("Pb_rk.asc")$band1)
> spplot(gridmaps["Pb_rk"], col.regions=grey(rev((1:59)^2/60^2)), at=seq(4,250,5))
```

[17] The data set consists of >50,000 points! Even if we are using a small search radius, this data set will always take a significant amount of time to generate predictions.

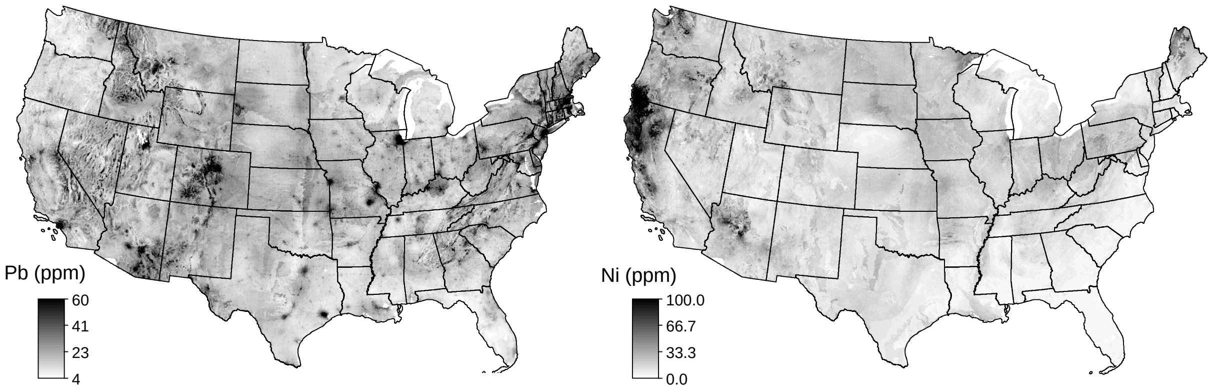

Fig. 6.9: Distribution of Pb and Ni predicted using regression-kriging. Note that many local hot-spots from Fig. 6.5 have been now smoothed out by the kriging algorithm.

6.4 Automated generation of HMC maps

Now that we have become familiar with the geostatistical steps, i.e. now that we have tested different methods and tidy up the R code, we can pack all the steps together. The results of fitting we will save as lists, so that we can review them later on; all other temporary files we can recycle[18]:

```
# generate empty lists:
> formula.list <- as.list(rep(NA, length(HMC.list)))
> vgm.list <- as.list(rep(NA, length(HMC.list)))
> vgmplot.list <- as.list(rep(NA, length(HMC.list)))
> for(j in 1:length(HMC.list)){
# fit a GLM:
>   formula.list[[j]] <- as.formula(paste(HMC.list[j], "~",
+      paste(sub(".asc", "", grid.list[-c(6,8,14)]), collapse="+"), "+geomap.c+glwd31.c"))
>   glm.HMC <- glm(formula.list[[j]], ngs.ov@data, family=poisson(link="log"))
...
# sum the regression and residual part:
>   rsaga.geoprocessor("grid_calculus", module=1,
+     param=list(INPUT=paste(HM.list[j], "_trend_GLM.sgrd", ";", HM.list[j],
+     "_res_OK.sgrd", sep=""), RESULT=paste(HM.list[j], "_rk.sgrd", sep=""), FORMUL="a+b"))
>   rsaga.sgrd.to.esri(in.sgrds=paste(HM.list[j], "_rk.sgrd", sep=""),
+     out.grids=paste(HM.list[j], "_rk.asc", sep=""), out.path=getwd())
>   gridmaps@data[,paste(HM.list[j], "_rk", sep="")] <- exp(readGDAL(paste(HM.list[j],
+     "_rk.asc", sep=""))$band1)
>   write.asciigrid(gridmaps[paste(HM.list[j], "_rk", sep="")],
+     paste(HM.list[j], "_rk.asc", sep=""), na.value=-1)
> }
```

In summary, the script follows previously described steps, namely:

(1.) Fit the GLM using the regression matrix. Derive the residuals (log-scale) and export them to a shapefile.

(2.) Predict values using the fitted GLM. Convert the predictions to the same grid as the predictor maps.

(3.) Fit the variogram model for residuals. Save the fitted variogram parameters and the variogram plot.

(4.) Interpolate the residuals using ordinary kriging in SAGA.

(5.) Sum the predicted trend (GLM) and residuals (OK) and import the maps back into R.

[18] This will take a lot of your memory, hence consider using `gc()` to release some memory from time to time. It is especially important to recycle the results of GLM modeling. The resulting GLM object will often take a lot of memory because it makes copies of the original data set, masked observations and observations used to build the model.

(6.) Back-transform the values to the original scale; export the map to a GIS format.

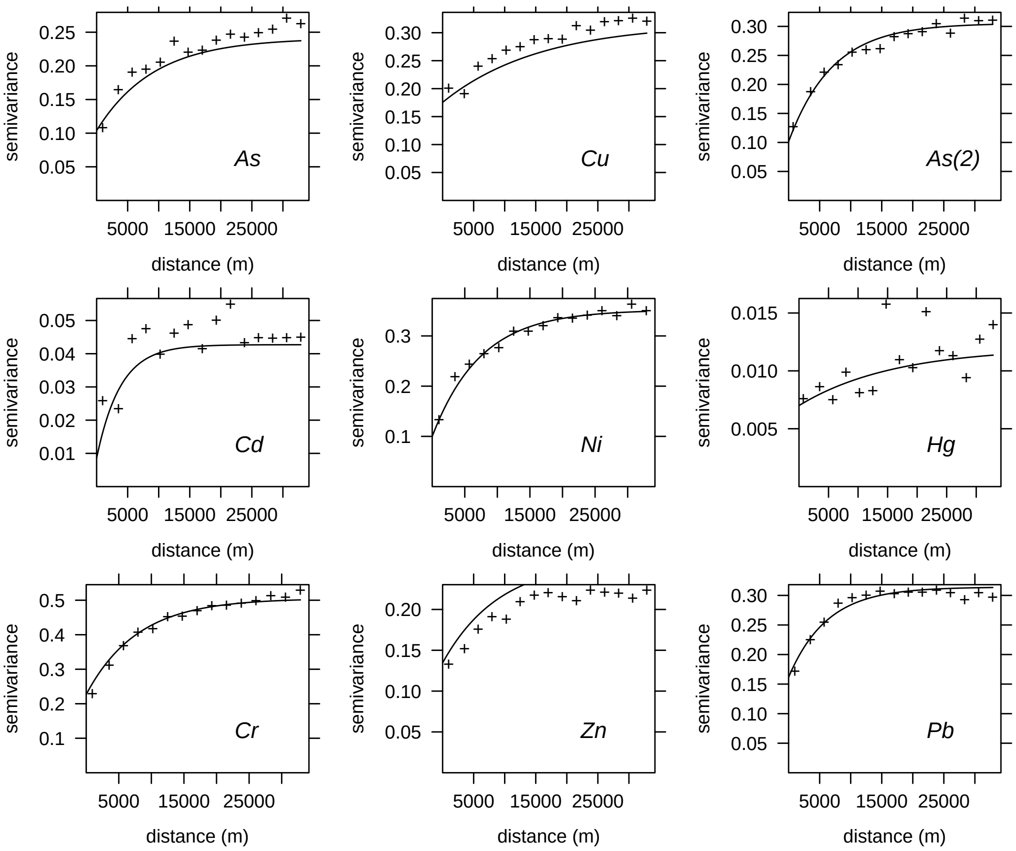

Fig. 6.10: Variograms fitted for GLM residuals.

To review the results of model fitting we can best look at the fitted variograms (Fig. 6.10). If the variograms are stable and fitted correctly, and if they do not exceed the physical range of values, we can be confident that the predictions will be meaningful. In this case, we can see that all variograms have a standard shape, except for Hg, which seems to show close to pure nugget effect. We can repeat the variogram fitting *by-eye* for this HMC, and then re-interpolate the data, at least to minimize artifacts in the final map. Note also that the nugget variation is significant for all variables.

The final predictions for various HMCs can be used to extract the principal components, i.e. reduce eight maps to two maps. Recall from Fig. 6.3 that there are basically two big groups of HMCS: Zn, Cu, Ni, Cr; and As, Pb, Hg, Cd. The first component derived using these maps is shown in Fig. 6.11. This map can be considered to show the overall pollution by HMCs with '*industrial*' origin (As, Pb, Hg and Cd) for the whole of USA.

Based on the results of analysis, we can conclude the following. First, auxiliary maps such as density of toxic releases, urbanization intensity, geology and similar, can be used to improve interpolation of various heavy metals. For example, distribution of Pb can be largely explained by density of toxic releases and night light images, several heavy metals can be explained by geological soil mapping units. Second, selected heavy metals are positively correlated — principal component plots for NGS are similar to the results of the European case study (Rodriguez Lado et al., 2009). Third, most of HMCs have distributions skewed towards low values. This proves that HMCs can in general be considered to follow a *Poisson*-type distribution.

This results also confirm that some local hot-spots shown in Fig. 6.5 are not really probable, and therefore have been smoothed out (compare with Fig. 6.9). Interpolation of some HMCs is not trivial. Mercury is, for example, a difficult element for which to obtain accurate analyzes (Grossman et al., 2008). Samples can easily be contaminated with Hg during handling, storage, and preparation for analysis.

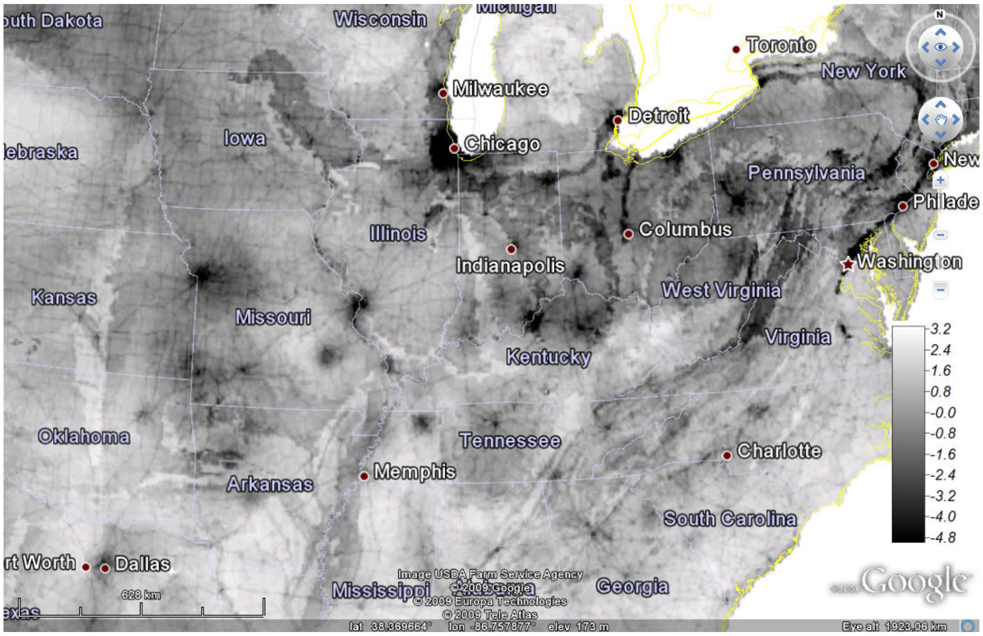

Fig. 6.11: First principal component derived using a stack of predicted maps of eight heavy metals. This PC basically represents mapped overall concentration of As, Pb and Cd (compare with Fig. 6.3); shown as a ground overlay in Google Earth.

6.5 Comparison of ordinary and regression-kriging

Finally we can also run a comparison between OK and RK methods to analyze the benefits of using auxiliary predictors (or are there benefits at all)? The recommended software to run such analysis is geoR, because it is more suited to analyzing skewed variables, and because it provides more insight into the results of model fitting. To speed up processing, we can focus on two US states (Illinois and Indiana) and only the most significant predictors. We can subset (using the bounding box coordinates) the auxiliary maps using:

```
# subset the original predictors:
> grid.list.s <- c("dairp.asc", "dTRI.asc", "nlights03.asc", "sdroads.asc")
> rsaga.esri.to.sgrd(in.grids=grid.list.s, out.sgrds=set.file.extension(grid.list.s,
+     ".sgrd"), in.path=getwd())
> for(i in 1:length(grid.list.s)) {
# first, create a new grid:
>   rsaga.geoprocessor(lib="grid_tools", module=23, param=list(GRID="tmp2.sgrd",
+        M_EXTENT=1, XMIN=360000, YMIN=1555000, XMAX=985000, YMAX=2210000, CELLSIZE=5000))
# 0.01 decimal places:
>   rsaga.geoprocessor("grid_calculus", module=1, param=list(INPUT="tmp2.sgrd",
+        RESULT=paste("m_", set.file.extension(grid.list.s[i], ".sgrd"), sep=""),
+        FORMUL="a/100")) # 0.01 decimal places
# now, resample all grids:
>   rsaga.geoprocessor(lib="grid_tools", module=0,
+        param=list(INPUT=set.file.extension(grid.list.s[i], ".sgrd"),
+        GRID=paste("m_", set.file.extension(grid.list.s[i], ".sgrd"), sep=""),
+        GRID_GRID=paste("m_", set.file.extension(grid.list.s[i], ".sgrd"), sep=""),
+        METHOD=2, KEEP_TYPE=FALSE, SCALE_DOWN_METHOD=0))
> }
> rsaga.sgrd.to.esri(in.sgrds=paste("m_", set.file.extension(grid.list.s, ".sgrd"),
+        sep=""), out.grids=paste("m_", set.file.extension(grid.list.s, ".asc"),
+        sep=""), out.path=getwd(), pre=3)
# read maps into R:
> gridmaps.s <- readGDAL(paste("m_", set.file.extension(grid.list.s[1], ".asc"), sep=""))
> for(i in 2:length(grid.list.s)) {
```

```
>   gridmaps.s@data[i] <- readGDAL(paste("m_", set.file.extension(grid.list.s[i],
+        ".asc"), sep=""))$band1
> }
> names(gridmaps.s) <- sub(".asc", "", grid.list.s)
> str(gridmaps.s@data)
```

```
  'data.frame':   16632 obs. of  4 variables:
   $ dairp    : num  0.031 0.03 0.031 0.032 0.033 ...
   $ dTRI     : num  0.007 0.007 0.007 0.008 0.008 ...
   $ nlights03: num  6 3 6 2 0 4 5 16 5 5 ...
   $ sdroads  : num  0 0 7497 0 0 ...
```

which has resampled the original grids to a 125×131 pixels block. We also need to subset the point data (we focus on Pb) using the same window (X_{min}=360000, X_{max}=985000, Y_{min}=1555000, Y_{max}=2210000) with the help of SAGA module shapes_tools:

```
# subset the point data:
> rsaga.geoprocessor(lib="shapes_tools", module=14,
+     param=list(SHAPES="PB_ICP40.shp", CUT="m_PB_ICP40.shp", METHOD=0, TARGET=0,
+     CUT_AX=360000, CUT_BX=985000, CUT_AY=1555000, CUT_BY=2210000))
> m_PB <- readOGR("m_PB_ICP40.shp", "m_PB_ICP40")
```

```
  OGR data source with driver: ESRI Shapefile
  Source: "m_PB_ICP40.shp", layer: "m_PB_ICP40"
  with  2787  rows and  1  columns
  Feature type: wkbPoint with 2 dimensions
```

which limits the analysis to only 2787 points within the area of interest. We convert the data to the native geoR format:

```
> Pb.geo <- as.geodata(m_PB["PB_ICP40"])
```

```
  as.geodata: 622 redundant locations found
  WARNING: there are data at coincident or very closed locations, some of the geoR's
  functions may not work. Use function dup.coords to locate duplicated coordinates.
```

which shows that there might be some problems for further analysis because there are many duplicate points and the calculation might fail due to singular matrix problems. Even though the data set is much smaller than the original NGS data set, geoR might still have problems running any analysis. Hence, a good idea is to (1) remove duplicates, and (2) randomly subset point data:

```
> m_PB <- remove.duplicates(m_PB)
> str(Pb.geo[[2]])
```

```
   num [1:2165] 9 10 10 9 16 14 8 15 11 9 ...
```

```
> m_PB.ov <- overlay(gridmaps.s, m_PB)
# subset to speed up:
> sel <- runif(length(m_PB@data[[1]]))<0.5
> Pb.geo1 <- as.geodata(m_PB[sel, "PB_ICP40"])
> str(Pb.geo1[[2]])
```

```
   num [1:1120] 9 10 14 11 11 18 14 13 10 8 ...
```

```
# copy values of covariates:
> Pb.geo1$covariate <- m_PB.ov@data[sel, sub(".asc", "", grid.list.s)]
```

We can now proceed with variogram modeling. First, we estimate the variogram for the original variable:

```
> Pb.vgm <- likfit(Pb.geo1, lambda=0, messages=FALSE, ini=c(var(log1p(Pb.geo$data)),
+      50000), cov.model="exponential")
> Pb.vgm
```

```
  likfit: estimated model parameters:
         beta       tausq     sigmasq        phi
  "2.889e+00" "2.952e-01" "2.170e-01" "5.000e+04"
  Practical Range with cor=0.05 for asymptotic range: 149786.6

  likfit: maximised log-likelihood = -1736
```

then for the residuals[19]:

```
> Pb.rvgm <- likfit(Pb.geo1, lambda=0, trend= ~ dairp+dTRI+nlights03+sdroads,
+   messages=FALSE, ini=c(var(log1p(Pb.geo$data))/5, 25000), cov.model="exponential")
> Pb.rvgm

  likfit: estimated model parameters:
         beta0        beta1        beta2        beta3        beta4
  "    2.7999" "   -0.4811" "    2.4424" "    0.0022" "    0.0000"
         tausq      sigmasq          phi
  "    0.2735" "    0.1737" "24999.9999"
  Practical Range with cor=0.05 for asymptotic range: 74893.3

  likfit: maximised log-likelihood = -2763
```

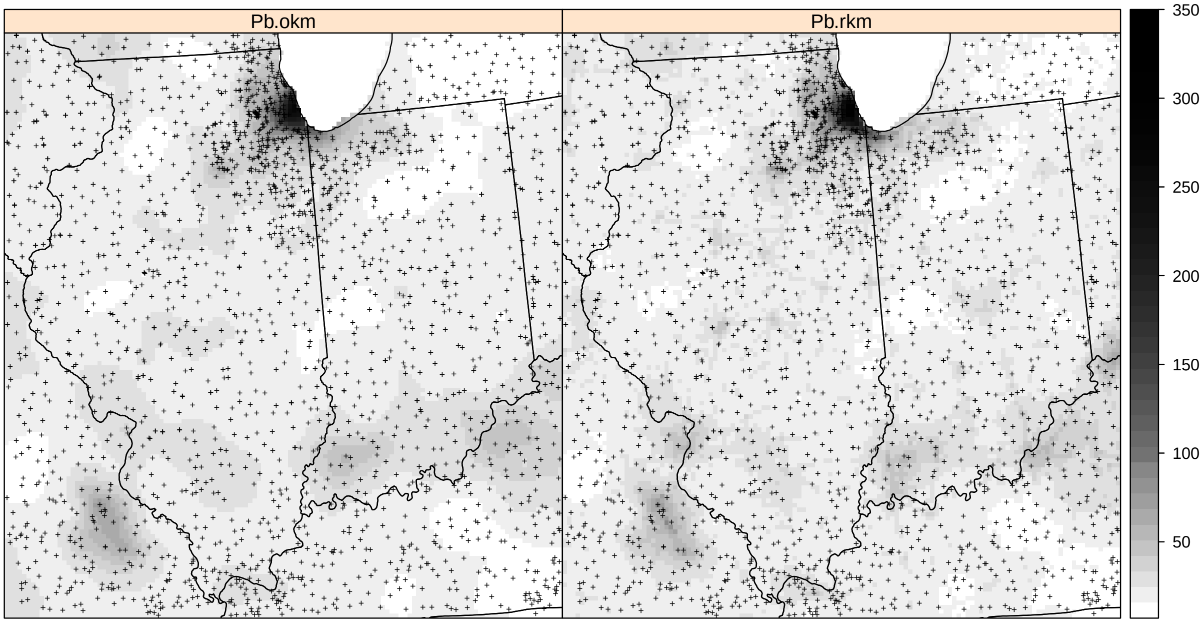

Fig. 6.12: Comparison of results of predicting values of Pb (ppm) using ordinary and regression-kriging (subset of 1120 points) for two US states (Illinois and Indiana). See text for more details.

Now that we have fitted the geostatistical model, we can prepare the prediction locations and run both ordinary and regression-kriging:

```
# prepare the covariates:
> locs.sp <- locs
> coordinates(locs.sp) <- ~ Var1+Var2
> gridmaps.gr <- overlay(gridmaps.s, locs.sp)
# Ordinary kriging:
> Pb.ok <- krige.conv(Pb.geo1, locations=locs, krige=krige.control(obj.m=Pb.vgm))
```

[19] These are residuals fitted using linear modeling, but after the Box-Cox transformation.

```
  krige.conv: model with constant mean
  krige.conv: performing the Box-Cox data transformation
  krige.conv: back-transforming the predicted mean and variance
  krige.conv: Kriging performed using global neighbourhood
```

```
# Regression-kriging:
> KC <- krige.control(trend.d = ~ dairp+dTRI+nlights03+sdroads, trend.l =
+    ~ gridmaps.gr$dairp+gridmaps.gr$dTRI+gridmaps.gr$nlights03+gridmaps.gr$sdroads,
+    obj.m = Pb.rvgm)
> Pb.rk <- krige.conv(Pb.geo1, locations=locs, krige=KC)
```

```
  krige.conv: model with mean defined by covariates provided by the user
  krige.conv: performing the Box-Cox data transformation
  krige.conv: back-transforming the predicted mean and variance
  krige.conv: Kriging performed using global neighbourhood
```

This time we did not use any mask (border coordinates), so that we need to mask the water bodies after converted the data to sp class (compare with §5.5.3):

```
# sp plot:
> locs.geo <- data.frame(X=locs.sp@coords[,1], Y=locs.sp@coords[,2],
+     Pb.rk=Pb.rk[[1]], Pb.ok=Pb.ok[[1]], Pb.rkvar=Pb.rk[[2]], Pb.okvar=Pb.ok[[2]])
> coordinates(locs.geo) <- ~ X+Y
> gridded(locs.geo) <- TRUE
# mask out water bodies:
> mask.s <- as.vector(t(as.im(gridmaps.s["geomap"])$v))  # flip pixels up-side down
> locs.geo$Pb.ok <- ifelse(is.na(mask.s), NA, locs.geo$Pb.ok)
> locs.geo$Pb.rk <- ifelse(is.na(mask.s), NA, locs.geo$Pb.rk)
> spplot(locs.geo[c("Pb.ok", "Pb.rk")], col.regions=grey(rev(seq(0,1,0.025)^2)),
+       at=seq(5,350,l=40), sp.layout=list(list("sp.points", m_PB, pch="+", col="black"),
+       list("sp.lines", USA.borders, col="black")))
> summary(locs.geo$Pb.okvar)
```

```
      Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
     25.91   145.80   198.40   336.10   271.00 26860.00
```

```
> summary(locs.geo$Pb.rkvar)
```

```
     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
    22.13   144.10   190.40   306.50   258.60 42200.00
```

The results are illustrated in Fig. 6.12. RK does seem to be more efficient in reflecting the spatial pattern of industrial activities and pollution sources. The range of values in the map predicted using RK is somewhat higher, which is due the fact that we have now used geoR package that deals very well with skewed distributions so that many peaks, invisible in the OK predictions map, have been emphasized with RK. The prediction error of RK is, as expected, somewhat smaller than for OK, but the difference is rather small (378 vs 391). This is also because the prediction error of the RK is proportionally higher in areas with high values, so that the overall average error is higher.

Self-study exercises:

(1.) At which locations are the maps shown in Fig. 6.5 and 6.9 the most different? (HINT: derive and plot a difference map)

(2.) Which predictors are most highly correlated with each other? Plot first and second component derived using all maps — what do they reflect?

(3.) Which HMC is the most difficult to interpolate? (HINT: look at the residuals of the regression model, nugget parameter etc.)

(4.) Split the original NGS point data set based on source (stream sediments, soils etc.) and then repeat the analysis for at least three HMCs. Is there a difference between the regression models? (HINT: plot correlation lines for various media in the same graph.)

(5.) How much does the accuracy in the HMC maps decrease if we use only 10% of samples (randomly selected) versus the complete data set? (HINT: use standard measures described in §1.4 to run a comparison.)

(6.) Which US state has highest concentration of Pb on average?

(7.) Run the cross-validation following the exercise in §6.5 and see if the predictions made using RK are significantly better than with OK.

(8.) Compare the accuracy of predictions for element Pb using ordinary kriging on untransformed data and using the Box–Cox transformation — are there significant differences? (HINT: randomly split the NGS data set to two equal size data sets; then use one for validation only.)

Further reading:

★ Grossman, J. N. and Grosz, A. E. and Schweitzer, P. N. and Schruben, P. G., 2008. The National Geochemical Survey — Database and Documentation, Version 5. U.S. Geological Survey, Reston, VA.

★ Papritz, A. and Reichard, P. U., 2009. Modeling the risk of Pb and PAH intervention value exceedance in allotment soils by robust logistic regression. Environmental Pollution, 157(7): 2019–2022.

★ Reimann, C., Filzmoser, P., Garrett, R., Dutter, R., 2008. **Statistical Data Analysis Explained Applied Environmental Statistics with R**. Wiley, Chichester, 337 p.

★ Rodriguez Lado, L. and Hengl, T. and Reuter, H. I., 2009. Heavy metals in European soils: a geostatistical analysis of the FOREGS Geochemical database. Geoderma, 148(2): 189–199.

★ `http://tin.er.usgs.gov/geochem/` — The National Geochemical Survey website.

★ `http://www.gtk.fi/publ/foregsatlas/` — The Geochemical atlas of Europe.