Warsaw University of Technology
Faculty of Mathematics and Information Science

Spatial Data in R

Marcin Bujarski, Natalia Potocka

Warsaw, 09.06.2014

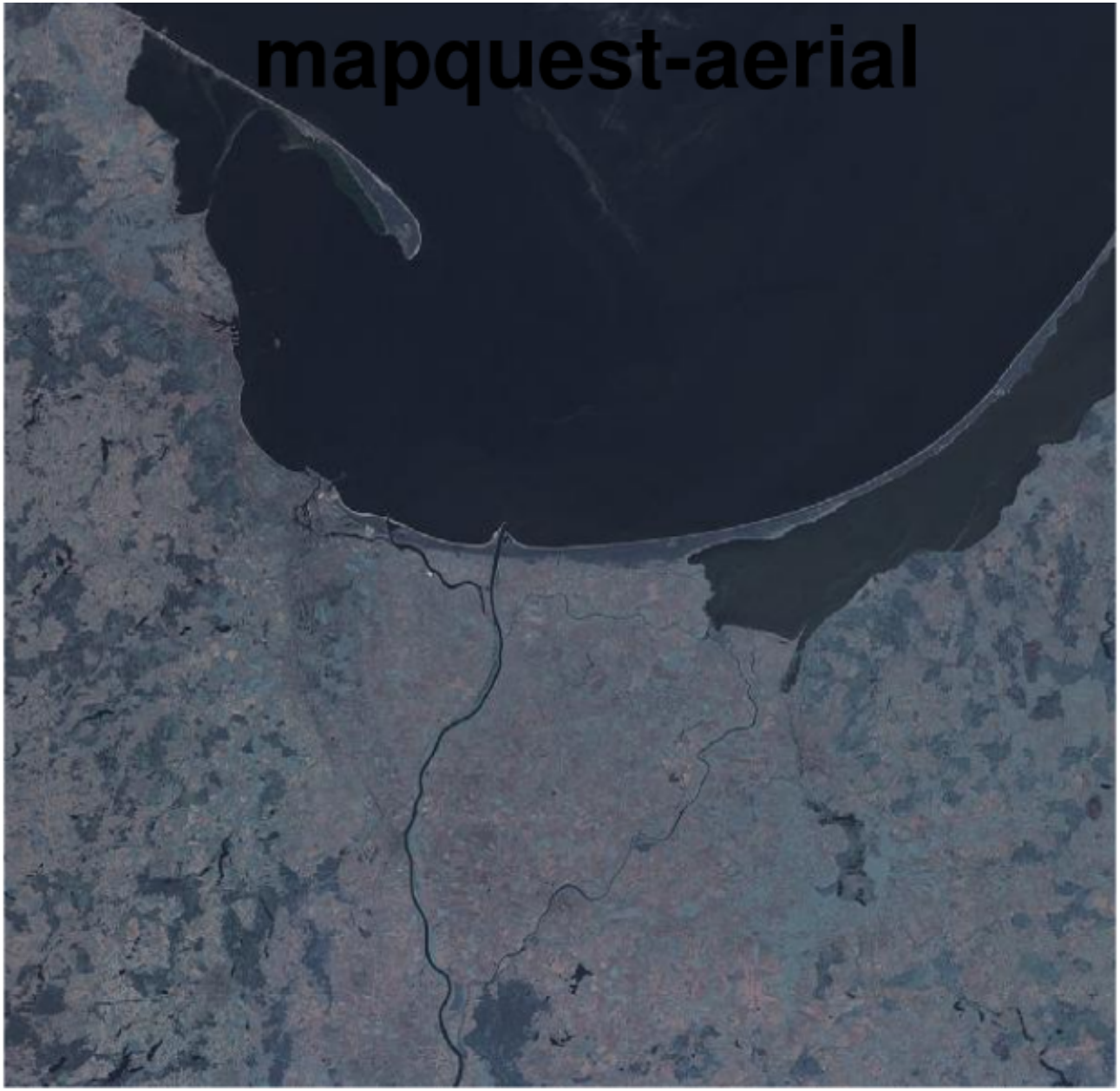
OpenStreetMap

```
map <- openmap(upperLeft = c(lat= 54.8, lon= 18.2), lowerRight
= c(lat= 53.9, lon= 19.8), type="osm")
```

```
plot(map)
```


osm
Władysławowo
Puck
Jastarnia
Hel
Rumia
Gdynia
Sopot
Gdańsk
Żukowo
Krynica Morska
Frombork
Tolkmicko
Pruszcz Gdański
Nowy Dwór Gdański
Młynary
Elbląg
Nowy Staw
Tczew
Skarszewy
Malbork
Pasłęk
Starogard Gdański
Pelplin
Sztum
Dzierzgoń


maptoolkit-topo
Władysławowo
Jastarnia
Żelistrzewo
Hel
Reda
Rumia
Leszczynki
Gdynia
Dąbrowa
Sopot
Piaski
Przymorze Wielkie
Piecki-Migowo
Gdańsk
Sztutowo
Tolkmicko
Orunia-Św. Wojciech-Lipce
Łapino
Leszków
Nowy Dwór Gdański
Przywidz
Pszczółki
Młynary
Elbląg
Nowy Staw
Tczew
Robity
Skarszewy
Markusy
Malbork
Krąg
Rychliki
Starogard Gdański
Zblewo
Pelplin
Sztum
Dzierzgoń
Małdyty


mapquest-aerial


bing


esri
GDYNIA
Gdansk
Elblag
Tczew
Zblewo
7
E77
54
504
S22
221
22
7
Warminsko-mazurskie

Data structure in OpenStreetMap:

- nodes
- ways
- relations

Data structure in OpenStreetMap:

- nodes
- ways
- relations

```
library("osmar")

## Error:  there is no package called 'osmar'

pl <- get_osm(relation(49715), osmsource_api())

## Error:  could not find function "get_osm"
```

```
class(pl)

## Error:  object 'pl' not found
```

```
names(pl)

## Error:  object 'pl' not found
```

```
library("osmar")

## Error:  there is no package called 'osmar'

pl <- get_osm(relation(49715), osmsource_api())

## Error:  could not find function "get_osm"
```

```
class(pl)

## Error:  object 'pl' not found
```

```
names(pl)

## Error:  object 'pl' not found
```

```
library("osmar")

## Error:  there is no package called 'osmar'

pl <- get_osm(relation(49715), osmsource_api())

## Error:  could not find function "get_osm"
```

```
class(pl)

## Error:  object 'pl' not found
```

```
names(pl)

## Error:  object 'pl' not found
```

```
unclass(pl$nodes)

## Error:  object 'pl' not found
```

```
unclass(pl$ways)

## Error:  object 'pl' not found
```

```
names(pl$relations)

## Error:  object 'pl' not found
```

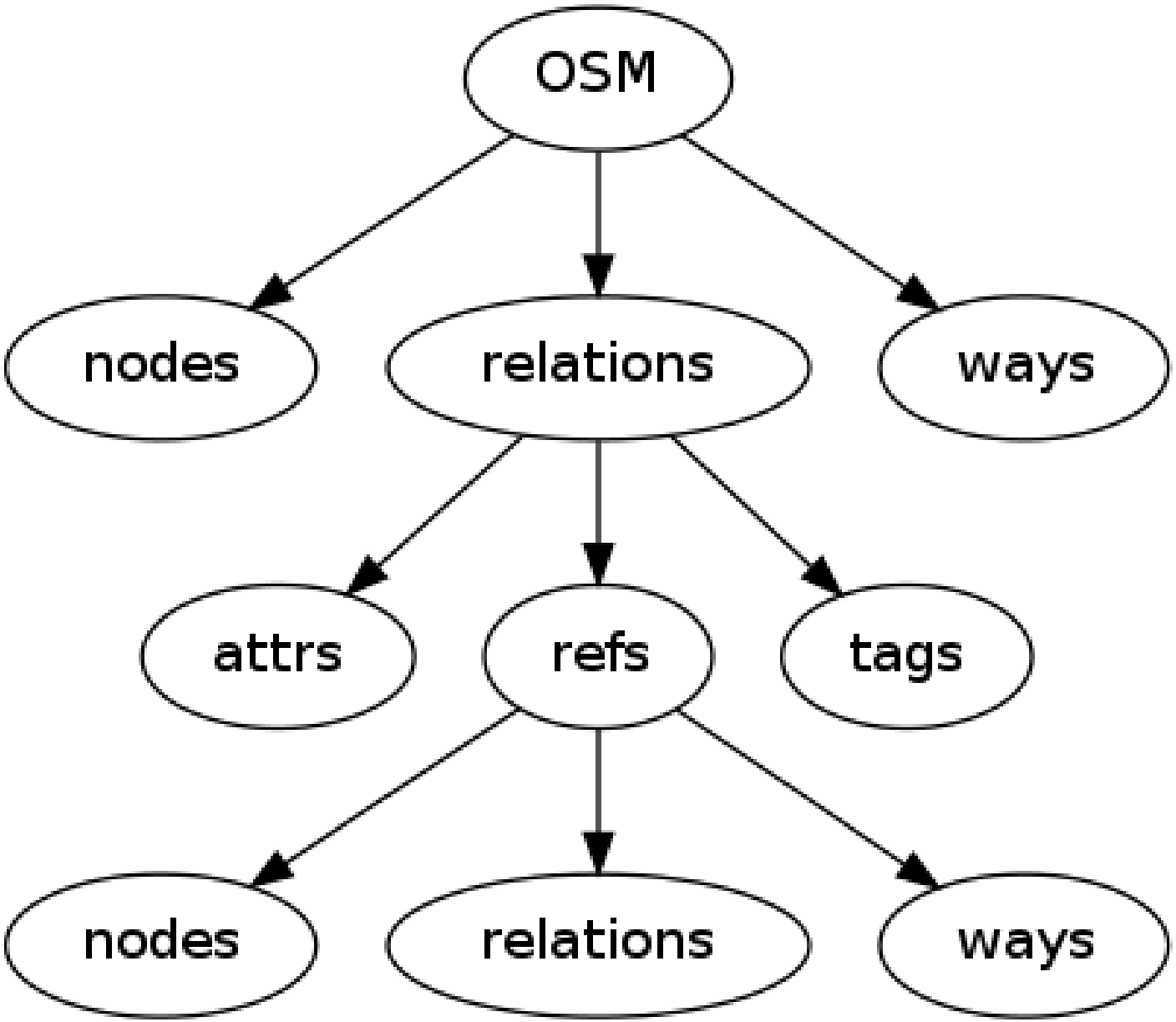
```
unclass(pl$ways)
```

```
## Error:  object 'pl' not found
```

```
names(pl$relations)
```

```
## Error:  object 'pl' not found
```

```
pl$relations$refs[c(1:3, 745:754),]

## Error:  object 'pl' not found
```

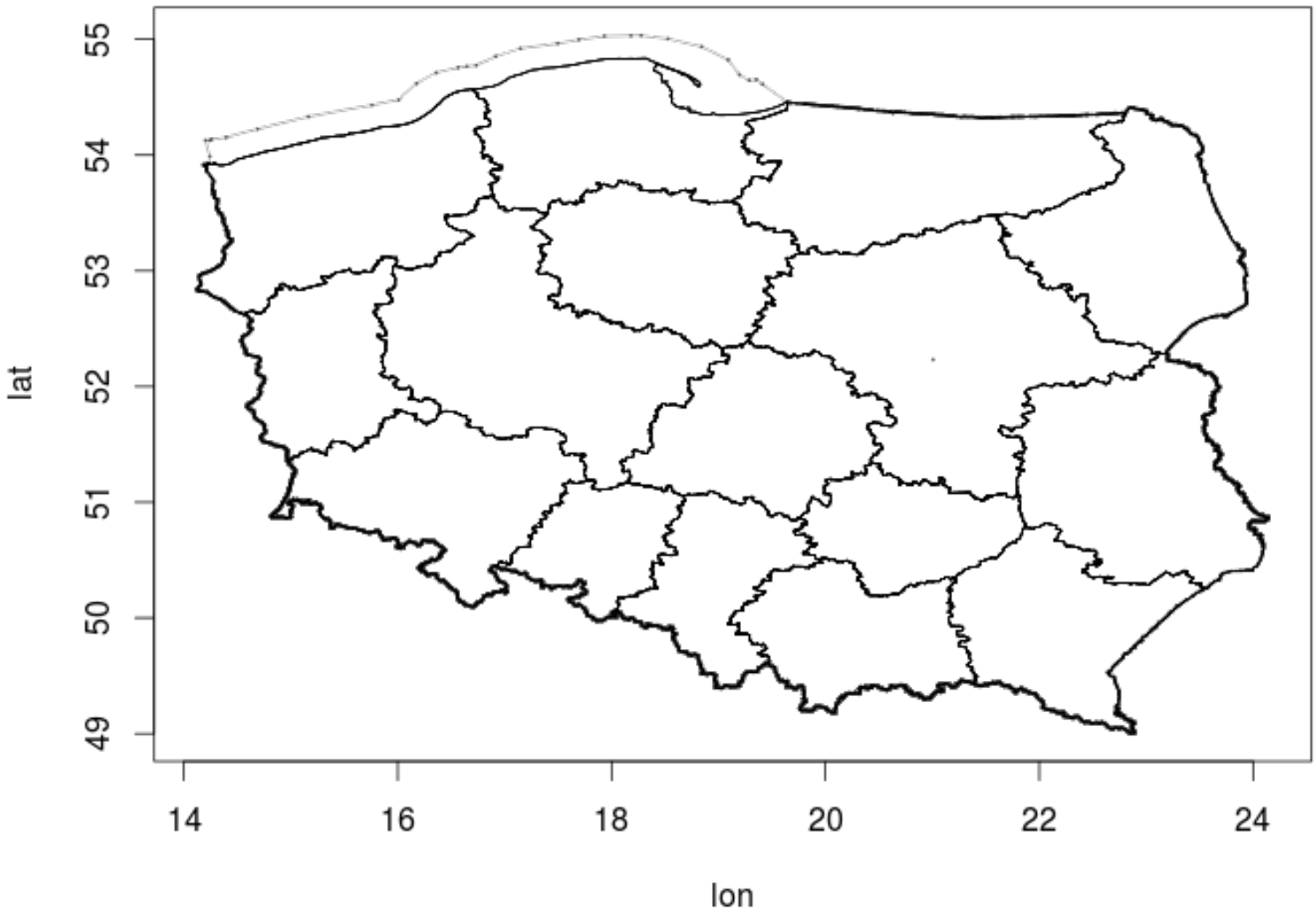

OSM
nodes
relations
ways
attrs
refs
tags
nodes
relations
ways

```
pl <- get_osm(relation(49715), osmsource_api())
```

```
pl <- get_osm(relation(49715), osmsource_api(), full=T)
```

```
pl <- get_osm(relation(49715), osmsource_api())
```

```
pl <- get_osm(relation(49715), osmsource_api(), full=T)
```


lat
55
54
53
52
51
50
49
14
16
18
20
22
24
lon

```
pl <- get_osm(relation(49715), osmsource_api(), full=T)


plot(pl)


subareas <- pl$relations$refs$ref[
                pl$relations$refs$role=="subarea"]


for (i in subareas) {
   tmp <- get_osm(relation(i), osmsource_api(), full=TRUE)
   tmpsp <- as_sp(tmp, what="lines")
   lines(tmpsp)
}
```

```r
pl <- get_osm(relation(49715), osmsource_api(), full=T)
```

```r
plot(pl)
```

```r
subareas <- pl$relations$refs$ref[
                pl$relations$refs$role=="subarea"]
```

```r
for (i in subareas) {
   tmp <- get_osm(relation(i), osmsource_api(), full=TRUE)
   tmpsp <- as_sp(tmp, what="lines")
   lines(tmpsp)
}
```

```
pl <- get_osm(relation(49715), osmsource_api(), full=T)
```

```
plot(pl)
```

```
subareas <- pl$relations$refs$ref[
                    pl$relations$refs$role=="subarea"]
```

```
for (i in subareas) {
   tmp <- get_osm(relation(i), osmsource_api(), full=TRUE)
   tmpsp <- as_sp(tmp, what="lines")
   lines(tmpsp)
}
```

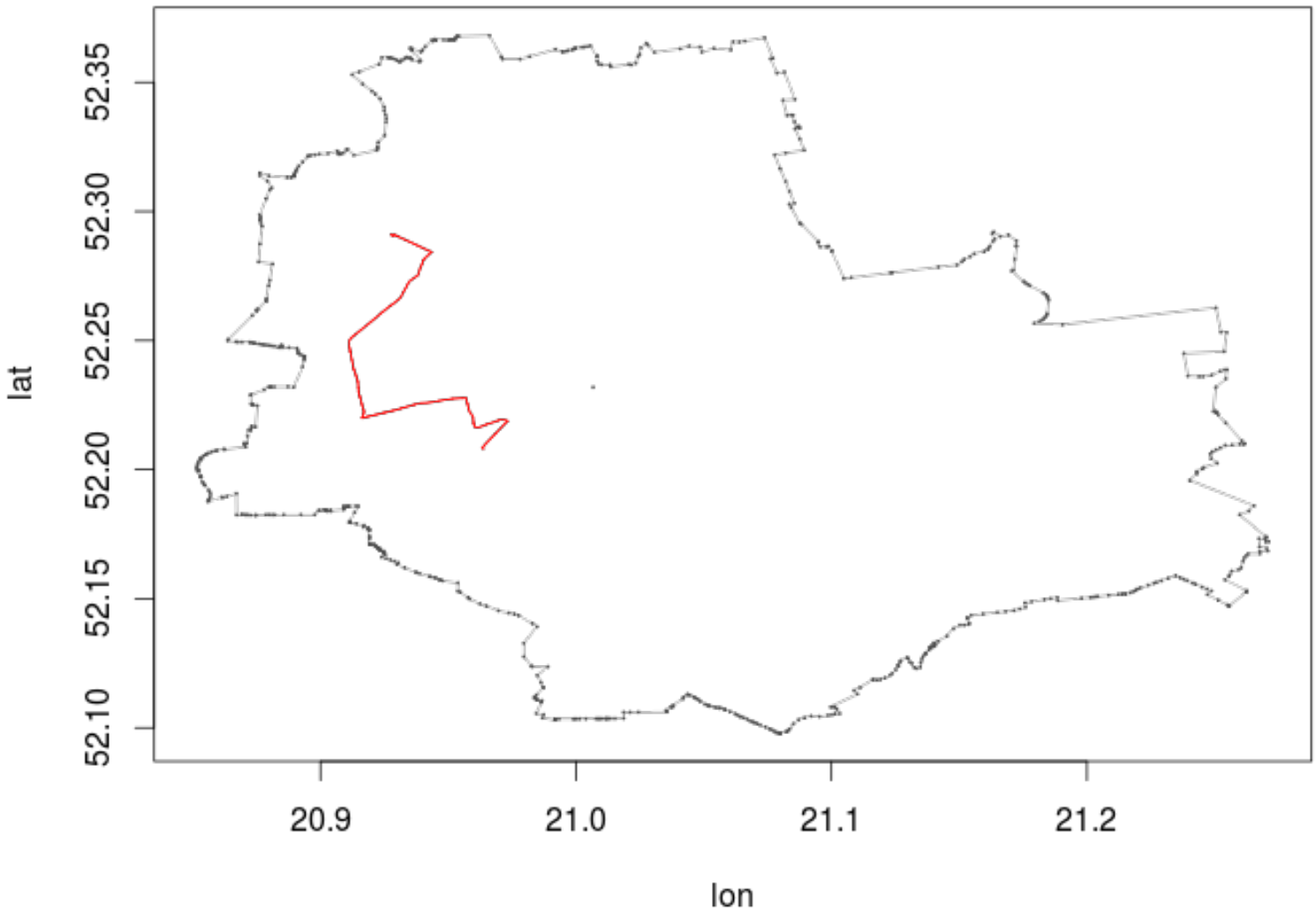
```
pl <- get_osm(relation(49715), osmsource_api(), full=T)
```

```
plot(pl)
```

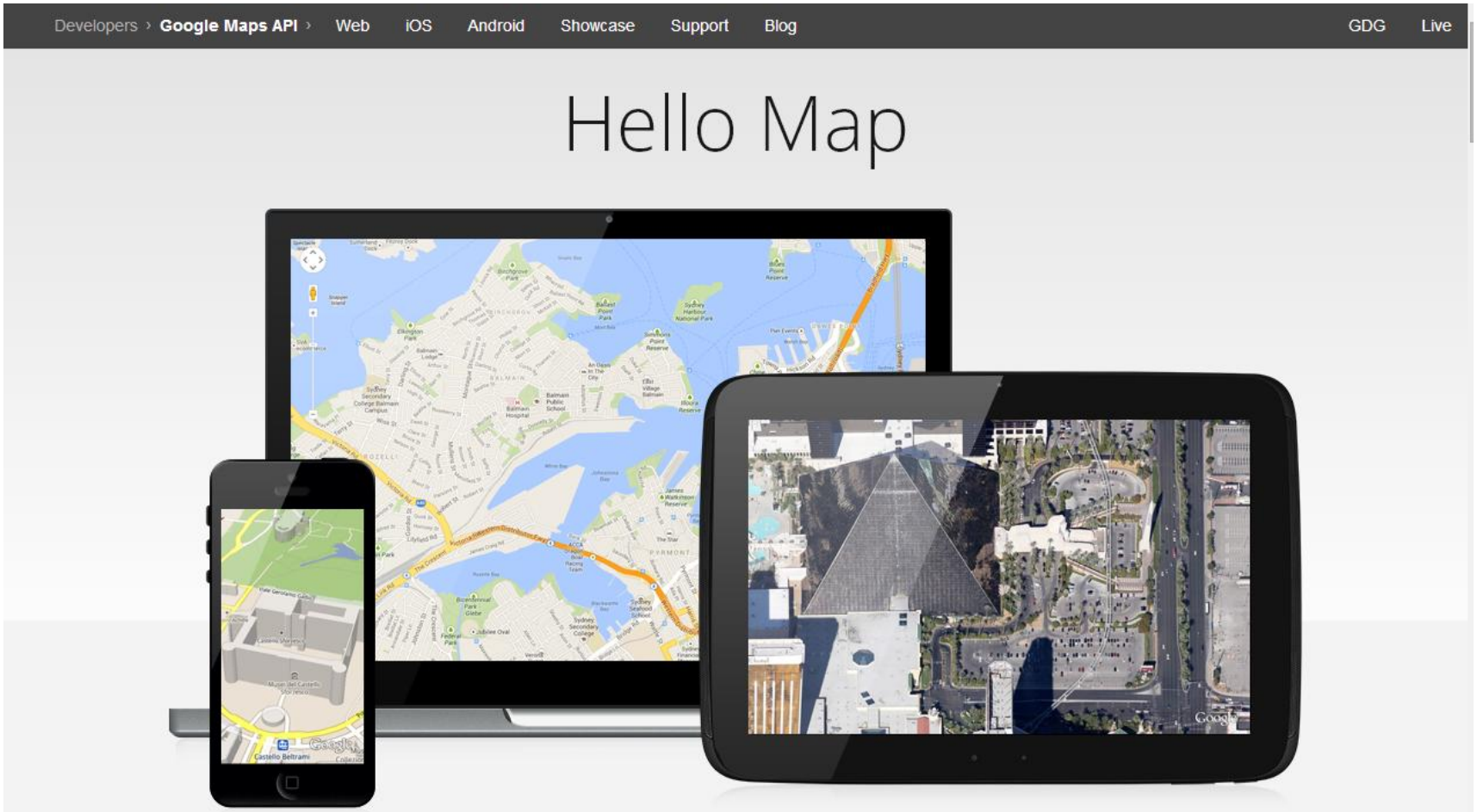
```
subareas <- pl$relations$refs$ref[
                    pl$relations$refs$role=="subarea"]
```

```
for (i in subareas) {
   tmp <- get_osm(relation(i), osmsource_api(), full=TRUE)
   tmpsp <- as_sp(tmp, what="lines")
   lines(tmpsp)
}
```

```
warsaw <- get_osm(relation(336074), osmsource_api(), full=T)
plot(warsaw)
l_184 <- get_osm(relation(3073087), osmsource_api(), full=T)
l_sp <- as_sp(l_184, what="lines")
lines(l_sp, col="red")
```


lat
52.35
52.30
52.25
52.20
52.15
52.10
20.9
21.0
21.1
21.2
lon

plotGoogleMaps



plotGoogleMaps

The plotGoogleMaps package:

- provides an interactive plot device for handling the geographic data within web browsers
- is designed for the automatic creation of web maps as a combination of users' data and Google Maps layers
- uses web browser as plotting device instead of default R graphic device.

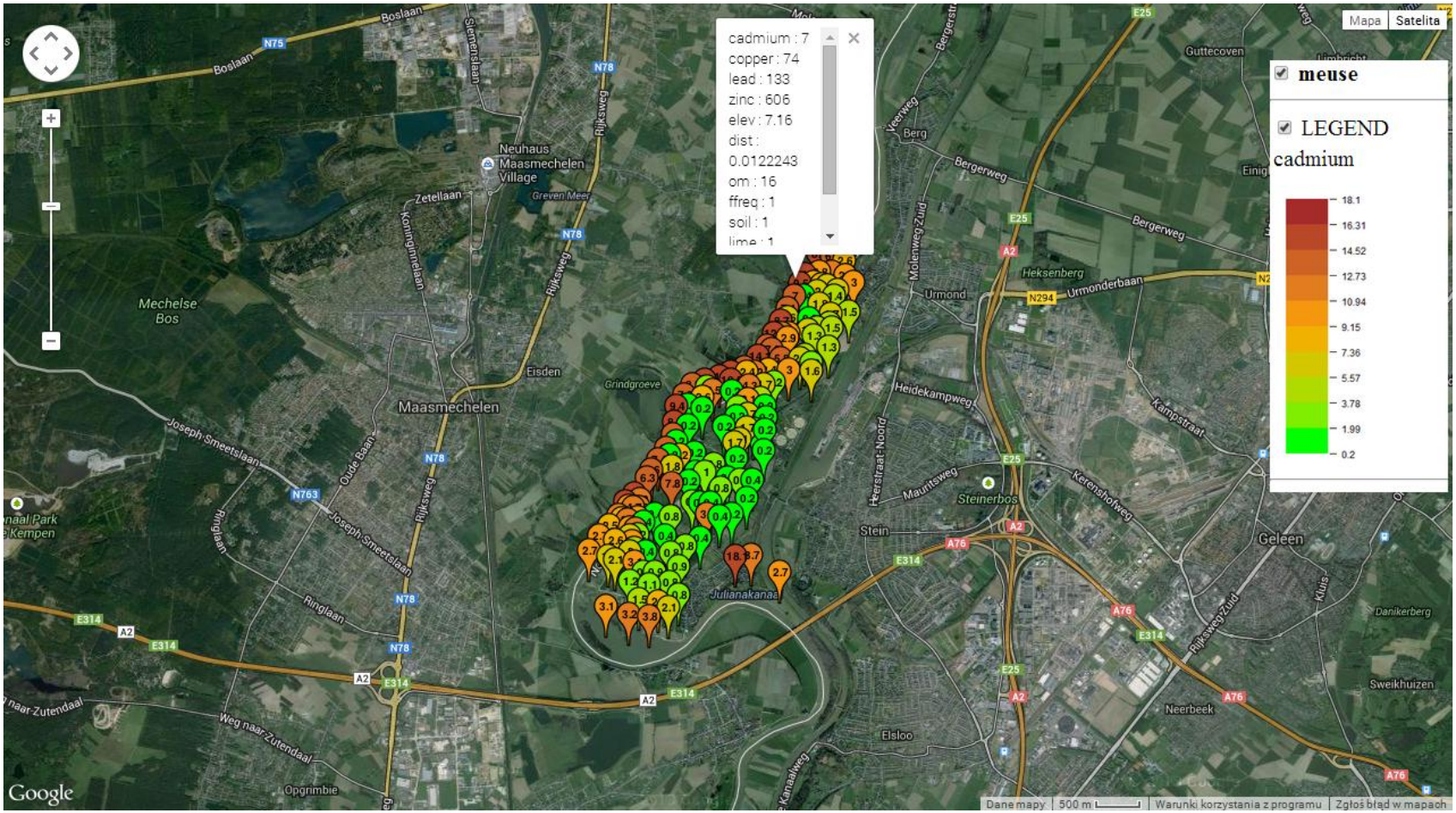
```
library(plotGoogleMaps)
data(meuse)
head(meuse[,1:8])

##        x      y cadmium copper lead zinc  elev     dist
## 1 181072 333611    11.7     85  299 1022 7.909 0.001358
## 2 181025 333558     8.6     81  277 1141 6.983 0.012224
## 3 181165 333537     6.5     68  199  640 7.800 0.103029
## 4 181298 333484     2.6     81  116  257 7.655 0.190094
## 5 181307 333330     2.8     48  117  269 7.480 0.277090
## 6 181390 333260     3.0     61  137  281 7.791 0.364067
```

```r
coordinates(meuse) <- ~x+y # convert to SPDF
# Class for spatial attributes that
# have spatial point locations
typeof(meuse)

## [1] "S4"

class(meuse)

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

proj4string(meuse) <- CRS('+init=epsg:28992')
# adding Coordinate Referent Sys.
```

```r
m <- plotGoogleMaps(meuse,filename='myMap1.htm')
```

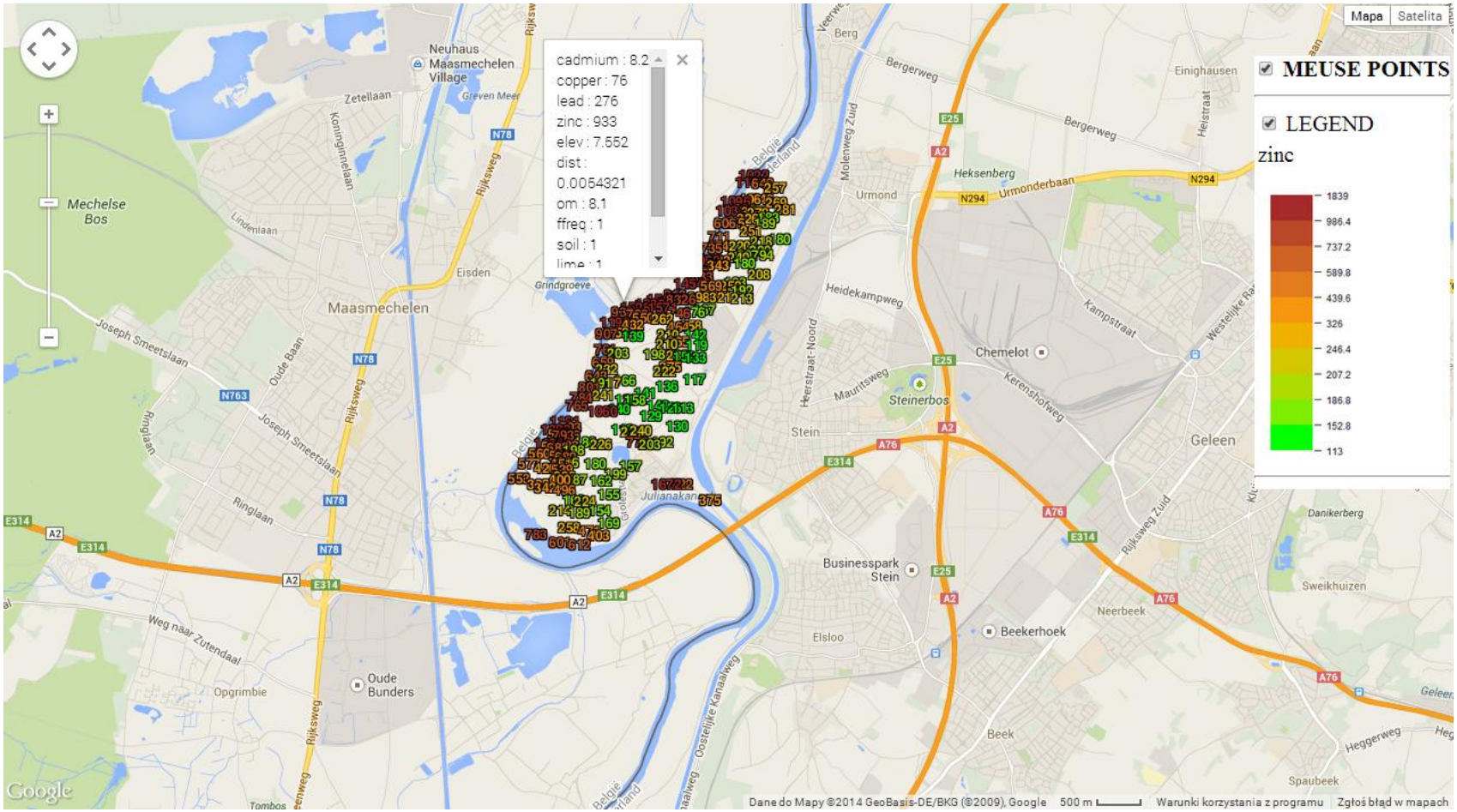
S4 is the 4th version of S. S is a language that has two implementation: S-plus is commercial, R is free. The main characteristic of S4 compared to S3 is the development of functions which allow to consider S as an object language. By extension, S4 stand for object oriented programming with S. And thus with R or S-plus. More at: A (Not So) Short Introduction to S4

plotGoogleMaps



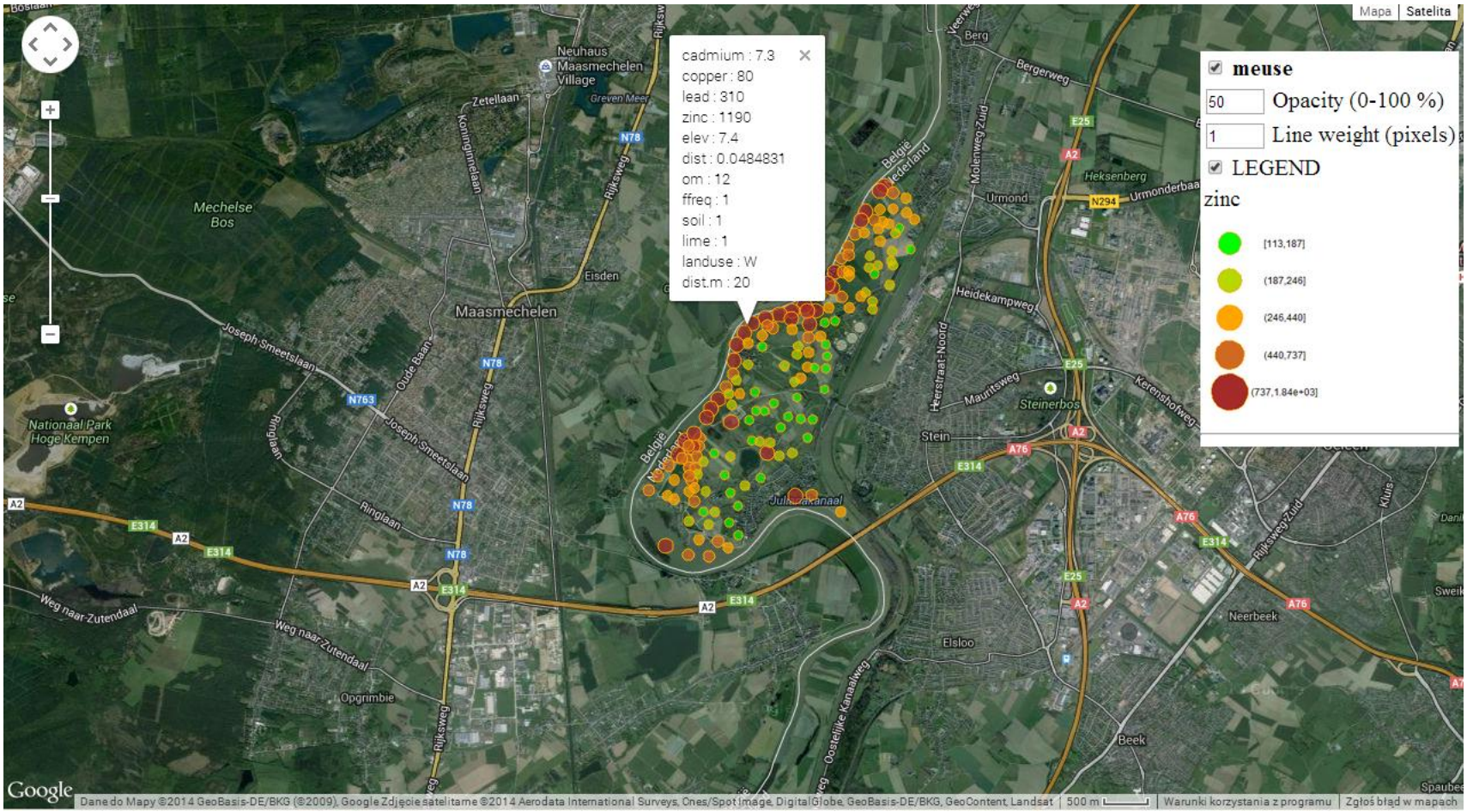
```
ic <- iconlabels(meuse$zinc, height=12)
m <- plotGoogleMaps(meuse,zcol='zinc',
      filename='myMap_z2.htm',
      iconMarker=ic, mapTypeId='ROADMAP',
      layerName = 'MEUSE POINTS')
# zcol - the column we want to label
```

plotGoogleMaps



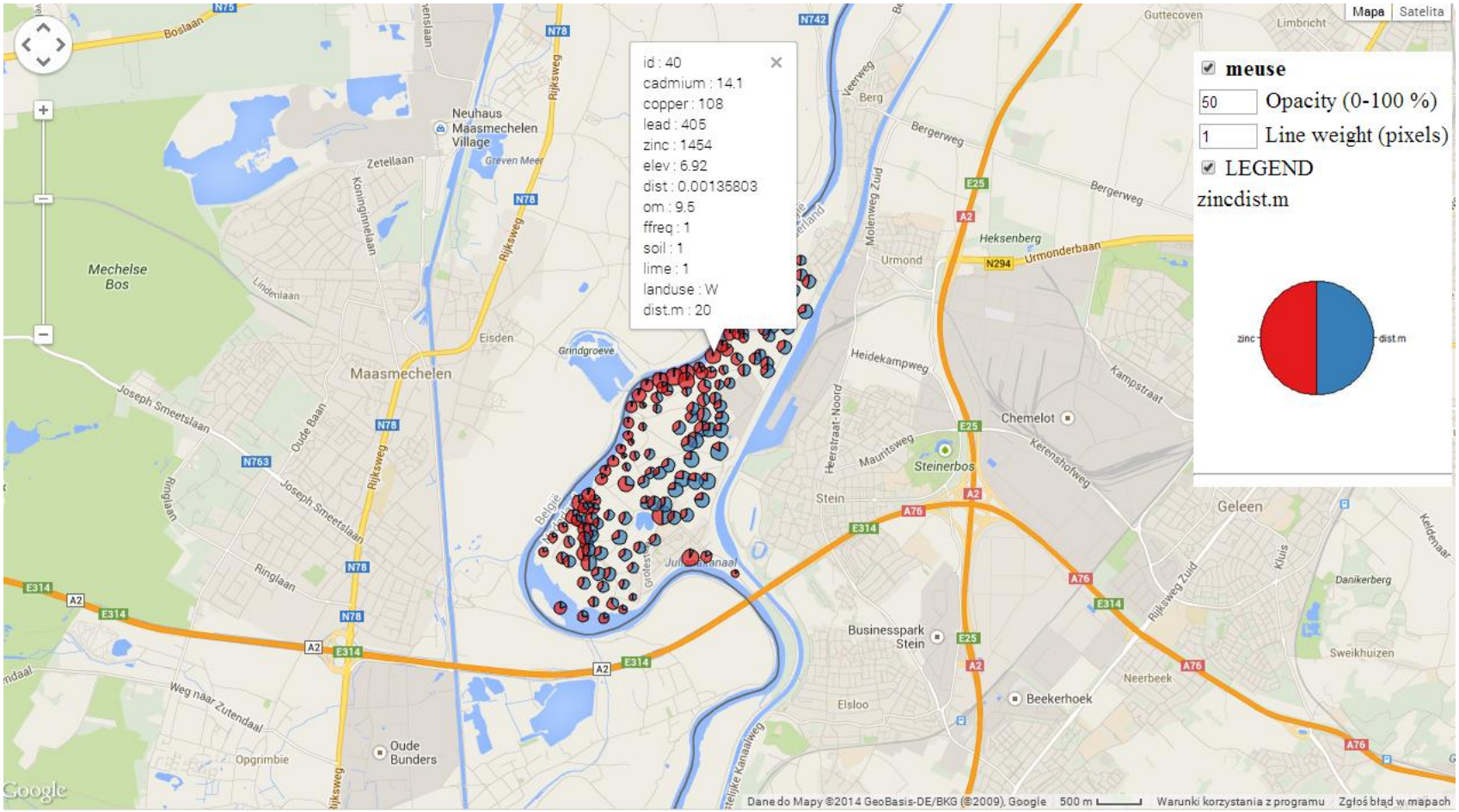
```
m <- bubbleGoogleMaps(meuse,zcol='zinc',max.radius = 80,
                      filename='myMap3.htm',strokeOpacity=0)
```

plotGoogleMaps

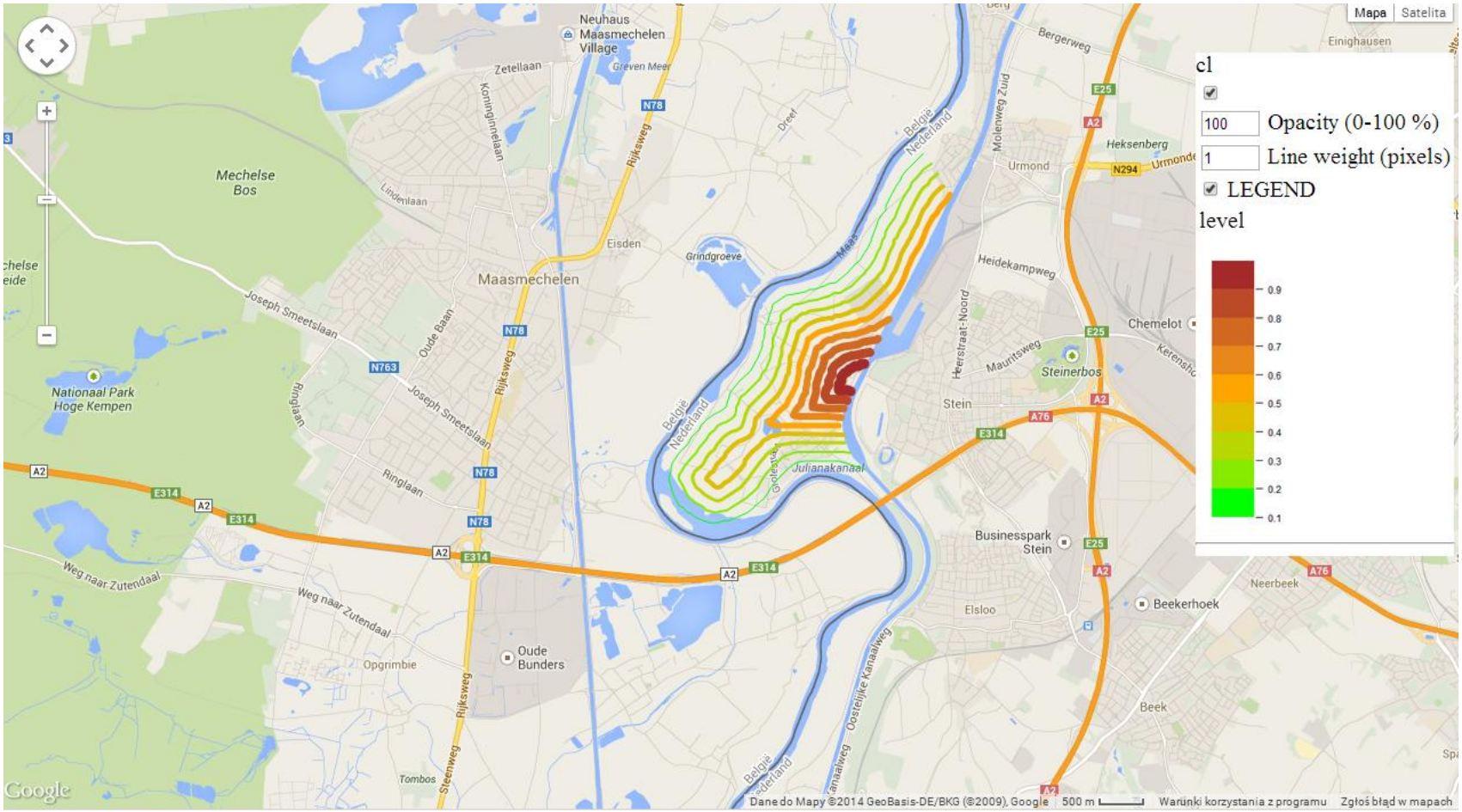


```
m <- segmentGoogleMaps(meuse, zcol=c('zinc','dist.m'),
      mapTypeId='ROADMAP', filename='myMap4.htm',
      colPalette=c('#E41A1C','#377EB8'), strokeColor='black')
```

plotGoogleMaps

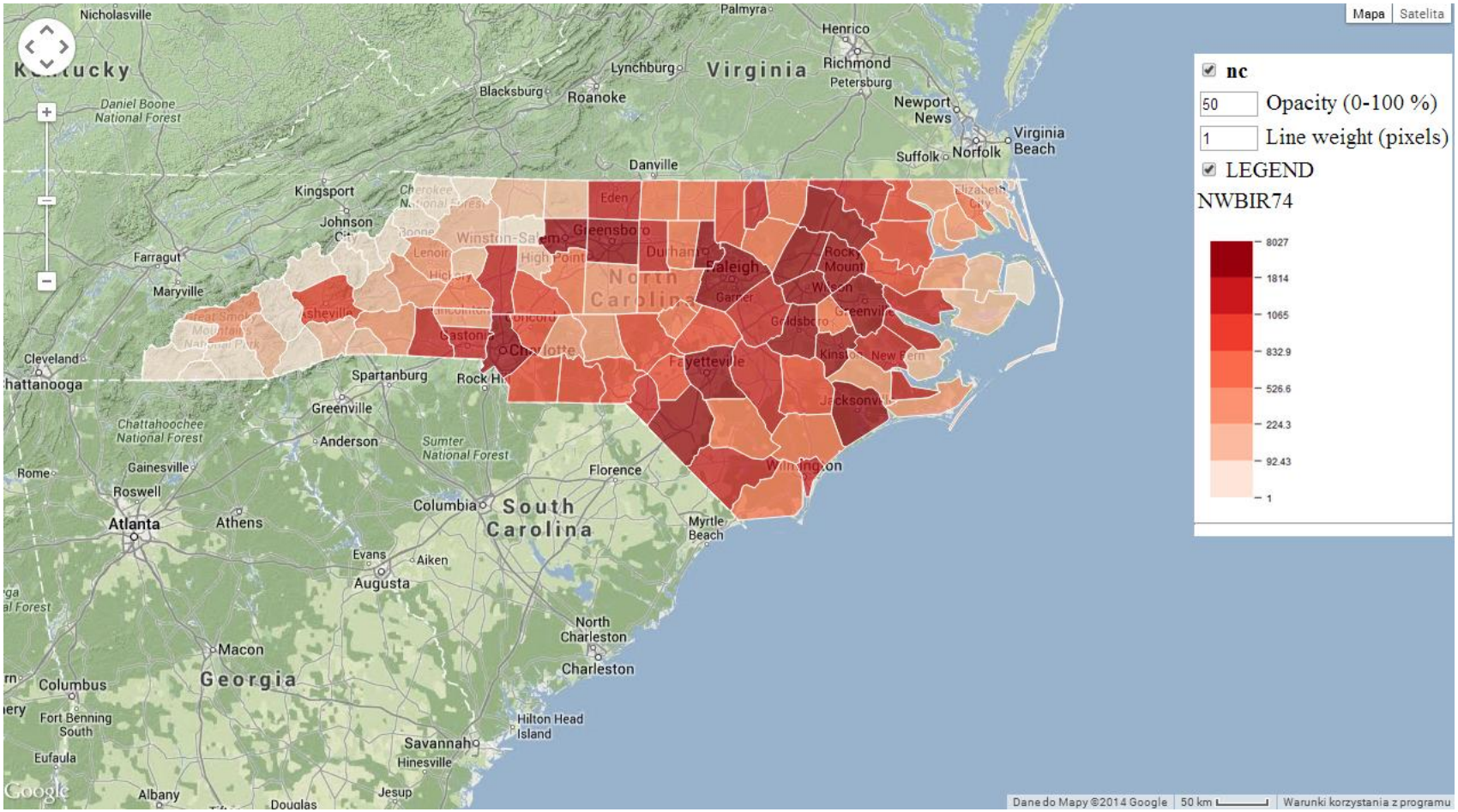


plotGoogleMaps



```
nc <- readShapeSpatial(system.file("shapes/sids.shp",
            package="maptools")[1],
            proj4string=CRS("+proj=longlat +datum=NAD27"))
#install.packages("RColorBrewer")
library(RColorBrewer)
m <- plotGoogleMaps(nc,zcol="NWBIR74",filename='MyMap6.htm',
      mapTypeId='TERRAIN',colPalette= brewer.pal(7,"Reds"),
      strokeColor="white")
```

plotGoogleMaps



```
m1 <- plotGoogleMaps(cl,zcol='level',
                     strokeWeight=1:9 ,
                     colPalette='grey',
                     add=TRUE)
m2 <- bubbleGoogleMaps(meuse,zcol='zinc',
                       colPalette= brewer.pal(5,"Accent"),
                       max.radius = 80,
                       previousMap= m1,
                       filename='comb.html')
```

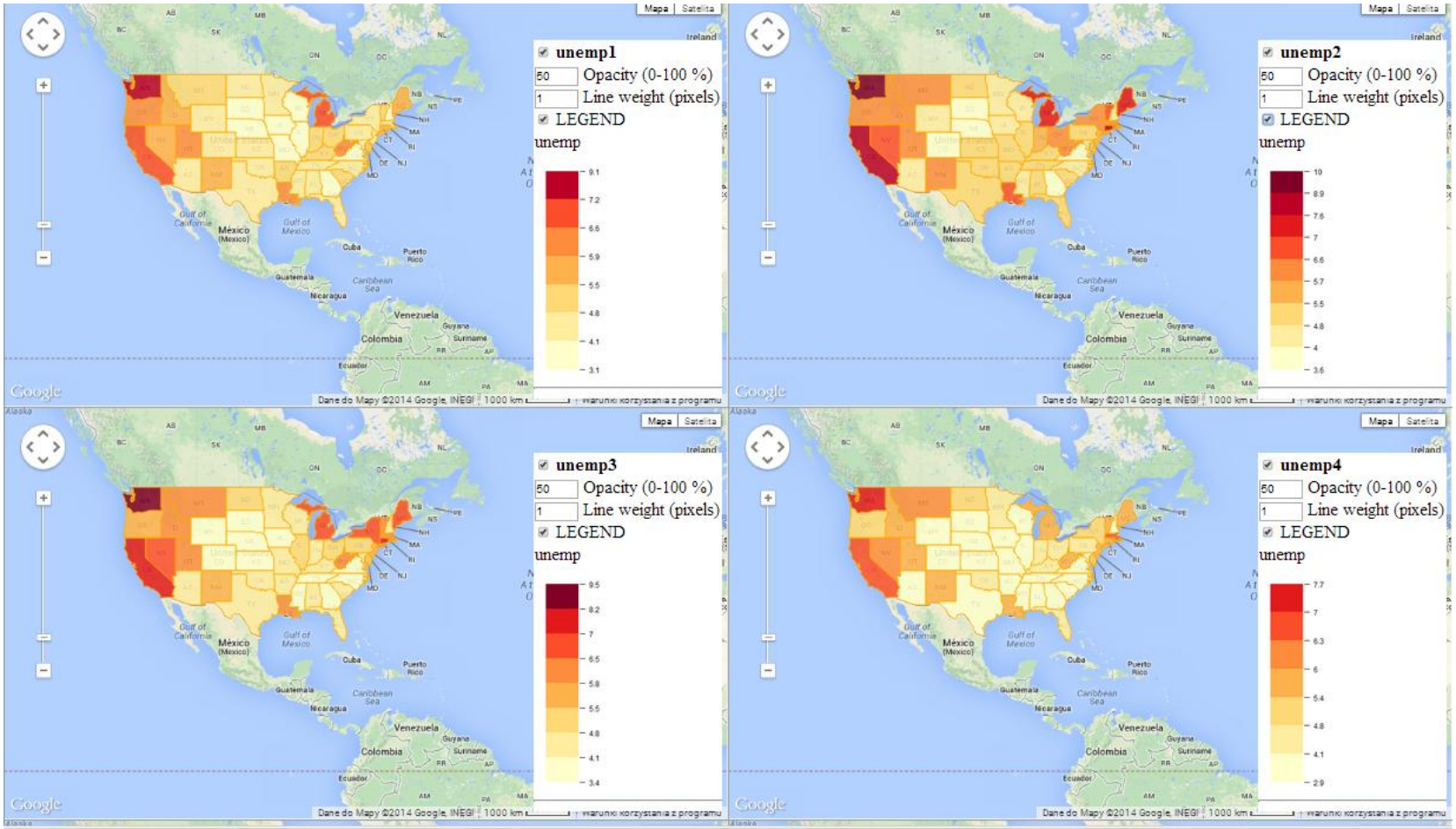
plotGoogleMaps

```
library("spacetime")
library("maps")
library("maptools")
library("plm")
states.m <- map('state',plot=FALSE,fill=TRUE)
IDs <- sapply(strsplit(states.m$names, ":"),function(x) x[1])
states <- map2SpatialPolygons(states.m,IDs=IDs)
yrs <- 1970:1986
time <- as.POSIXct(paste(yrs, "-01-01", sep=""),tz = "GMT")
data("Produc")
Produc.st = STFDF(states[-8], time,
                  Produc[order(Produc[2], Produc[1]),])
```

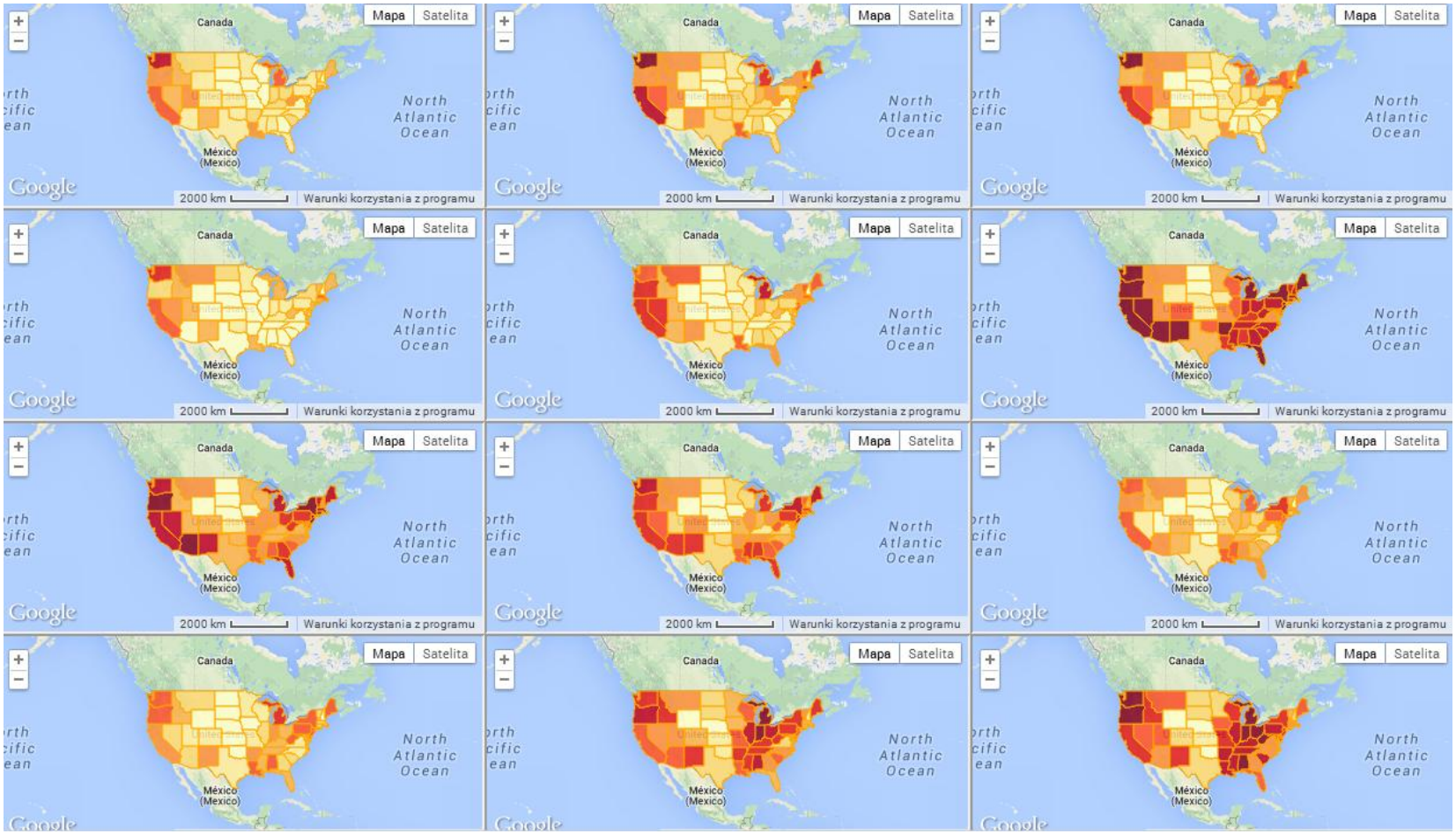
STFDF – spatio-temporal data for which each location has data for each time.

```
Produc.st@sp@proj4string=CRS('+proj=longlat +datum=WGS84')
library(RColorBrewer)
ee <- stplotGoogleMaps(Produc.st,zcol='unemp',
          stfilename='USA.htm',colPalette=brewer.pal(9, "YlOrRd"
          mapTypeId='ROADMAP',w='49%',h='49%', fillOpacity=0.85)
# without control
ee <- stplotGoogleMaps(Produc.st,zcol='unemp',
          stfilename='USA2.htm',colPalette=brewer.pal(9, "YlOrRd
          mapTypeId='ROADMAP',w='33%',h='25%',
          fillOpacity=0.85, control.width=0)
```

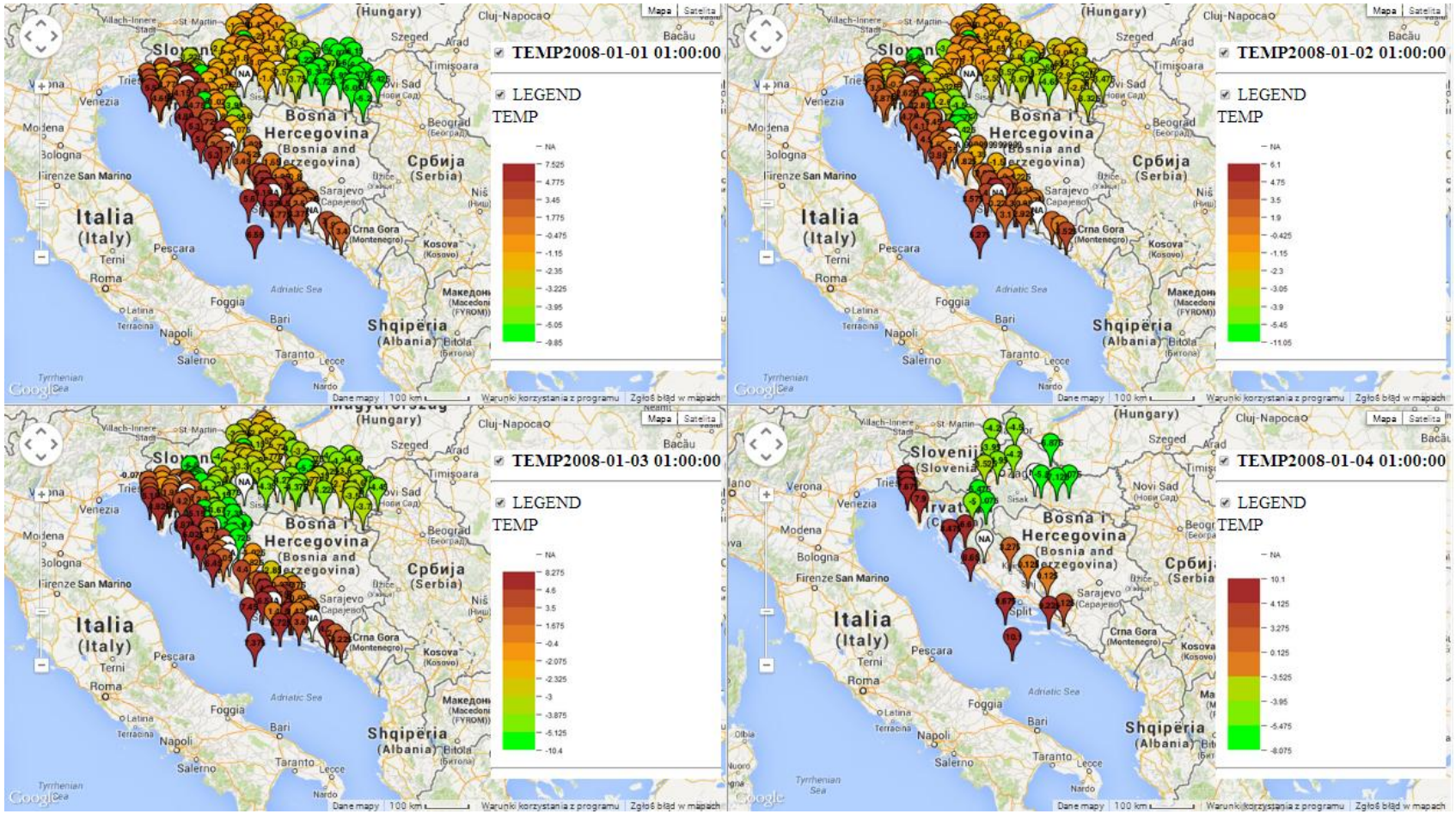
plotGoogleMaps



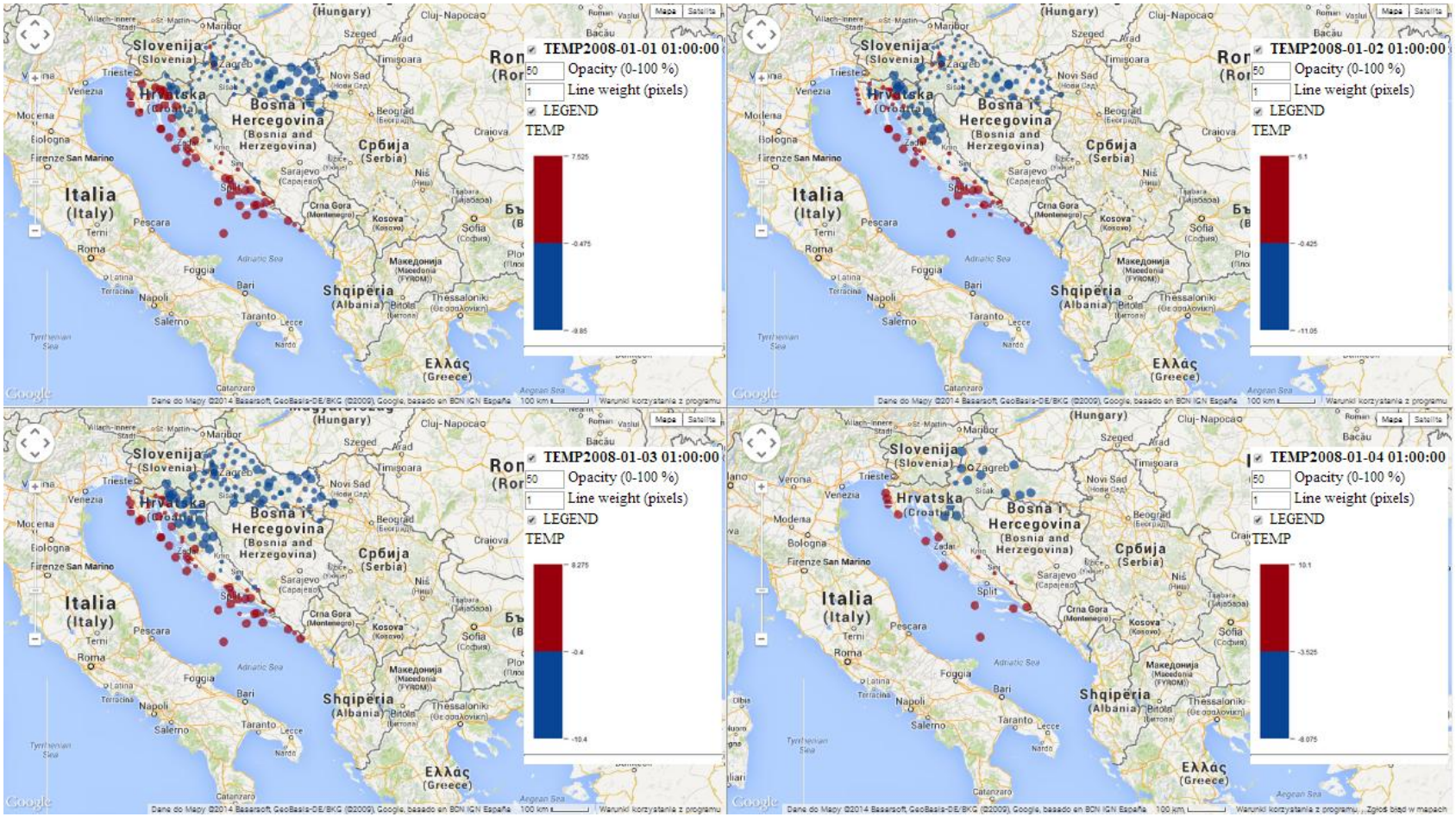
plotGoogleMaps



plotGoogleMaps



plotGoogleMaps



Bibliografia

 A plotGoogleMaps tutorial

 A (Not So) Short Introduction to S4

 spacetime: Spatio-Temporal Data in R

Thank you.