13: Map Overlays and Spatial Modeling

John H Maindonald

September 6, 2015

```
doFigs <- TRUE
```

Start by attaching required packages:

```
figset13 <- function(){
  if(!requireNamespace('DAAG', quietly = TRUE))stop('DAAG must be installed')
  if(!require('latticeExtra', quietly = TRUE))stop('latticeExtra must be installed')
  if(!requireNamespace('oz', quietly = TRUE))stop('oz must be installed')
  if(!requireNamespace('rgdal', quietly=TRUE))stop('rgdal must be installed')
  if(!require('sp', quietly = TRUE))stop('sp must be installed')
  }
```

```
figset13()
```

```
fig13.1 <- function(){
## ---- oz-sites ----
oz::oz(sections=c(3:5, 11:16), col="gray")
chh <- par()$cxy[2]
with(DAAG::possumsites, {
  points(Longitude,
         Latitude+c(0,0,0,.2,-.2, 0,0)*chh,
         col="blue")
  text(Latitude ~ Longitude,
       labels=rownames(DAAG::possumsites),
       col="red", pos=c(2,4,2,1,3,2,2), xpd=TRUE)
  # pos = 1:below, 2:left, 3:above, 4:right
  # xpd=TRUE allows plotting outside figure region
})
}
```

```
opar <- par(mar=c(4,4,1.6,3.1))
fig13.1()
par(opar)
```

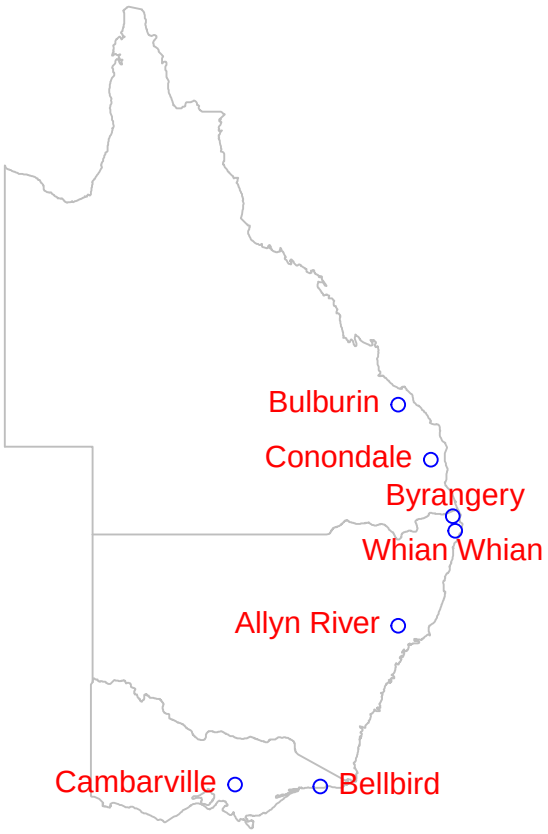
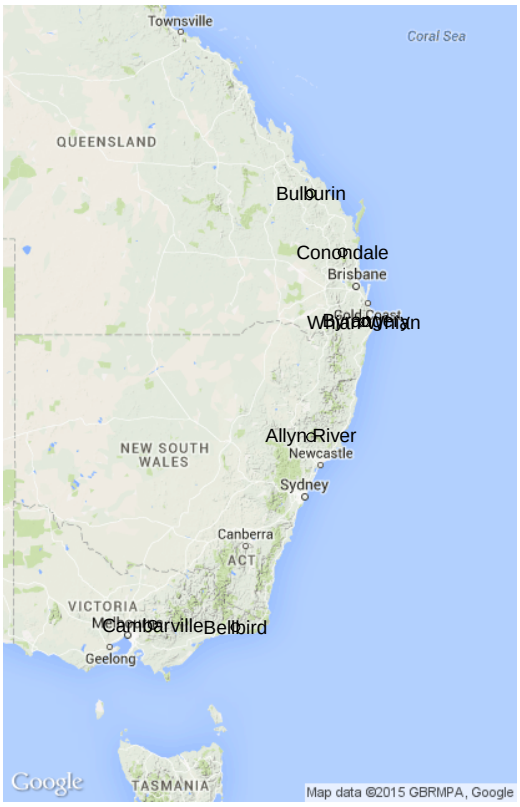


Figure 1: Sites at which possums were collected.

```
supp13.1()
```




```
supp13.1 <- function(){
if(!require(dismo))stop('dismo must be installed')
## ---- google-possums ----
## Extend longitude & latitude ranges slightly
lonlat <- with(DAAG::possumsites,
               c(range(Longitude)+c(-3,3),
                 range(Latitude)+c(-2,2))
)
## Obtain map, as a ``RasterLayer'' object
googmap <- gmap(extent(lonlat))
plot(googmap, inter=TRUE)
## From latitude/longitude to Mercator projection
xy <- Mercator(with(DAAG::possumsites,
                    cbind(Longitude, Latitude)))
## Points show location of sites on the map
points(xy)
## Add labels that give the names
text(xy, labels=row.names(DAAG::possumsites))
}
```

```
supp13.2 <- function(){
    if(!require(plotKML))stop("plotKML must be installed.")
## ---- plotKML ----
plotKML(quakes['Energy'], points_names="")
}
```

```
## Input data from internet
from <-
  paste(c("http://wfs-beta.geonet.org.nz/",
          "geoserver/geonet/ows?service=WFS",
          "&version=1.0.0",
          "&request=GetFeature",
          "&typeName=geonet:quake",
          "&outputFormat=csv",
          "&cql_filter=origintime%3E='2009-08-01'",
          "+AND+magnitude>4.5"),
        collapse="")
quakes <- read.csv(from)
z <- strsplit(as.character(quakes$origintime),
              split="T")
quakes$Date <- sapply(z, function(x)as.Date(x[1]))
quakes$Time <- sapply(z, function(x)x[2])
quakes$Energy <- 10^quakes$magnitude/1000000
  # Will make circles area proportional to Energy
```

```
## Prepare data for plotting
coordinates(quakes) <- ~ longitude+latitude
proj4string(quakes) <-
  CRS("+proj=longlat +datum=WGS84")
```

```
## ---- logofile ----
logofile <- system.file("pictures/Rlogo.jpg",
                        package = "rgdal")[1]
rlogo <- rgdal::readGDAL(logofile, silent=TRUE)

Warning in rgdal::readGDAL(logofile, silent = TRUE): GeoTransform values not
available

fig13.2 <- function(){
## ---- rlogo-image ----
image(rlogo, red="band1",
      green="band2",
      blue="band3")
}
```

```
## Input data from internet
from <-
  paste(c("http://wfs-beta.geonet.org.nz/",
          "geoserver/geonet/ows?service=WFS",
          "&version=1.0.0",
          "&request=GetFeature",
          "&typeName=geonet:quake",
          "&outputFormat=csv",
          "&cql_filter=origintime%3E='2009-08-01'",
          "+AND+magnitude>4.5"),
        collapse="")
quakes <- read.csv(from)
z <- strsplit(as.character(quakes$origintime),
              split="T")
quakes$Date <- sapply(z, function(x)as.Date(x[1]))
quakes$Time <- sapply(z, function(x)x[2])
quakes$Energy <- 10^quakes$magnitude/1000000
  # Will make circles area proportional to Energy
## Prepare data for plotting
coordinates(quakes) <- ~ longitude+latitude
proj4string(quakes) <-
  CRS("+proj=longlat +datum=WGS84")
supp13.2()
```

```
fig13.2()
```

Figure 2: The function `image()` has been used to display the R logo image that had been input as a GDAL grid map.

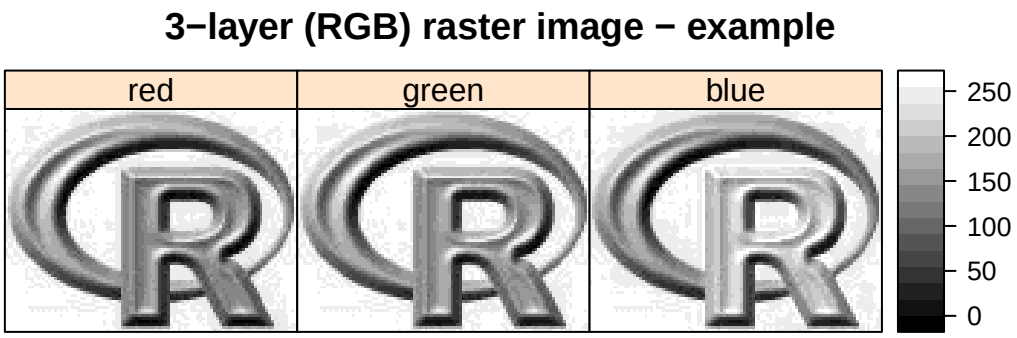


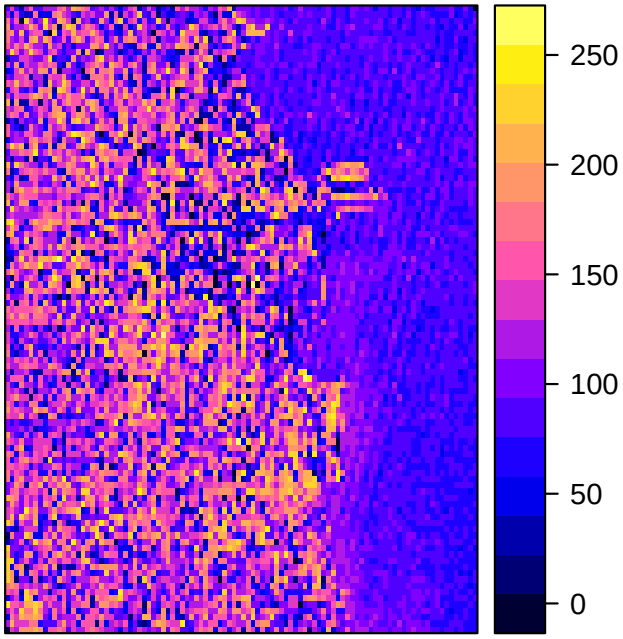
Figure 3: Red, green and blue layers from the R logo image.

```
fig13.3 <- function(){
## ---- spplot-col3 ----
## Code
col3 <- c("red","green","blue")
spplot(rlogo, zcol=1:3, names.attr=col3,
       col.regions=grey(0:100/100), as.table=TRUE,
       layout=c(3,1), main=paste("3-layer (RGB)",
       "raster image - example"))
}
```

```
supp13.3 <- function(){
## ---- sp27 ----
sp27 <- system.file("pictures/SP27GTIF.TIF",
                    package = "rgdal")[1]
## ---- sp27-read ----
SP27GTIF <- rgdal::readGDAL(sp27, output.dim=c(100,100),
                     silent=TRUE)
class(SP27GTIF)
## ---- sp27-spplot ----
spplot(SP27GTIF)
}
```

```
fig13.4 <- function(){
## ---- meuse-bubble ----
data(meuse); data(meuse.riv)
coordinates(meuse) <- ~ x + y
gph <- bubble(meuse, "zinc", pch=1,
              key.entries =  100 * 2^(0:4),
              main = "Zinc(ppm)",
              scales=list(axes=TRUE, tck=0.4))
add <-
```

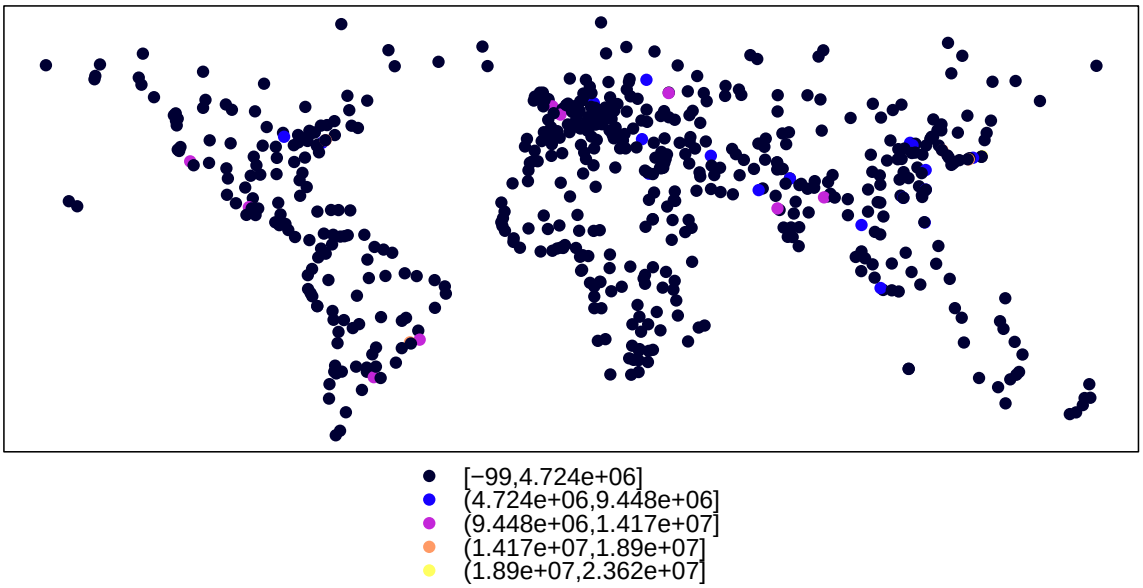
```
supp13.3()
```




```
  latticeExtra::layer(panel.lines(meuse.riv[,1],
                                  meuse.riv[,2],
                        col="gray"))
gph+add
}
```

```
supp13.4 <- function(){
## ---- show-shape ----
dsn <- system.file("vectors", package = "rgdal")[1]
## ---- shape-in ----
cities <- rgdal::readOGR(dsn=dsn, layer="cities", verbose=FALSE)
## ---- canada-plot ----
canada <- subset(cities, COUNTRY=="Canada")
trellis.par.set(sp.theme())
spplot(cities, zcol="POPULATION")
}
```

```
supp13.4()
```



Supplementary Figure 1: Plot derived from shapefile information.

```
fig13.4()
```

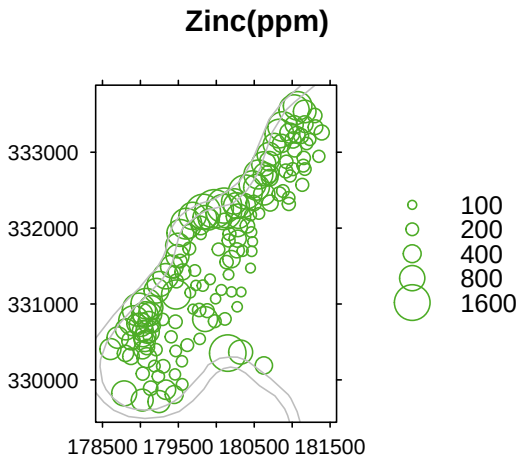


Figure 4: Bubble plot for `zinc`, with area of bubbles proportional to concentration. River Meuse boundaries are in gray.