# The mapmisc package

Patrick Brown

October 31, 2026

This document will be incomplete if `rgdal` is unavailable or there is on internet connection when this document is compiled. The full document is at `http://diseasemapping.r-forge.r-project.org/`.

## 1 Introduction

This package provides a few utilities for making nice maps in R, with an emphasis on enabling maps in short tidy code chunks which are suitable for Sweave and knitr documents. The package duplicates the capabilities of packages such as classInt, geonames, and OpenStreetMap, and the price of having tidier code is much of the flexibility from these other packages has been lost here.

The meuse data from the sp package

```
library('sp')
data('meuse')
coordinates(meuse) <- ~x+y
class(meuse)

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

```
library('rgdal')
proj4string(meuse) <- CRS("+init=epsg:28992")
meuseLL = spTransform(meuse, CRS("+init=epsg:4326"))
```

Get elevation data with

```
library('raster')
nldElev = raster::getData("alt", lon=raster::xmin(meuseLL),
    lat=raster::ymin(meuseLL), path=tempdir() )
raster::getData("SRTM", lon=raster::xmin(meuseLL),
```

```
    lat=raster::ymin(meuseLL), path=tempdir() )
nldElev = raster(file.path(tempdir(), "srtm_38_02.tif"))
nldElev = crop(nldElev, extent(meuseLL))
nldElev = projectRaster(nldElev, crs=proj4string(meuse))
```

Or simply do

```
library('mapmisc')
data('netherlands')
```

The elevation data is a Raster.

```
class(nldElev)

## [1] "RasterLayer"
## attr(,"package")
## [1] "raster"

nldElev = crop(nldElev, extend(extent(meuse), 1000))
```

# 2 Downloading background maps and city locations

Get a background map covering the extent of the meuse data

```
library('mapmisc')
if(haveRgdal)
  nldTiles = openmap(meuse)
```

`nldTiles` is a Raster with the same projection as `meuse`

```
class(nldTiles)

## [1] "RasterLayer"
## attr(,"package")
## [1] "raster"

projection(nldTiles)

## [1] "+proj=sterea +lat_0=52.1561605555556 +lon_0=5.38763888888889 +k=0.9999079 +x_0=1

projection(meuse)

## [1] "+proj=sterea +lat_0=52.15616055555555 +lon_0=5.38763888888889 +k=0.9999079 +x_0=
```

Maps which can be downloaded are shown at `http://diseasemapping.r-forge.r-project.org/openmap/`

`GNcities` is a wrapper for the function of the same name in the geonames package.

```
nldCities = GNcities(meuse, maxRows=6)
```

A SpatialPointsDataFrame, with same map projection.

```
class(nldCities)

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

names(nldCities)

##  [1] "lng"         "geonameId"   "countrycode" "name"        "fclName"
##  [6] "toponymName" "fcodeName"   "wikipedia"   "lat"         "fcl"
## [11] "population"  "fcode"

projection(nldCities)

## [1] "+proj=sterea +lat_0=52.1561605555556 +lon_0=5.38763888888889 +k=0.9999079 +x_0=1
```

# 3 Making maps

The `map.new` function sets up a map in the current plot window with the correct limits and aspect ratio for the object supplied, and without margins or white space. `scaleBar` adds a scale and north arrow. It uses the map projection of the argument supplied to calculate distances and find north.

Some of the code in the following sections require the `rgdal` package to run.

```
# plot the data locations
map.new(meuse)
plot(nldTiles, add=TRUE)
points(meuse,col="red", cex=0.3)
scaleBar(crs=proj4string(meuse),pos="topleft", bg="white")

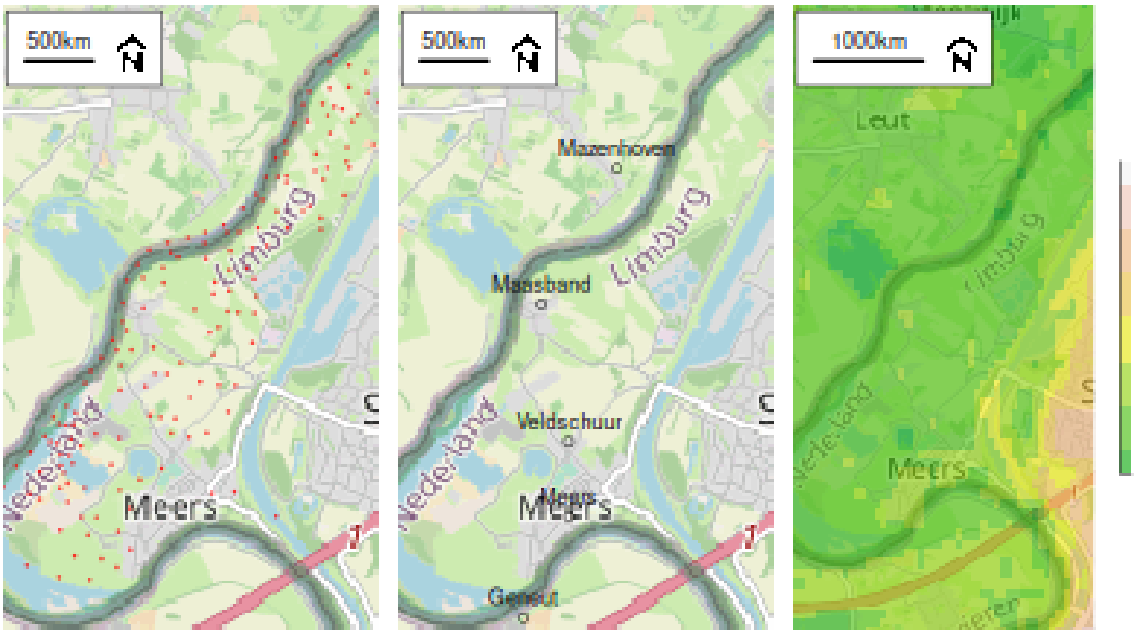
## Warning in scaleBar(crs = proj4string(meuse), pos = "topleft", bg = "white"):
rgdal not intalled, assuming coordinates are either long-lat or metres, and north
is up
## Warning in scaleBar(crs = proj4string(meuse), pos = "topleft", bg = "white"):
rgdal not intalled, assuming the plot is long-lat
```

```
# plot city names
map.new(meuse)
plot(nldTiles, add=TRUE)
points(nldCities)
text(nldCities, labels=nldCities$name, pos=3)
scaleBar(proj4string(meuse),pos="topleft", bg="white")
```

```
## Warning in scaleBar(proj4string(meuse), pos = "topleft", bg = "white"): rgdal
not intalled, assuming coordinates are either long-lat or metres, and north is
up
## Warning in scaleBar(proj4string(meuse), pos = "topleft", bg = "white"): rgdal
not intalled, assuming the plot is long-lat
```

```
# plot elevation
map.new(meuse, legendRight=TRUE)
plot(nldTiles, add=TRUE)
plot(nldElev,add=TRUE,col=terrain.colors(8),alpha=0.6,legend.mar=2, legend.line=0,)
scaleBar(meuse,pos="topleft",bg="white")
```

```
## Warning in scaleBar(meuse, pos = "topleft", bg = "white"): rgdal not intalled,
assuming coordinates are either long-lat or metres, and north is up
## Warning in scaleBar(meuse, pos = "topleft", bg = "white"): rgdal not intalled,
assuming the plot is long-lat
```




(a) data locations (b) cities (c) elevation

Figure 1: simple map

# 4 Legends

Create a colour scale for plotting copper concentrations

```
cuScale = colourScale(meuse$copper, breaks=5, style='equal',
    opacity=0.8, dec=-1, firstBreak=0)
```

and elevation, with transparency decreasing as elevation increases.

```
elevScale = colourScale(nldElev, style='equal',
    breaks=6, col=terrain.colors,
    firstBreak=0, dec=-1,opacity=c(0.2, 0.9))
```

Soil type is a categorical variable, create a factor and create a colour scale of unique values

```
meuse$soilFac = factor(meuse$soil, levels=c(1,2,3),
    labels=c("Calcareous","Non-Calc's","Red Brick"))
soilScale = colourScale(meuse$soilFac, col="Set2")
```

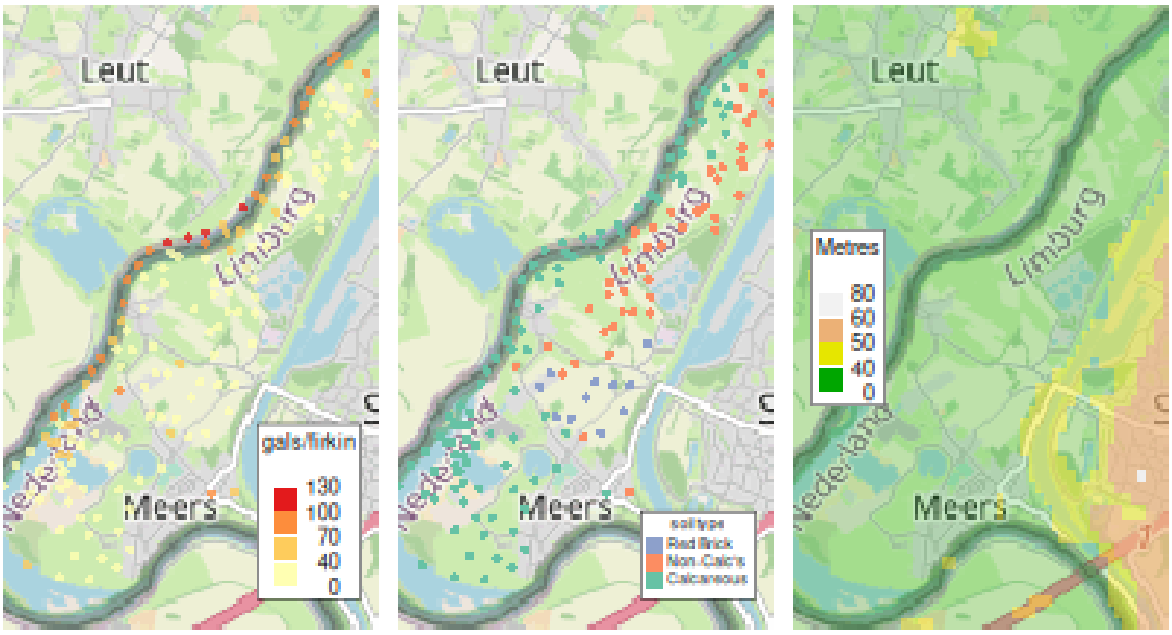
```
map.new(meuse)
plot(nldTiles, add=TRUE)
plot(meuse, col=cuScale$plot,add=TRUE,pch=16)
legendBreaks("bottomright",  breaks=cuScale,
    title="gals/firkin")


map.new(meuse)
plot(nldTiles, add=TRUE)
plot(meuse, col=soilScale$plot,add=TRUE,pch=16)
legendBreaks("bottomright", breaks=soilScale,
    title="soil type", cex=0.7,bg="white")

map.new(meuse)
plot(nldTiles, add=TRUE)
image(nldElev, breaks=elevScale$breaks, col=elevScale$colOpacity,
    legend=FALSE,add=TRUE)
legendBreaks("left", breaks=elevScale, title='Metres',bg="white")
```

# 5 More plots

Rotate the data 50 degrees clockwise with an oblique mercator projection.




(a) Copper (b) soil (c) elevation

Figure 2: Meuse data again

```
if(requireNamespace('rgdal', quietly=TRUE)){
  meuseRot = spTransform(meuse, omerc(meuse, -50))
  tilesRot = openmap(meuseRot, fact=2)
  elevRot = projectRaster(nldElev, crs=projection(meuseRot))
  nldCitiesRot = spTransform(nldCities, CRS(projection(meuseRot)))
}
```

And create new plots

```
# first elevation
map.new(meuseRot)
plot(tilesRot, add=TRUE)
plot(elevRot,add=TRUE,alpha=0.5,col=terrain.colors(8), legend=FALSE)
points(nldCitiesRot)
text(nldCitiesRot, labels=nldCitiesRot$name, pos=3)

scaleBar(meuseRot,pos="topleft", bg="white")

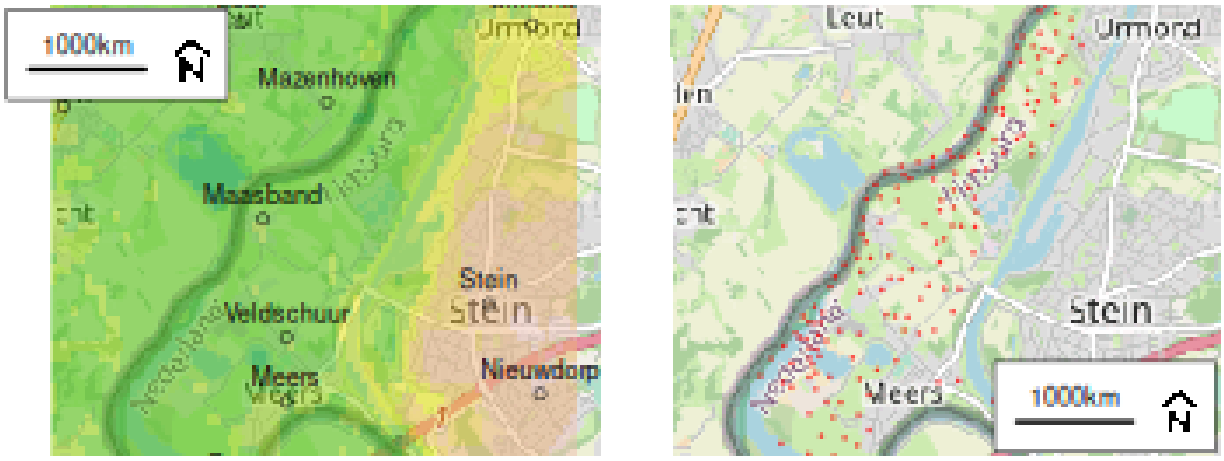
## Warning in scaleBar(meuseRot, pos = "topleft", bg = "white"): rgdal not intalled,
assuming coordinates are either long-lat or metres, and north is up
## Warning in scaleBar(meuseRot, pos = "topleft", bg = "white"): rgdal not intalled,
assuming the plot is long-lat

# then data locations
```

```
map.new(meuseRot)
plot(tilesRot, add=TRUE)
points(meuseRot,col="red", cex=0.3)

scaleBar(proj4string(meuseRot), bg="white")
```

```
## Warning in scaleBar(proj4string(meuseRot), bg = "white"): rgdal not intalled,
assuming coordinates are either long-lat or metres, and north is up
## Warning in scaleBar(proj4string(meuseRot), bg = "white"): rgdal not intalled,
assuming the plot is long-lat
```




(a) elevation (b) data

Figure 3: Rotated map

# 6 Inset maps

```
world = openmap(
    extent(-10,30,40,60),
    path="osm-no-labels")
```

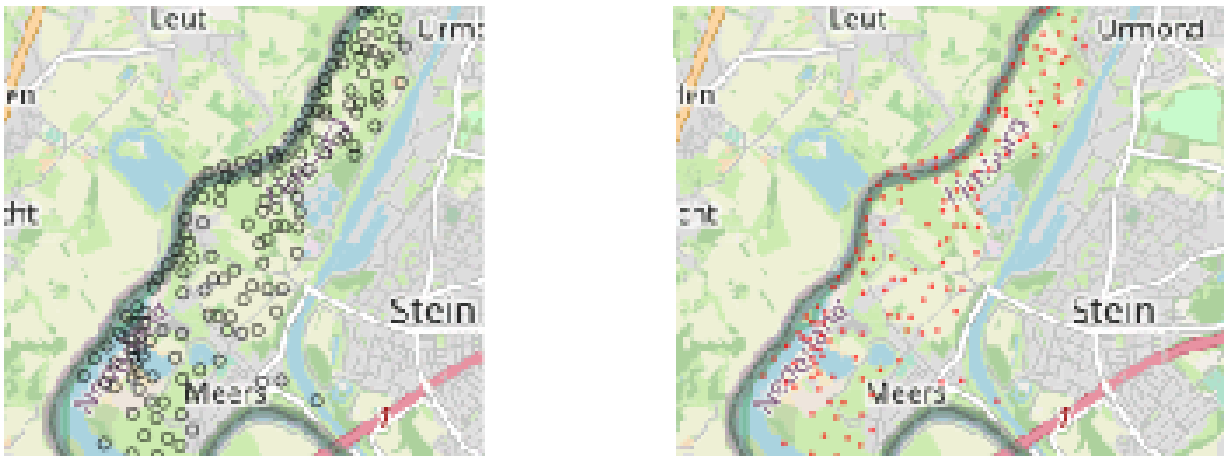
```
# not rotated
map.new(meuse,legendRight=TRUE)
plot(nldTiles, add=TRUE)
points(meuse)

if(requireNamespace('rgdal', quietly=TRUE)) {
  scaleBar(proj4string(meuse),pos="bottomright", bg="white")
```

```
  insetMap(crs=meuse, pos="topright",map=world)
}

# rotated
map.new(meuseRot)
plot(tilesRot, add=TRUE)
points(meuseRot,col="red", cex=0.3)

if(requireNamespace('rgdal', quietly=TRUE)) {
  scaleBar(proj4string(meuseRot), bg="white")
  insetMap(meuseRot, "bottomleft",map=world)
}
```




(a) elevation (b) rotated

Figure 4: Inset map