# GEOmap: mapping and geology in R

Jonathan M. Lees
University of North Carolina, Chapel Hill
Department of Geological Sciences
CB #3315, Mitchell Hall
Chapel Hill, NC 27599-3315
email: jonathan.lees@unc.edu
ph: (919) 962-0695

November, 2007

**Abstract**

Geomap software is aimed at geological applications in mapping.

## 1 Introduction

I developed a set of programs for making complex geological maps in R. These program parallel, to a certain extent, the maps and the mapdata packages already available but they are different in significant ways and provide a slightly differnt set of the utilities. Maps currently available in the mapdata package can be used by GEOmap, but most of the data required by GEOmap is included in a separate package called geomapdata, loaded independantly.

The main differnces between maps and geomap is the lower demands GEOmap has on requiring the maps information to be stored as independant strokes and topologically related polygones. This step, while useful and powerful for many applications, is onerous to set up for maps that are digitized on the fly, either from paper copies or from digital images on the screen.

The other difference is in the handling of projections. GEOmap has a few simple cartographic projections built in and can be expanded later by users.

## 2 Projections

There are 7 cartographic projections currently installed in GEOmap that can be called by the user and applied to data either in the forward mode (Lat-Lon to x-y) or in the inverse mode to go from the projected world back to geographic coordinates.

The set up of the projection is accomplished by running, for example,

```
> library(GEOmap)
```

Spatial Point Pattern Analysis Code in S-Plus
Version 2 - Spatial and Space-Time analysis GEOmap is loaded

```
> options(continue = " ")
> kliuLL = c(56.056, 160.64)
> PROJ = setPROJ(type = 2, LAT0 = kliuLL[1], LON0 = kliuLL[2],
     LATS = NULL, LONS = NULL, DLAT = NULL, DLON = NULL, FN = 0)
```

This makes this location (Kliuchevskoi volcano in Kamchatka, Russia) the origin of a utm sperical projetion. The structure PROJ must be passed as an argument to subsequent calls to GEOmap plotting routines and conversions. The choices for projections can be seen by calling projtype() as in,

```
> projtype()
```

[1] Projection Types [1] 0 = None [1] 1 = merc.sphr [1] 2 = utm.sphr [1] 3 = lambert.cc [1] 4 = stereo.sphr [1] 5 = utm.elps [1] 6 = equid.cyl [1] 99 = old crosson projection And we can see the usage of the projection by loading and plotting a map. First we plot the map with no projection, so the xy coordinates are Lat-Lon and the map will be distorted.

```
> require(geomapdata)
> data(kammap)
> plotGEOmap(kammap, add = FALSE, asp = 1)
```

Next we show how to plot the map in projected form,

```
> plotGEOmapXY(kammap, PROJ = PROJ, add = FALSE)
```

Notice that by resizing the window the map retains the proper aspect ration and the units are correct,

# 3 Simple Map

# 4 Map Structure

The internal structure of a GEOmap objection consists of three elements which are lists themselves. The raw XY coordinates are stored as long vectors on the POINTS list. These are all the geogrphic coordinates of the points int he map structure. The STROKES structure contains the meta data that allows one to acceess the POINTS and perform tasks and create graphical output. The STROKES structure includes a set of vectors which have the following structure:

## 5 Geologic Example

The following illustrates some of the features available in GEOmap. First we set up the data and then begin making the plot after manipulating the database.

```
> data(cosomap)
> data(faults)
> data(hiways)
> data(owens)
> data(cosogeol)
> proj = cosomap$PROJ
> plotGEOmapXY(cosomap, PROJ = proj, add = FALSE, ann = FALSE,
     axes = FALSE)
> cosogeol = boundGEOmap(cosogeol)
> plotGEOmapXY(cosogeol, PROJ = proj, add = TRUE, ann = FALSE,
     axes = FALSE)
> plotGEOmapXY(cosomap, PROJ = proj, add = TRUE, ann = FALSE, axes = FALSE)
> plotGEOmapXY(faults, PROJ = proj, add = TRUE, ann = FALSE, axes = FALSE)
```

The colors here are not very useful, so we can modify them by assigning colors from a given palette, in this case the palette of the program geotouch,

```
> XMCOL = setXMCOL()
> cosocolnumbers = 1:length(cosogeol$STROKES$col)
> newcol = XMCOL[cosocolnumbers]
> cosocolnums = cosogeol$STROKES$col
```

and lastly we must create a legend by matching the colors with the symbols or names of hte units:

```
> ss = strsplit(cosogeol$STROKES$nam, split = "_")
> geo = unlist(lapply(ss, FUN = "getmem", mem = 1))
> UGEO = unique(geo)
> mgeo = match(geo, UGEO)
> cosogeol = boundGEOmap(cosogeol)
> gcol = paste(sep = ".", geo, cosogeol$STROKES$col)
> ucol = unique(gcol)
> spucol = strsplit(ucol, split = "\\.")
> N = length(spucol)
> names = unlist(lapply(spucol, FUN = "getmem", mem = 1))
> shades = unlist(lapply(spucol, FUN = "getmem", mem = 2))
> ORDN = order(names)
> plotGEOmapXY(cosomap, PROJ = proj, add = FALSE, ann = FALSE,
     axes = FALSE)
> plotGEOmapXY(cosogeol, PROJ = proj, add = TRUE, ann = FALSE,
     axes = FALSE)
> plotGEOmapXY(cosomap, PROJ = proj, add = TRUE, ann = FALSE, axes = FALSE)
> plotGEOmapXY(faults, PROJ = proj, add = TRUE, ann = FALSE, axes = FALSE)
> geoLEGEND(names[ORDN], shades[ORDN], 0.28, 0.14, 16, 6)
```

# 6 Geologic Map Symbols

Several standard geological symbols are available for plotting specific faults on plots. These can be seen by executing the grided plot of many line dress ups:

```
> GEOsymbols()
```