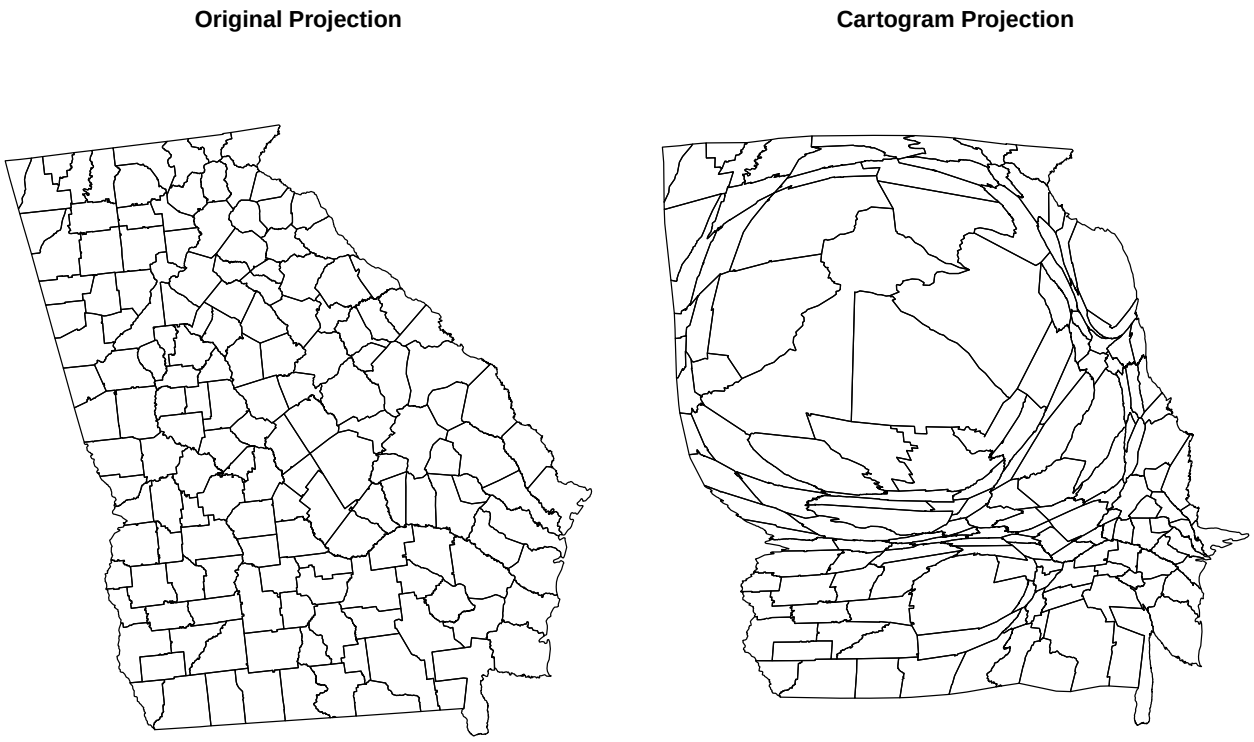
1 A Front End For the Rcartogram Package

This demonstrates the *getcartr* package. Firstly load it:

```
library(getcartr)
```

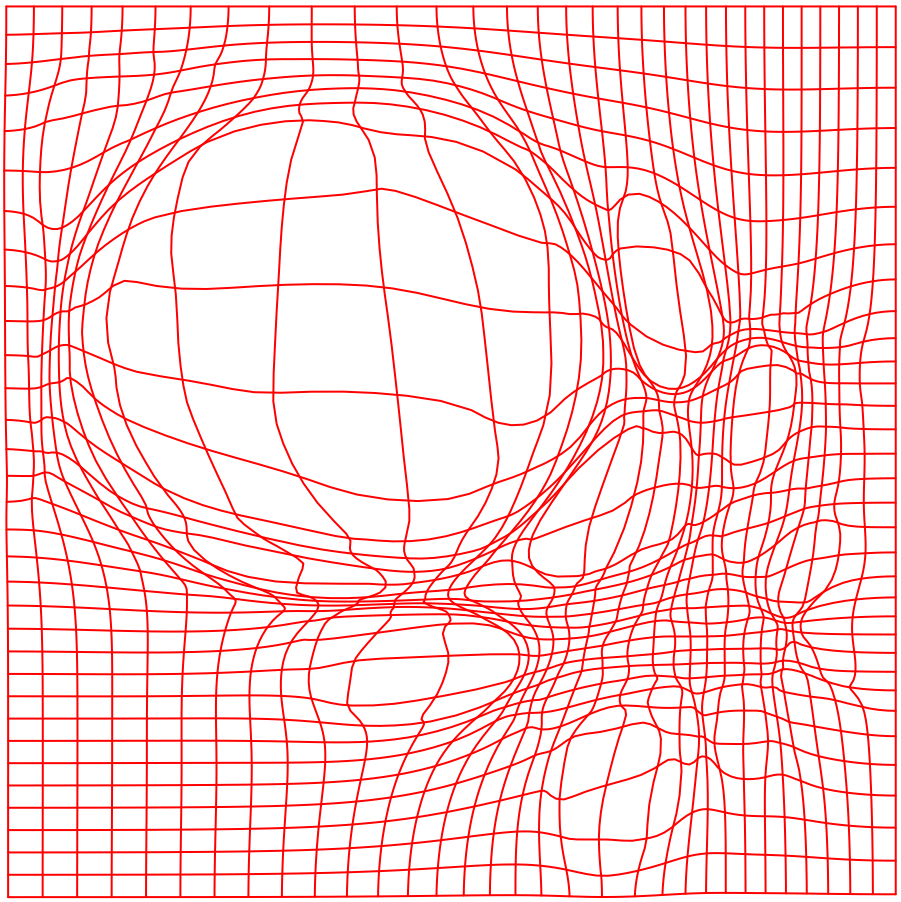
`quick.carto` gives you a cartogram very quickly:

```
data(georgia)
georgia.carto <- quick.carto(georgia,georgia2$TotPop90)
par(mfrow=c(1,2),mar=c(0.5,0.5,3,0.5))
plot(georgia2)
title("Original Projection")
plot(georgia.carto)
title("Cartogram Projection")
```



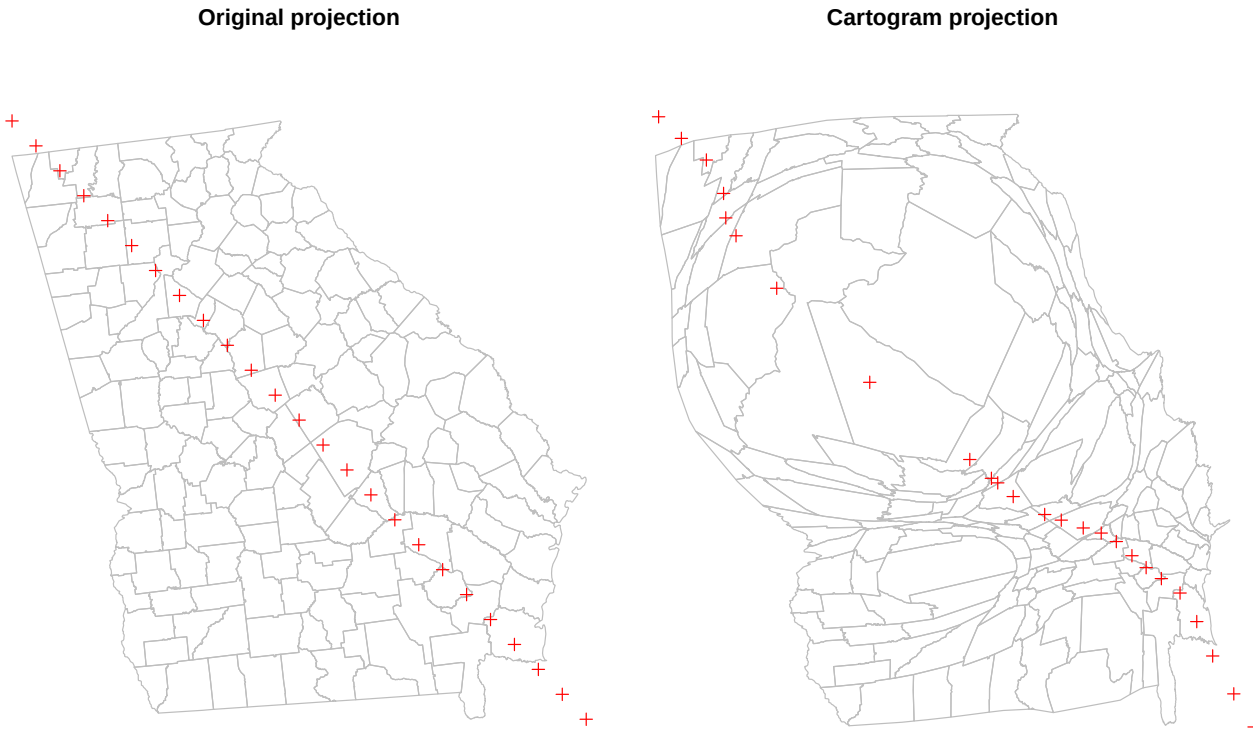
and plot the transformation mesh:

```
# Load an example SpatialPolygonsDataFrame from GISTools
data(georgia)
# Create the cartogram.  TotPop90 contains 1990 county populations
# "georgia2" is used as it has a plane projection coordinate system
georgia.carto <- quick.carto(georgia2,georgia2$TotPop90)
# Draw the mesh
mesh(georgia.carto)
```



and apply the cartogram transform to new sets of points:

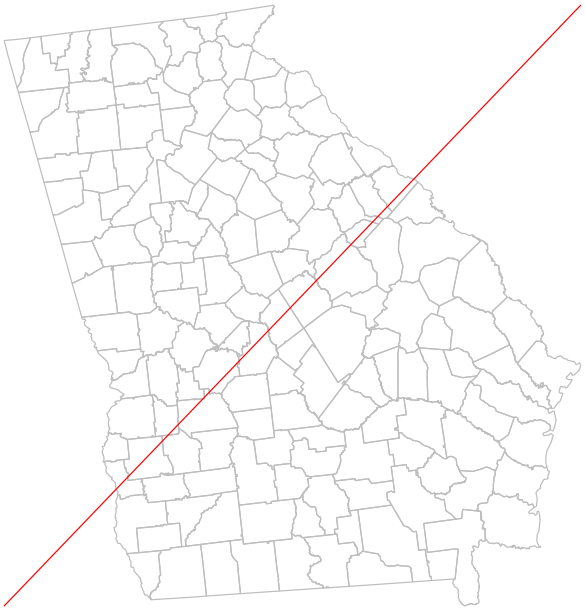
```
# Create a set of 25 points from northwest to southeast corners
xl <- seq(1419424,939220.7,l=25)
yl <- seq(905508,1405900,l=25)
pset <- readWKT(paste("MULTIPOINT(",paste(apply(cbind(xl,yl),1,function(x) paste("(",x[1],x[2],")")),co
#' #  Transform it
pset.carto <- warp.points(pset,georgia.carto)

# Plot the original points
par(mfrow=c(1,2),mar=c(0.5,0.5,3,0.5))
plot(georgia2,border='grey'); plot(pset, add=T, col='red')
title('Original projection')

# Plot in cartogram space
plot(georgia.carto,border='grey'); plot(pset.carto, add=T, col='red')
title('Cartogram projection')
```



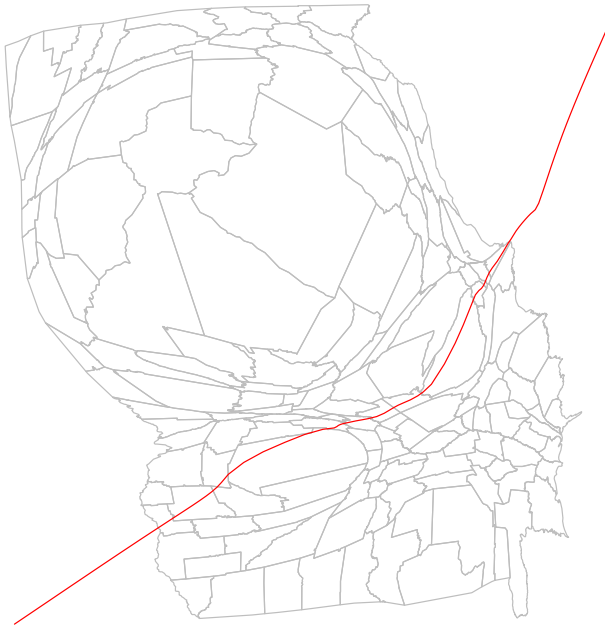
or lines:

```
# Create a line from southwest to northeast corners,  with 100 segments
xl <- seq(939220.7,1419424,l=100)
yl <- seq(905508,1405900,l=100)
aline <- readWKT(paste("LINESTRING(",paste(apply(cbind(xl,yl),1,function(x) paste(x[1],x[2])),collapse=

#  Transform it
aline.carto <- warp.lines(aline,georgia.carto)

# Plot the original line
par(mfrow=c(1,2),mar=c(0.5,0.5,3,0.5))
plot(georgia2,border='grey'); plot(aline, add=T, col='red')
title('Original projection')

# Plot in cartogram space
plot(georgia.carto,border='grey'); plot(aline.carto, add=T, col='red')
title('Cartogram projection')
```

Original projection

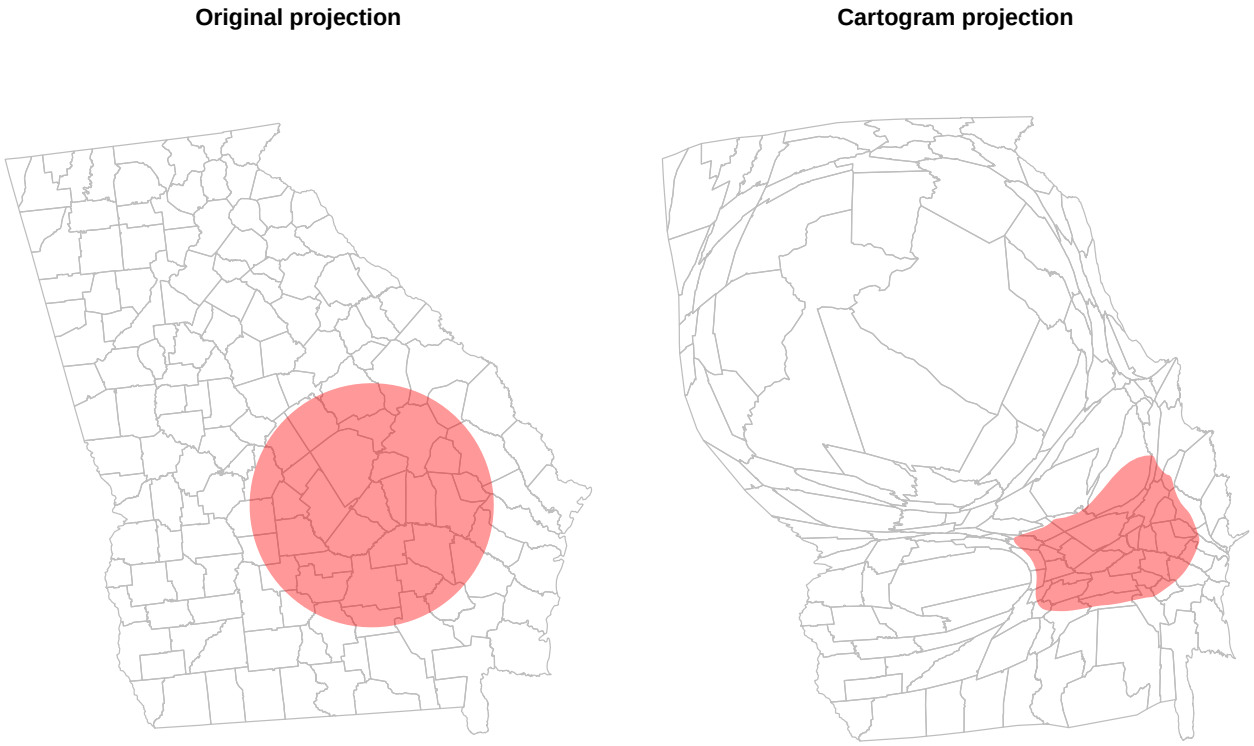


Cartogram projection



... or areas.

```
# Create a pseudocircle with 100 segments
xl <- cos(seq(0,2*pi,l=100))*100000 + 1239348
yl <- sin(seq(0,2*pi,l=100))*100000 + 1093155
circ <- readWKT(paste("MULTIPOLYGON(((",paste(apply(cbind(xl,yl),1,function(x) sprintf("%7.0f %7.0f",x[
#' #  Transform it
circ.carto <- warp.polys(circ,georgia.carto)

# Plot the original circle
par(mfrow=c(1,2),mar=c(0.5,0.5,3,0.5))
plot(georgia2,border='grey'); plot(circ, add=T, col=rgb(1,0,0,0.4), border=NA)
title('Original projection')

# Plot in cartogram space
plot(georgia.carto,border='grey'); plot(circ.carto, add=T, col=rgb(1,0,0,0.4), border=NA)
title('Cartogram projection')
```



2 Alternative approach

Use the `carto.transform` function to create a new *function* to apply the cartogram transform to any shapefile. This function automatically determines whether the input object is `SpatialPolygons*`, `SpatialLines*` or `SpatialPoints*`. Here the Newhaven data in `GISTools` will be used:

```
# Get the newhaven data
require(GISTools)
data(newhaven)
# Plot the untransformed data
par(mfrow=c(1,2),mar=c(0.5,0.5,3,0.5))
plot(blocks) # Census blocks
plot(roads,col='lightgrey',add=TRUE) # Roads
plot(burgres.f,col='darkred',pch=16,add=TRUE) # Forced entry burglaries

# Create the cartogram transform function
to.carto <- carto.transform(blocks,blocks$POP1990)

# Now create a cartogram version of the map
# Plot the blocks carto
plot(to.carto(blocks))
# Add roads,  transformed to cartogram space
plot(to.carto(roads),add=TRUE,col='lightgrey')
# Add forced entry residential burglaries, transformed to cartogram space
plot(to.carto(burgres.f),add=TRUE,col='darkred',pch=16)
```