Zoo955 - Map Projections

Hilary Dugan

1/15/2018

Why maps are wrong

VOX video –> (https://youtu.be/kIID5FDi2JQ?t=38s)

- Surface of the Earth cannot be represented as a plane without distortion
- Projections are just algorithms to move from sphere to plane
- Maps are data visualization at its best

Maps package

There are a lot of mapping packages in R. `Maps` package has its limitations, but is a great way to explore projection. Warning: There are bugs that will drive you nuts!

```
install.packages('maps')
library(maps)
```

```
map(database = "world",
  regions = ".",
  exact = FALSE,
  boundary = TRUE,
  interior = TRUE,
  projection = "",
  parameters = NULL,
  orientation = NULL,
  fill = FALSE,
  ...)
```

There are quite a few arguments that you can customize to explore different projections.

```
?map
```

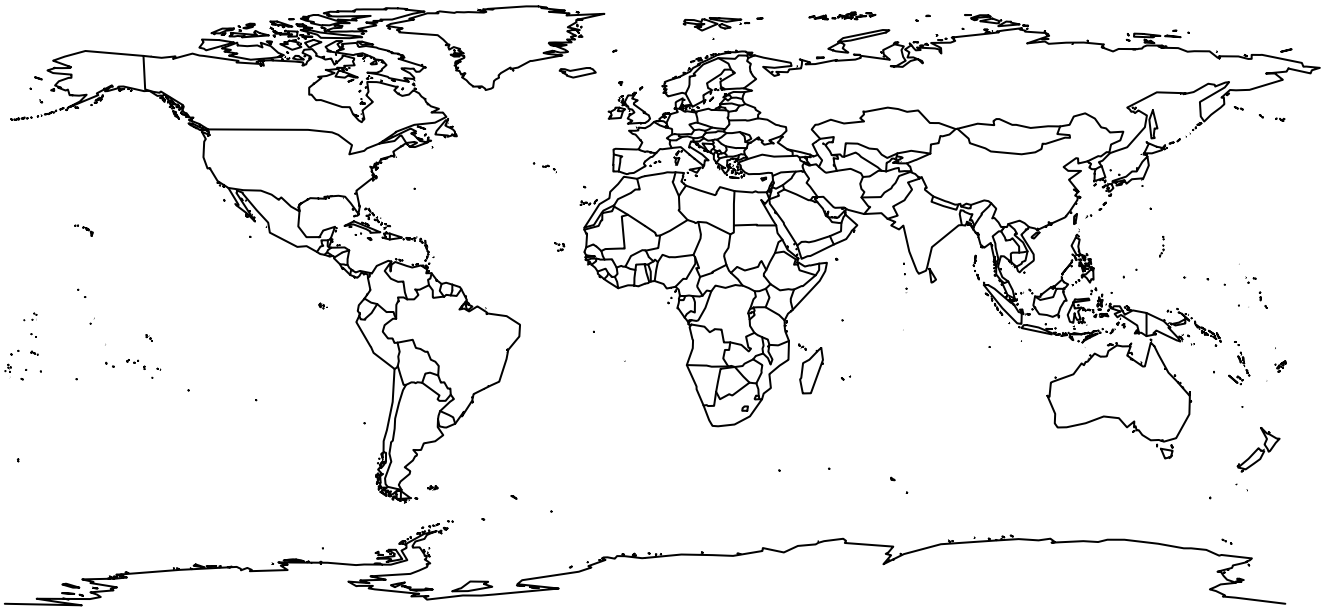
- database: string
  - `'world', 'usa', 'state'`
- regions: string vector of regions in database
  - `'Canada' or 'California'`
- projection: see `?mapproject`
- parameters: see `?mapproject`
- orientation: where the map should be centered
  - `c(latitude, longitude, rotation)`
- fill: draw lines or fill area
  - `TRUE or FALSE`
- wrap: lines that cross too far across the map (due to a strange projection) are omitted*
  - `TRUE or FALSE`
- xlim: range of longitudes
  - Ex: `c(40,60)`
- ylim: range of latitudes
- mar: default margins allow for axes
  - Ex: `c(0,0,0,0)`

- res: resolution of map

Plotting maps in R

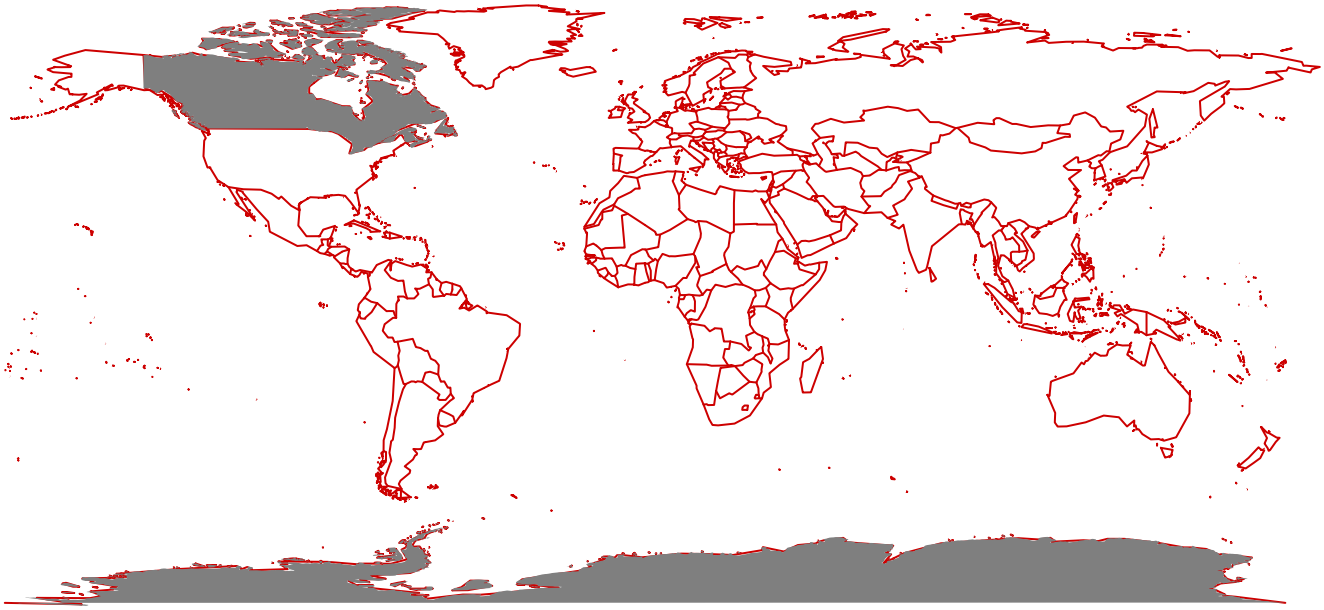
The most basic world map

```
map('world',mar=c(0,0,0,0)) #Default is albers equal-area
```



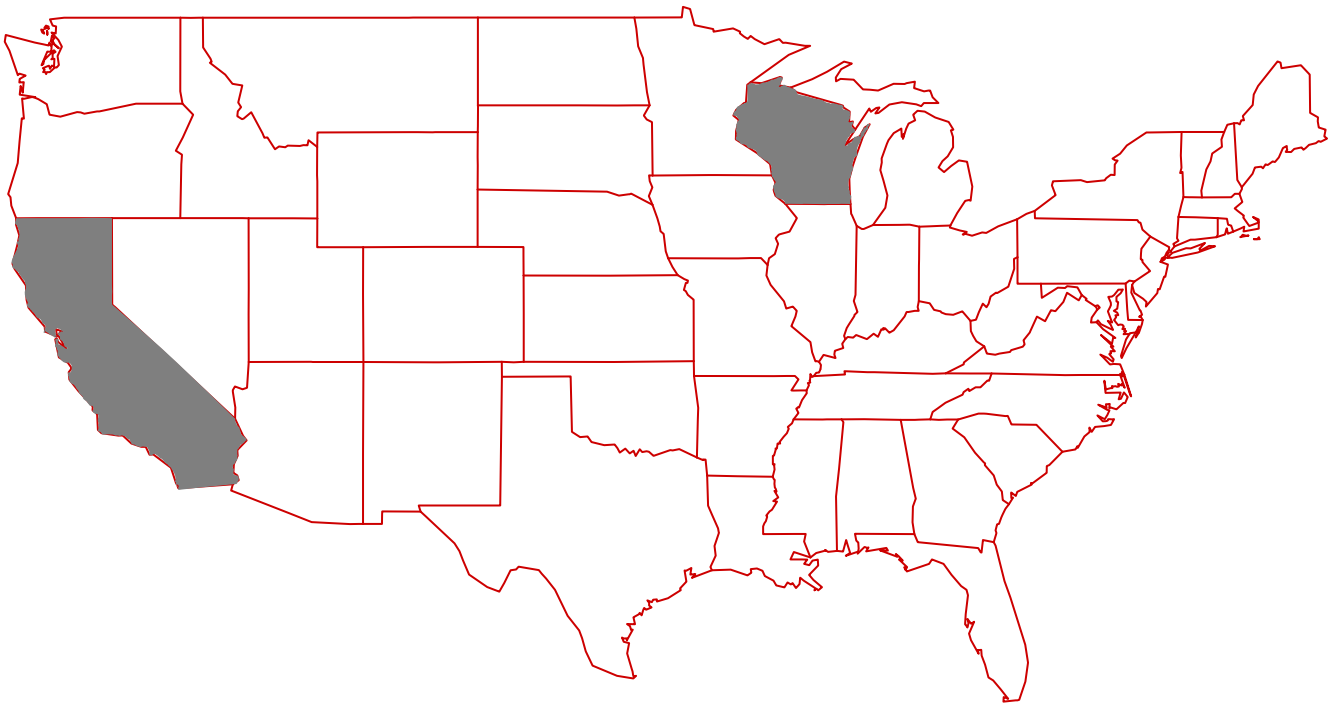
A map with two regions (Canada and Antarctica filled in)

```
map('world',col='red3',mar=c(0,0,0,0)) #Default is albers equal-area
map('world', regions = c('Canada','Antarctica'), add=T, col='grey50', fill = T, border=F)
```



A US state map with Wisconsin and California highlighted. Default projection is **Albers Equal-Area**

```
map('state', col='red3', mar=c(0,0,0,0))
map('state', regions = c('Wisconsin','California'), add=T, col='grey50', fill = T, border=F)
```



Types of Projections

Projections can be classified by surface or property **By surface**

- Cylindrical
- Pseudocylindrical
- Conic
- Azimuthal

By property

- Conformal
  - preserve angles locally
- Equal-area
- Equidistant
- Gnomonic
  - Great circles are straight lines
- or Compromise (Robinson or Winkel-Tripel)

Cylindrical Map Projections

Cylindrical mpas have straight coordinate lines

- The horizontal parallels cross the vertical meridians at right angles
- Different cyclindrical map projections have different scales in spacing the parallel lines

- Popular examples are: Mercator, Behrmann, Gall-Peters

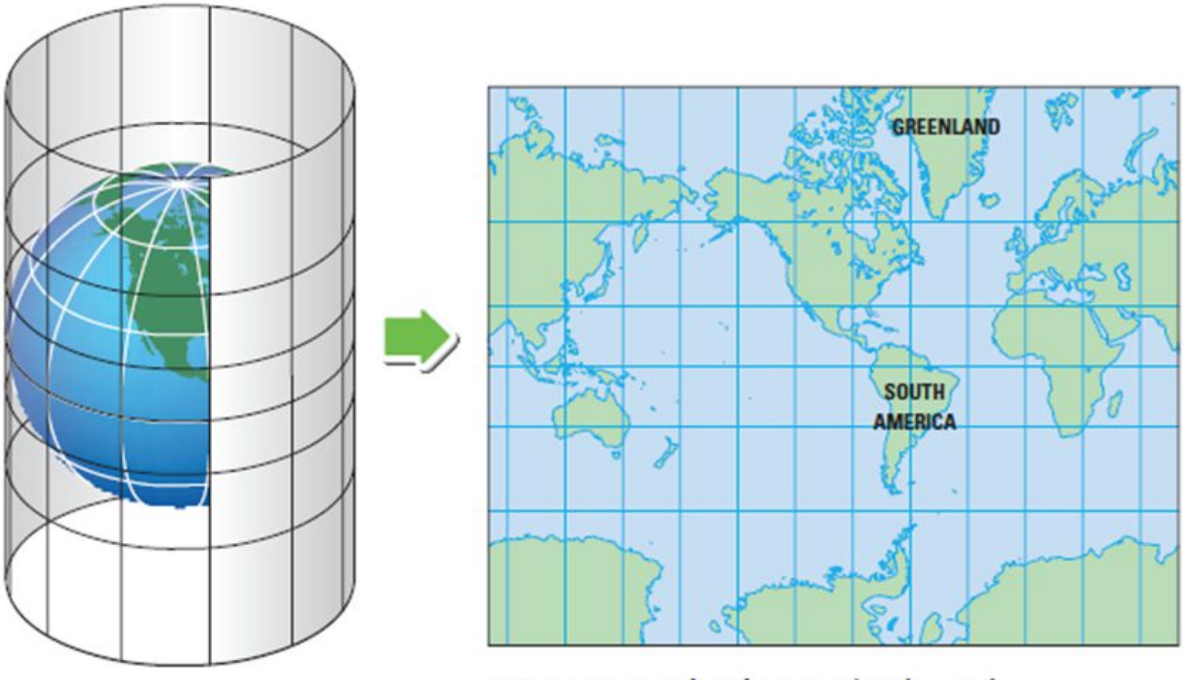
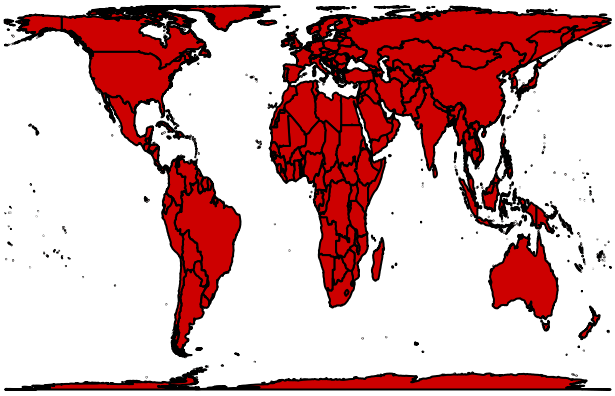


Figure 1: *Figure: Mercator Projection. The mercator projection procedure is often described as wrapping a piece of paper vertically around a globe and unfolding it*

- The specializations of the cylindric equal-area projection are different only in the ratio of the vertical to horizontal axis
- This ratio determines the **standard parallel**, which is the parallel at which there is no distortion and along which distances match the stated scale.
- There are always two standard parallels on the cylindric equal-area projection, each at the same distance north and south of the equator.
  - The standard parallels of the Gall–Peters are 45° N and 45° S.

Example: Cylindrical equal-area (Gall-Peters)

```
map('world', col='red3', proj='cylequalarea', parameters=45, wrap=T,fill=T,mar=c(0,0,0,0)) # Gall-Peter
```



Pseudocylindrical Projections

- Central meridian is a straight line
- Other meridians bow outward
- Parallels are straight lines

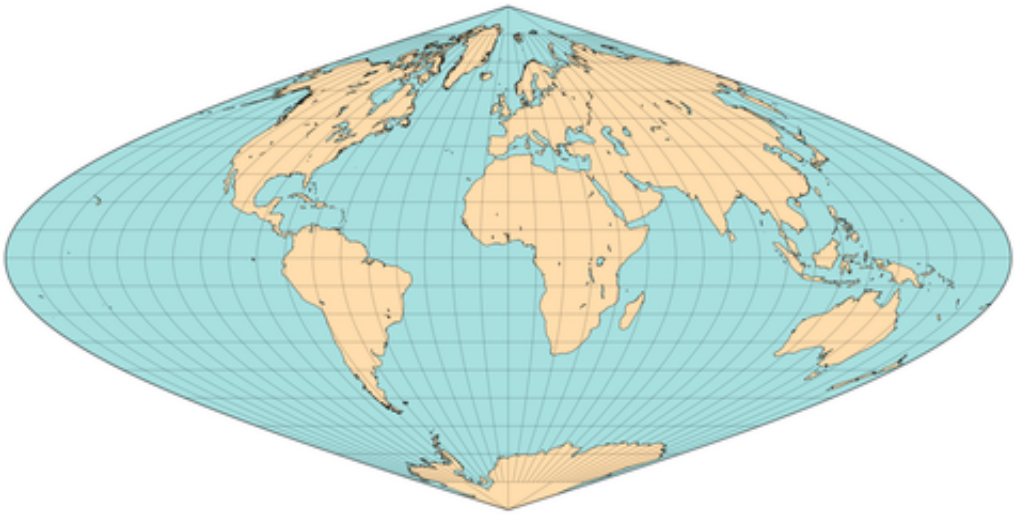
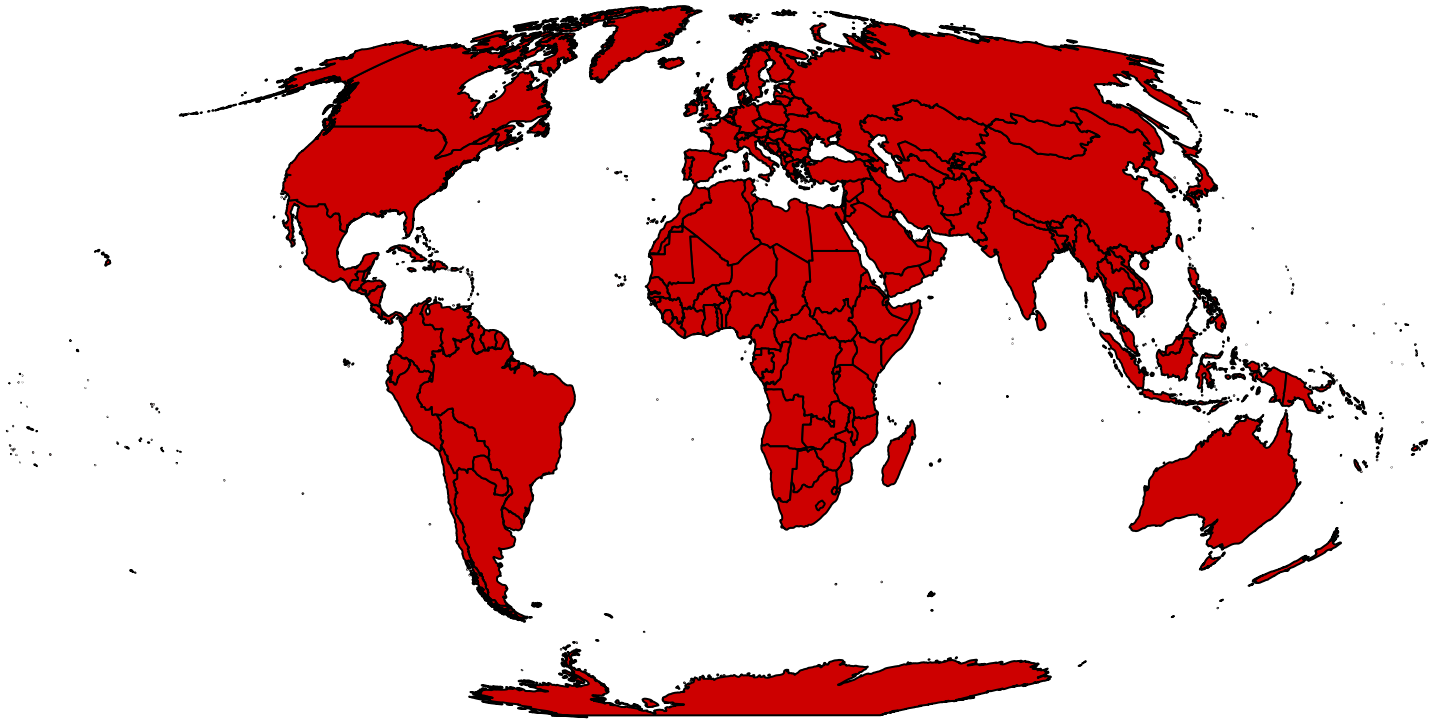


Figure 2: *Figure: Psuedo Projection*

Pseudocylindrical in R - Mollweide (World)

```
map('world', col='red3', proj='mollweide', res=0, fill=T, orientation = c(90,0,0))
```



Conic Projections

- Meridians are mapped to equally spaced lines radiating from apex
- Latitudes are mapped to circular arcs centered on the apex
- Pick two standard parallels
  - Where cone intersects globe
  - Low distortion in scale, shape, and area near standard parallels
- **Best suited regional maps**

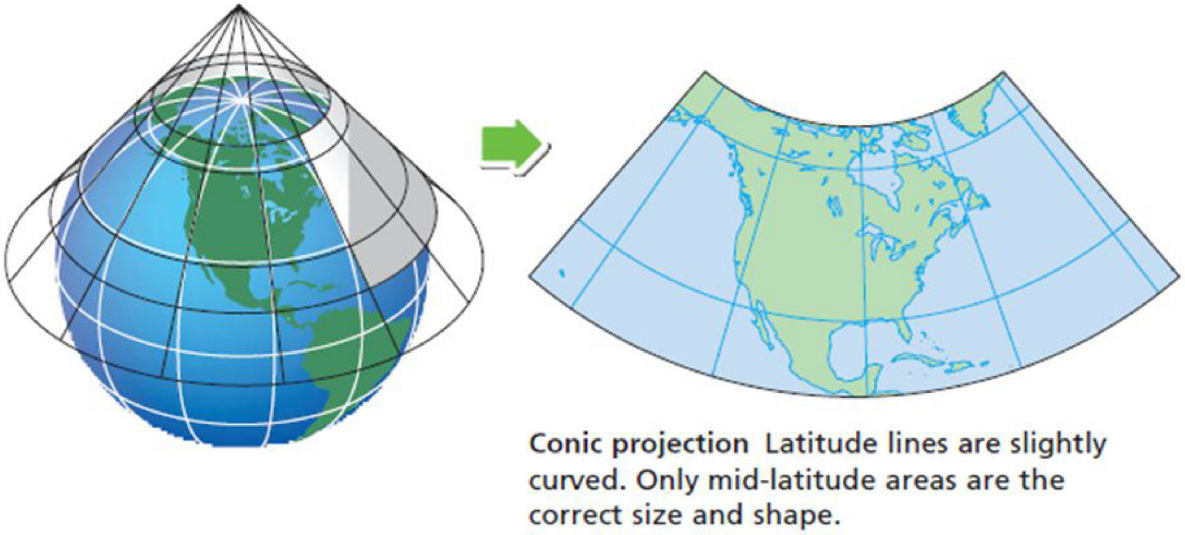


Figure 3: *Figure: Conic Projection*

Common conic projections

- Equidistant conic
  - parallels evenly spaced along the meridians
  - preserves a constant distance scale along each meridian
- Albers conic
  - adjusts the N-S distance between non-standard parallels to compensate for the E-W stretching or compression
  - equal-area map
- Lambert conformal conic
  - adjusts the N-S distance between non-standard parallels to equal the E-W stretching
  - conformal map

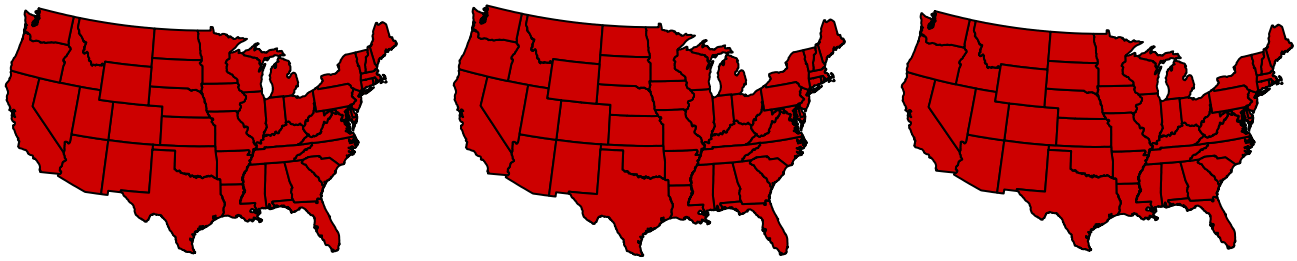
Conic - Equidistant conic (World)

```
map('world', col='red3', proj='simpleconic', parameters=c(20,50), res=0, orientation = c(90,0,0))
```

Conic maps in R - USA

Albers and Lambert are two common map projections that look great a local scale

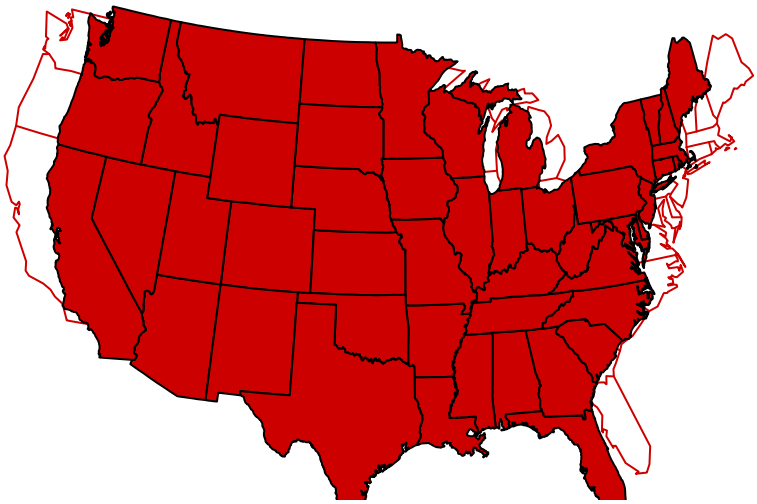
Equidistant Conic Albers Lambert



Conic maps in R - Albers (USA)

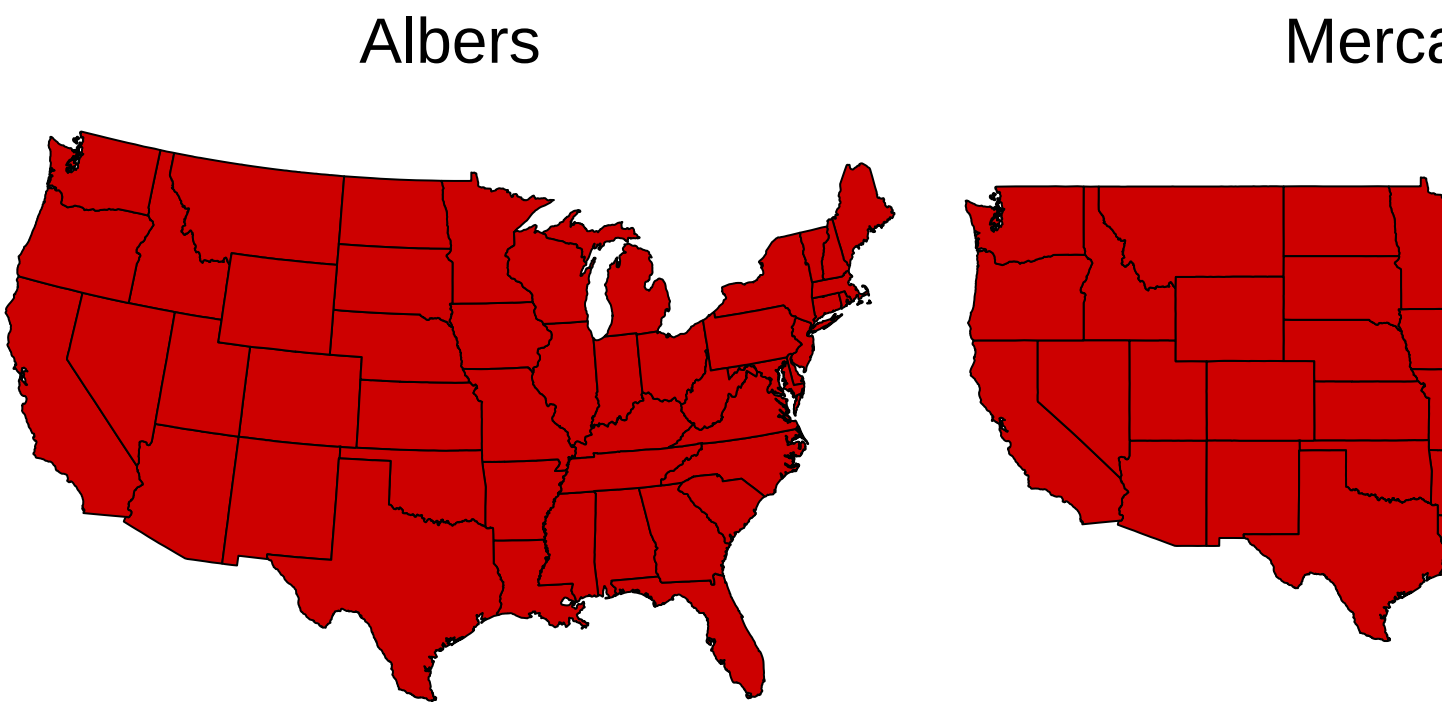
Example of how picking different standard parallels changes the projection

```
map('state',col='red3',proj='albers',parameters =c(30,50))
map('state',col='red3',proj='albers',parameters =c(10,70),add=T,fill=T)
```



Conic vs. Cylindrical - USA

```
par(mfrow=c(1,2),mar=c(0,0,2,0),mgp=c(1,0,0))
map('state',col='red3',proj='albers',parameters=c(20,50),wrap=T, mar=c(0,0,0,0),res=0,fill=T)
mtext('Albers',cex=2)

map('state',col='red3',proj='mercator',wrap=T, mar=c(0,0,0,0),res=0,fill=T)
mtext('Mercator',cex=2)
```



Azimuthal (Planar) Map Projections

- Mapping of radial lines can be visualized as a plane tangent to the Earth, with the central point as tangent point
- Great circles through the central point are represented by straight lines on the map

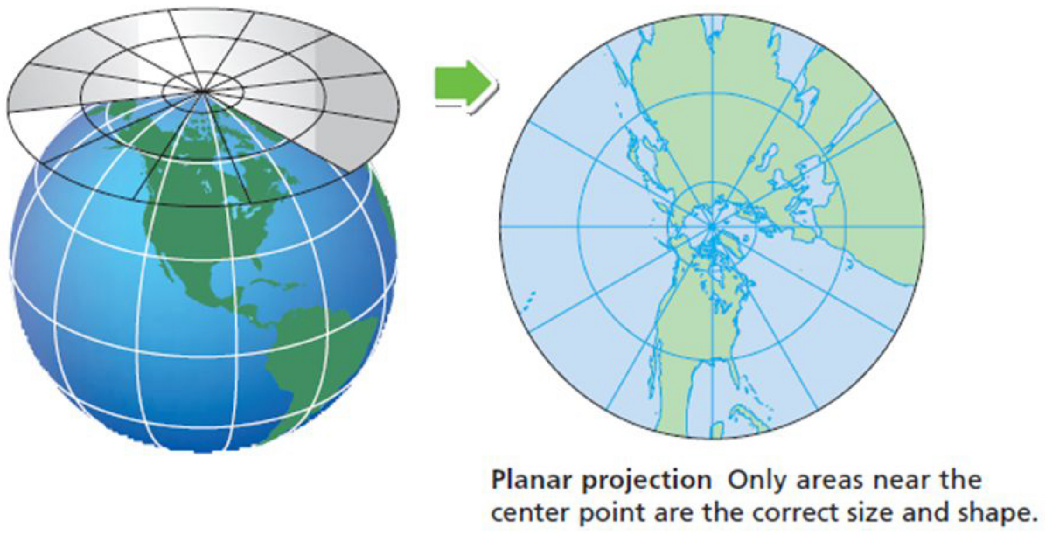
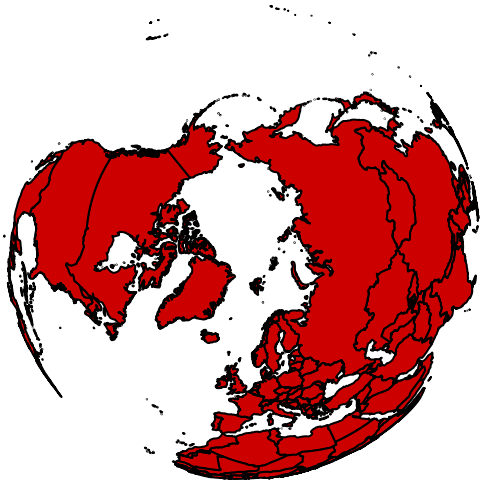


Figure 4: *Figure: Planar projection*

Plotting the poles

```
map('world', projection = 'orthographic',orientation = c(90,0,0),fill=T,
    col='red3')
```

```
## Warning in map("world", projection = "orthographic", orientation = c(90, :
## projection failed for some data
```



```
map('world', projection = 'orthographic',orientation = c(-90,0,0),fill=T,
    col='red3')
```

```
## Warning in map("world", projection = "orthographic", orientation = c(-90, :
## projection failed for some data
```

Tissot's indicatrix

A convienent way to illustrator disortion is by using Tissot's indicatrix.

"A single indicatrix describes the distortion at a single point. Because distortion varies across a map, generally Tissot's indicatrices are placed across a map to illustrate the spatial change in distortion. A common scheme places them at each intersection of displayed meridians and parallels. These schematics are important in the study of map projections, both to illustrate distortion and to provide the basis for the calculations that represent the magnitude of distortion precisely at each point." - Wikipedia entry

Basically, it's like plotting a circle on a globe and seeing how it's projected on a map

- It can characterize local distortions due to map projection
- and illustrates linear, angular, and areal distortions of maps
- Example: Berhmann and Mercator

Tissot's indicatrix - drawTissot

We can make a R function to do this. See github for function.

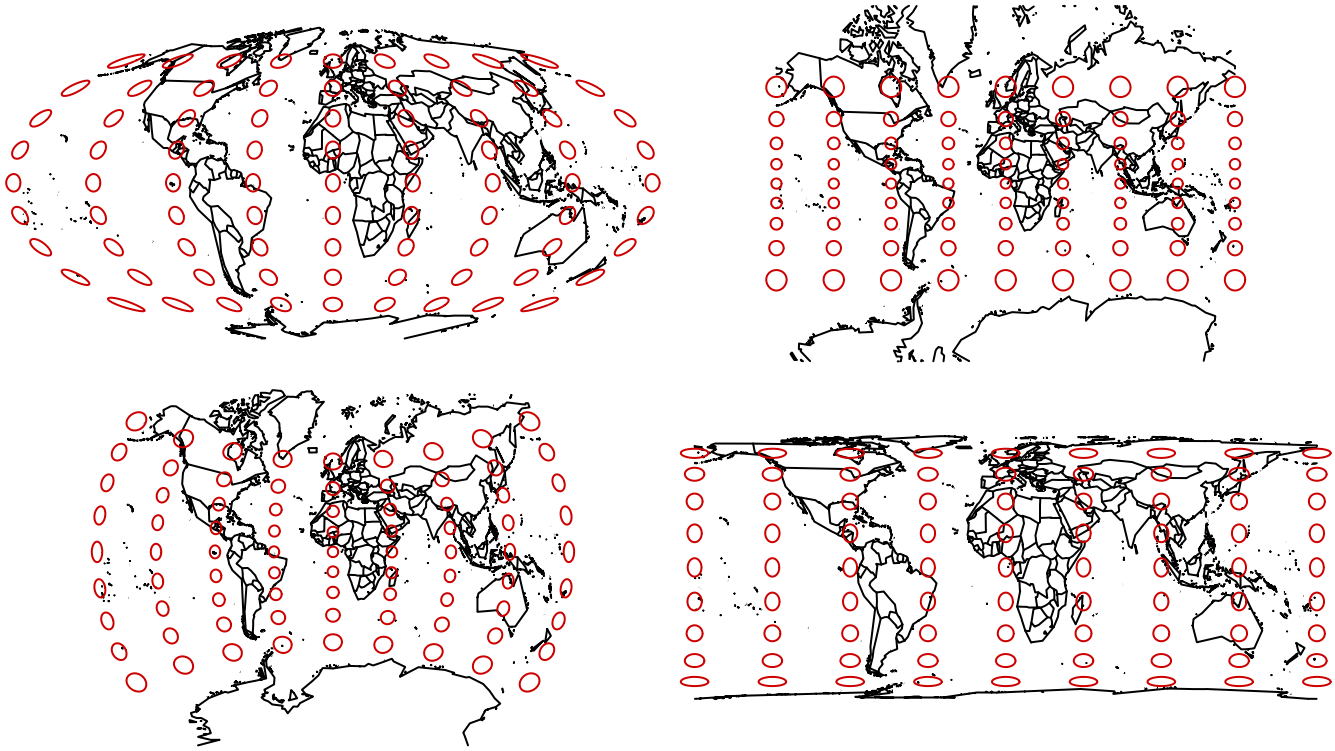
https://github.com/hdugan/Zoo955/tree/master/LectureX_MapProjections/Functions

```
drawTissot <- function(useProj,pars=NULL) {
  require(mapproj)
  #data.frame of distance between meridians with latitude
  df = data.frame(lat = 0:90, distance = 111.3 * cos((0:90)*pi/180))

  tissot <- function(long,lat,useProj,usePars=NULL) {
    ocentre <- c(long, lat)
    t=seq(0,2*pi,0.2)
```

```
    distance = df[df$lat == abs(lat),2]/111.3 #Need to calculate the decrease
    # in distance between meridians as one moves from the equator to the poles

    r=4 ## scale size of circles
    xx=cos(t)*(r/distance)+ocentre[1]
    yy=sin(t)*(r)+ocentre[2]

    lines(mapproject(xx,yy,proj=useProj,orientation = c(90,0,0),parameters = usePars),
          col='red3')
  }

  out1 = seq(-180,180,by = 45)
  out2 = seq(-60,60,by=15)
  for (i in 1:9){
    for (j in 1:9){
      tissot(out1[i],out2[j],useProj = useProj,usePars=pars)
    }
  }
}


par(mar=c(0,0,0,0),mfrow=c(2,2))
a = map('world',projection = 'mollweide',wrap = T,parameters = NULL,orientation = c(90,0,0))
drawTissot('mollweide',pars = NULL)
```

```
## Loading required package: mapproj
```

```
a = map('world',projection = 'mercator',wrap = T,parameters = NULL)
drawTissot('mercator',pars = NULL)

a = map('world',projection = 'vandergrinten',wrap = T,parameters = NULL,orientation = c(90,0,0))
drawTissot('vandergrinten',pars = NULL)
a = map('world',projection = 'cylequalarea',wrap = T,parameters = 30)
drawTissot('cylequalarea',pars = 30)
```



References and Learning Material

- xkcd - map projections
- proj4 - projections
- true size of
- interact with projections