Zoo955 - CRS projections

Hilary Dugan

1/30/2018

Packages

`sp` and `rgdal` will be the most common packages referenced here:

`library(rgdal)` gdal = Geospatial Data Abstraction Library

`library(sp)` sp = spatial

CRS

CRS = Coordinate Reference Systems

- In R, the notation used to describe the CRS is proj4string from the PROJ.4 library.
- Example:`+init=epsg:4121 +proj=longlat +ellps=GRS80 +datum=GGRS87 +no_defs +towgs84=-199.87,74.79,24`
- The CRS is a standardized way of indicating the projection, datum, and ellipsoid of data
- A particular CRS can be referenced by its EPSG code (i.e., epsg:4121). The EPSG is a structured dataset of CRS and Coordinate Transformations.
- See http://spatialreference.org/

Projection, datum, and ellipsoid

Ellipse: Describes the generalized shape of the Earth. All mapping and coordinate systems begin with this description. **Datum**: Defines origin and orientation of the coordinate axes. Where to anchor the 'abstact' coordinates of the Earth.

- All datums specify the ellipsoid. But a given ellipsoid can be used by many datums.
- Common datums: WGS84, NAD83, NAD27
- Common ellipses: WGS84, GRS80, clrk66
- Available datums in rgdal package: `projInfo(type = "datum")`

Projection: The method (or not) for mapping the globe onto a 2D surface

- Two options:
  - Unprojected: Geographic referencing on the ellipse (lat/long)
  - Projected: Easting/northing given on 2D plane
- Available projections in rgdal package: `projInfo(type = "proj")`

Common EPSG codes

credit: NCEAS Overview of Coordinate Reference Systems (CRS) in R

Latitude/Longitude

WGS84 (EPSG: 4326)

- +init=epsg:4326 +proj=longlat +ellps=WGS84+datum=WGS84 +no_defs +towgs84=0,0,0

- CRS used by Google Earth and the U.S. Department of Defense for all their mapping. Tends to be used for global reference systems. GPS satellites broadcast the predicted WGS84 orbits.

NAD83 (EPSG:4269)

- +init=epsg:4269 +proj=longlat +ellps=GRS80 +datum=NAD83 +no_defs +towgs84=0,0,0
- Most commonly used by U.S. federal agencies. Aligned with WGS84 at creation, but has since drifted. Although WGS84 and NAD83 are not equivalent, for most applications they are considered equivalent.

NAD27 (EPSG: 4267)

- +init=epsg:4267 +proj=longlat +ellps=clrk66 +datum=NAD27 +no_defs +nadgrids=@conus,@alaska,@ntv2_0.gsb,@nt
- Has been replaced by NAD83, but is still encountered!
- Almost identical in Wisconsin

Projected (Easting/Northing)

UTM, Zone 10 (EPSG: 32610)

- Zone 10 is used in the Pacific Northwest

Mercator (EPSG: 3857)

- Tiles from Google Maps, Open Street Maps, Stamen Maps

Reading shapefiles

There are a handful of ways to read/write shapefiles in R. I prefer the `rgdal` package.

Using the rgdal package, we are interested in the functions that read OGR data

Interesting fact: OGR used to stand for OpenGIS Simple Features Reference Implementation. However, since OGR is not fully compliant with the OpenGIS Simple Feature specification and is not approved as a reference implementation of the spec the name was changed to OGR Simple Features Library. The only meaning of OGR in this name is historical. OGR is also the prefix used everywhere in the source of the library for class names, filenames, etc. (https://gis.stackexchange.com/questions/216003/what-does-ogr-stand-for)

To read OGR data:

```r
library(rgdal)
```

```
## Warning: package 'rgdal' was built under R version 3.4.3
```

```
## Warning: package 'sp' was built under R version 3.4.3
```

```r
ogrInfo(dsn = 'Data/LakeMendota.shp',layer = 'LakeMendota')
```

```
## Source: "Data/LakeMendota.shp", layer: "LakeMendota"
## Driver: ESRI Shapefile; number of rows: 1
## Feature type: wkbPolygon with 2 dimensions
## Extent: (-89.48369 43.07384) - (-89.36737 43.14706)
## CRS: +proj=longlat +datum=NAD83 +no_defs
## LDID: 87
## Number of fields: 12
##          name type length  typeName
## 1  Permanent_    4     80    String
## 2       FDate    4     80    String
## 3  Resolution   12     10 Integer64
## 4     GNIS_ID    4     80    String
## 5   GNIS_Name    4     80    String
## 6    AreaSqKm    2     24      Real
## 7   Elevation    2     24      Real
## 8   ReachCode    4     80    String
## 9       FType   12     10 Integer64
## 10      FCode   12     10 Integer64
## 11 Shape_Leng    2     24      Real
## 12 Shape_Area    2     24      Real
```

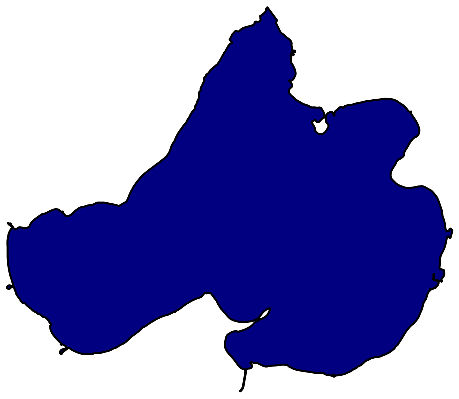
We can see that the shapefile for Lake Mendota is currently not projected. It has lat/long coordinates with NAD83 datum and GRS80 ellipse

To load shapefile:

```r
library(rgdal)
mendota = readOGR(dsn = 'Data/LakeMendota.shp',layer = 'LakeMendota')
```

```
## OGR data source with driver: ESRI Shapefile
## Source: "Data/LakeMendota.shp", layer: "LakeMendota"
## with 1 features
## It has 12 fields
## Integer64 fields read as strings:  Resolution FType FCode
```

```r
plot(mendota, col='navy')
```



Transform CRS

Let's transform the CRS from NAD83 to WGS84:

```r
mendota_WGS84 <- spTransform(mendota, CRS("+init=epsg:4326")) #shorter method using EPSG codes (however
mendota_WGS84 <- spTransform(mendota, CRS("+init=epsg:4326 +proj=longlat +datum=WGS84 +no_defs +ellps=W
```

Let's check if we were successful in transforming the CRS of the shapefile. I actually prefer `crs` in the `raster` package to `CRS` in the `rgdal` package.

```
library(raster)
```

```
## Warning: package 'raster' was built under R version 3.4.3
```

```
crs(mendota_WGS84)
```

```
## CRS arguments:
##  +init=epsg:4326 +proj=longlat +datum=WGS84 +no_defs +ellps=WGS84
## +towgs84=0,0,0
```

```
crs(mendota)
```

```
## CRS arguments:
##  +proj=longlat +datum=NAD83 +no_defs +ellps=GRS80 +towgs84=0,0,0
```

If you plot these both, you can't tell the difference in the datum. This is because NAD83 is centered over the middle of the North American plate, so drift is minimal in Wisconsin

How can we see if there is a difference? We can extract the x,y coordinates for two polygons. Shapefiles have a cryptic nested slot layout, but coordinates can be extracted from our polygon by:

`mendota@polygons[[1]]@Polygons[[1]]@coords`

If we subtract the two, we may see a difference. Most of the points are exactly the same. A few are e-15 off.

```
mendota@polygons[[1]]@Polygons[[1]]@coords - mendota_WGS84@polygons[[1]]@Polygons[[1]]@coords
```

Project shapefile from geographic to projected coordinates

It is just as easy to project the shapefile as it was to change the datum.

Dane County falls in UTM (Universal Transverse Mercator) zone 16. Note: The UTM system is not a single map projection. The system instead divides the Earth into sixty zones, each being a six-degree band of longitude, and uses a secant transverse Mercator projection in each zone.

```
mendota_UTM16 <- spTransform(mendota, CRS("+init=epsg:32616")) #project to UTM zone 16
mendota_UTM16
```

```
## class       : SpatialPolygonsDataFrame
## features    : 1
## extent      : 297862.2, 307362.5, 4771864, 4779978  (xmin, xmax, ymin, ymax)
## coord. ref. : +init=epsg:32616 +proj=utm +zone=16 +datum=WGS84 +units=m +no_defs +ellps=WGS84 +towgs
## variables   : 12
## names       : Permanent_,      FDate, Resolution,  GNIS_ID,    GNIS_Name, AreaSqKm, Elevation,      
## value       :  143249470, 2012/05/25,          2, 01569337, Lake Mendota,   39.612,         0, 07090
```

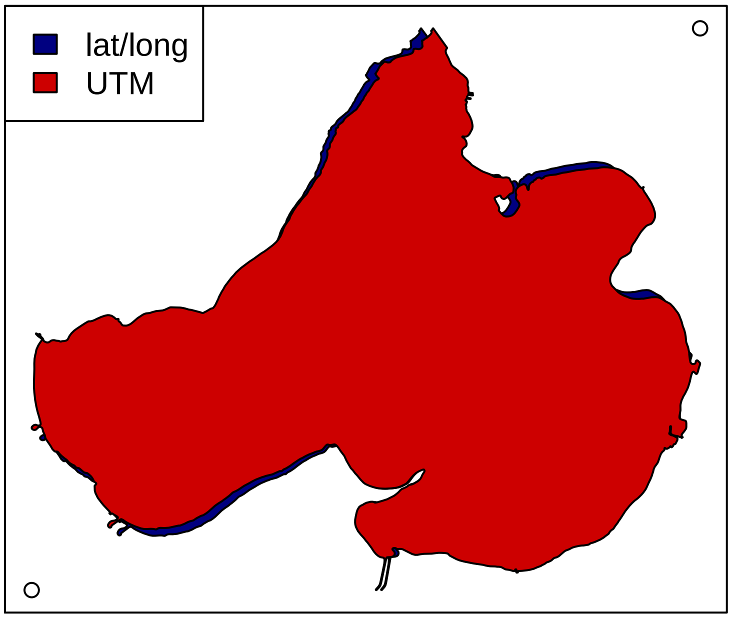
You can see the data extent is now `297862.2, 307362.5, 4771864, 4779978 (xmin, xmax, ymin, ymax)`.

UTM units are in meters, rather than degrees, which makes arithmetic calculations much easier. Here we can tell that Mendota is about 9550 m across (307362.5 - 297862.2)

Visualizing the difference between CRS

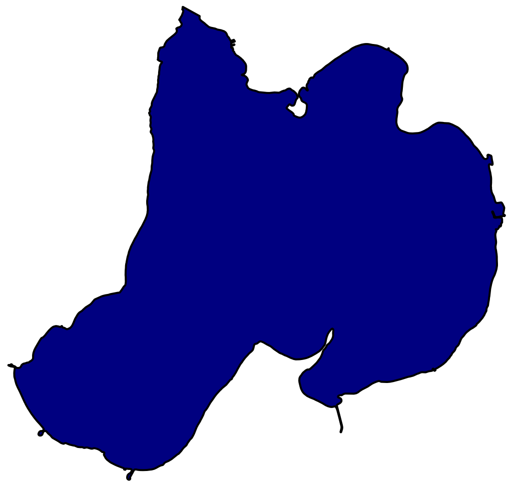
Ok, this is a little hacky, because it's difficult to (because you usally shouldn't!!) plot two shapefiles with different coordinates systems on top of one another. However, if we do, we can see the difference in shape.

```r
plot(mendota,col='navy')
par(new =T)
plot(c(297862.2, 307362.5), c(4771864, 4779978), yaxt='n',
     xaxt='n', ann=FALSE) # Have to make a new empty plot
plot(mendota_UTM16,add=T,col='red3')
legend('topleft',fill = c('navy','red3'),legend = c('lat/long','UTM'))
```



There isn't that much of a difference, but that's because both of these CRS represent Mendota well. What if we chose the wrong UTM zone? Like UTM zone 38?

```r
mendota_UTM38 <- spTransform(mendota, CRS("+init=epsg:26910")) #project to UTM zone 10 (pacific north w
plot(mendota_UTM38,col='navy')
```



*Note, the x distance is now closer to 9750 meters. Distances are distorted as well.

Homework

Reminder: Email assignment .pdf file. Recommended to use RMarkdown, so code and output are all in one place. Email by Monday at 9am.

1) Download the LTER lake shapefiles from the LTER database. Map Lake Mendota. Add a point for the location of the CFL. (Keep these files, we'll use them in future classes)
2) Download the National Land Cover (NLCD) dataset for 2011. Load it into R. What is the CRS? (Keep this data, we'll use it in future classes)
3) What's the best way to check that CRS of two objects are identical?