Python for Bioinformatics

adventures in bioinformatics

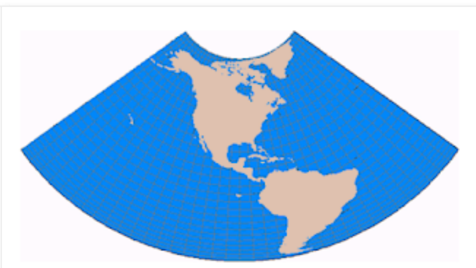
Thursday, June 25, 2020

Albers projection

The standard projection used in converting latitudes and longitudes on the (roughly) spherical earth to a planar map depends on what you're projecting.

Most people know about the Mercator projection.

The one used extensively for maps of the United States is called the Albers Equal-Area Conic projection.



The wikipedia article gives some formulas:

$$x = \rho \sin \theta$$
$$y = \rho_0 - \rho \cos \theta$$

where

$$n = \tfrac{1}{2} (\sin \varphi_1 + \sin \varphi_2)$$
$$\theta = n (\lambda - \lambda_0)$$
$$C = \cos^2 \varphi_1 + 2n \sin \varphi_1$$
$$\rho = \tfrac{R}{n} \sqrt{C - 2n \sin \varphi}$$
$$\rho_0 = \tfrac{R}{n} \sqrt{C - 2n \sin \varphi_0}$$

You must first choose two reference latitudes. Latitudes are referred to

Jackson's Mill WV

Search This Blog

Search

Labels

- 16S rRNA (10)
- alignments (7)
- bayes (17)
- binary (5)
- bindings (1)
- Bioconductor (7)
- bioinformatics (77)
- BLAST (8)
- book (1)
- C (8)
- calculus (14)
- command line (15)
- cool stuff (1)
- COVID-19 (17)
- crypto (10)
- ctypes (5)
- Cython (3)
- dental project (6)
- distributions (20)
- DNA binding sites (6)
- duly quoted (31)

by the letter phi and longitudes by the letter lambda. So the two references are phi_1 and phi_2.

You also choose a center for the map, at phi_0, lambda_0.

From these four values you calculate n, C and then rho_0, which are each the same for every transformation in this projection.

Finally, for each coordinate phi, lambda one calculates rho and theta and then finally

```
x = rho sin theta
y = rho_0 - rho cos theta
```

This, however, is for the assumption of a spherical earth. The equations for an ellipsoid are a bit harder.

The wikipedia article gives this url for a pdf and I found the same report referenced in this answer to a question on Stack Exchange.

so that was lucky, because it gave me not only the equations but also a worked numerical example for each, which helped greatly in finding my mistakes in the code.

The math is explained in a write-up done in LaTeX, as a Dropbox link to a pdf. The math has been encoded as Python scripts sphere.py and ellipsoid.py here.

Here are screenshots from the manual:

- EMBOSS (3)
- fun (16)
- Geometry (26)
- go (4)
- HMM (6)
- homework (5)
- Illumina (12)
- Instant Cocoa (74)
- linear algebra (12)
- links (1)
- Linux (8)
- maps (5)
- matplotlib (38)
- matrix (7)
- maximum likelihood (5)
- meta (21)
- motif (11)
- Note to self (1)
- numpy (18)
- OS X (46)
- phy trees (32)
- phylogenetics (64)
- Pretty code (7)
- probability (8)
- puzzles (2)
- PyCogent (34)
- PyObjC (59)
- Python (2)
- Qiime (9)
- Quick Objective-C (15)
- Quick Python (4)
- Quick Unix (3)
- R (29)
- RPy2 (14)
- sequence models (11)
- simple math (68)
- simple Python (115)
- simulation (43)
- software installs (41)
- ssh (8)
- stats (39)

Given: Clarke 1866 ellipsoid: $a = 6378206.4$ m
$e^2 = 0.00676866$
or $e = 0.0822719$
Standard parallels: $\phi_1 = 29°\ 30'$ N. lat.
$\phi_2 = 45°\ 30'$ N. lat.
Origin: $\phi_0 = 23°$ N. lat.
$\lambda_0 = 96°$ W. long.
Point: $\phi = 35°$ N. lat.
$\lambda = 75°$ W. long.

Find: ρ, θ, x, y, k, h, ω

From equation (14–15),

$$
\begin{aligned}
m_1 &= \cos 29.5°/(1-0.00676866 \sin^2 29.5°)^{1/2} \\
&= 0.8710708 \\
m_2 &= \cos 45.5°/(1-0.00676866 \sin^2 45.5°)^{1/2} \\
&= 0.7021191
\end{aligned}
$$

From equation (3–12),

$$
\begin{aligned}
q_1 &= (1-0.00676866) \{\sin 29.5°/(1-0.00676866 \sin^2 29.5°) \\
&\quad -[1/(2\times 0.0822719)] \ln [(1-0.0822719 \sin 29.5°)/ \\
&\quad (1+0.0822719 \sin 29.5°)]\} \\
&= 0.9792529
\end{aligned}
$$

Using the same formula for q_2 (with ϕ_2 instead of ϕ_1),

$$q_2 = 1.4201080$$

APPENDIX A: NUMERICAL EXAMPLES

Using the same formula for q_0 (with ϕ_0 instead of ϕ_1),

$$q_0 = 0.7767080$$

From equations (14–14), (14–13), and (14–12a) in order,

$$
\begin{aligned}
n &= (0.8710708^2 - 0.7021191^2)/(1.4201080-0.9792529) \\
&= 0.6029035 \\
C &= 0.8710708^2 + 0.6029035 \times 0.9792529 \\
&= 1.3491594 \\
\rho_0 &= 6378206.4 \times (1.3491594-0.6029035\times 0.7767080)^{1/2}/0.6029035 \\
&= 9{,}929{,}079.6 \text{ m}
\end{aligned}
$$

These are the constants for the map. For $\phi = 35°$ N. lat. and $\lambda = 75°$ W. long.: Using equation (3–12), but with ϕ in place of ϕ_1,

$$q = 1.1410831$$

$$
\begin{aligned}
\rho &= 6378206.4 \times (1.3491594-0.6029035\times 1.1410831)^{1/2}/0.6029035 \\
&= 8{,}602{,}328.2 \text{ m} \\
\theta &= 0.6029035 \times [-75°-(-96°)] = 12.6609735° \\
x &= 8602328.2 \sin 12.6609735° = 1{,}885{,}472.7 \text{ m} \\
y &= 9929079.6 - 8602328.2 \cos 12.6609735° \\
&= 1{,}535{,}925.0 \text{ m}
\end{aligned}
$$

The test is a latitude of 35°N., -96°W., which should give a result (for the ellipsoid) of

$$
\begin{aligned}
x &= 8602328.2 \sin 12.6609735° = 1{,}885{,}472.7 \text{ m} \\
y &= 9929079.6 - 8602328.2 \cos 12.6609735° \\
&= 1{,}535{,}925.0 \text{ m}
\end{aligned}
$$

- Sudoku (1)
- Swift (14)
- Ubuntu (8)
- Unifrac (8)
- Unix (7)
- What we're eating (2)
- what we're listening to (5)
- what we're reading (30)
- what we're saying (1)
- What we're thinking (2)
- Xcode (9)
- Xgrid (12)
- XML (9)

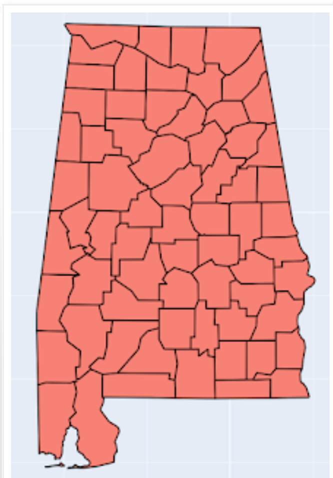
Unique visitors since Feb 21, 2011

```
> python ellipsoid.py
test1
p0: 23.0
l0: -96.0
p:  35.0
l:  -75.0
x:  1885472.7
y:  1535925.0
```

The first part of the output is the center we chose. The result for x and y matches the source.

So then, for a script in the same directory as `ellipsoid.py`, we can do `import project` and do

```
> python3 map_counties.py counties.geo.txt AL
```



No longer squashed.

Naturally, there is software out there that will do this. In particular, the Proj transformation software.

I obtained it with Homebrew

> brew install proj

```
> echo 55.2 12.2 | proj +proj=merc +lat_ts=56.5 +ellps=GRS80
3399483.80      752085.60
```

which matches their example.

Blog Archive

- 2022 (2)
- 2021 (13)
- 2020 (34)
  - December (2)
  - July (2)
  - June (6)
    - Shapefiles
    - Pythagorean theorem redux
    - Albers projection
    - Plotting polygons
    - GeoJSON
    - COVID-19 data analysis
  - May (19)
  - April (4)
  - March (1)
- 2018 (1)
- 2017 (7)
- 2016 (1)
- 2015 (16)
- 2014 (3)
- 2013 (2)
- 2012 (77)
- 2011 (174)
- 2010 (224)
- 2009 (265)
- 2008 (76)

About Me

telliott99

I'm retired, but used to teach and do research in Microbiology. This blog started as a record of my adventures learning bioinformatics and using Python. It has expanded

Of course, we really want the Albers equal area conical projection based on the ellipsoid.

```
> echo -75 35 | proj +proj=aea +lat_1=29.5 +lat_2=44.5 +lat_0=23 +lon_0=-96 +ellps=clrk66
1887211.95      1533994.75
```

These are close but don't quite match. I will have to explore Proj more to know why.

Put the input data into a file and do:

```
> cat data.txt | proj +proj=aea +lat_1=29.5 +lat_2=44.5 +lat_0=23 +lon_0=-96 +ellps=clrk6
1887211.95      1533994.75
>
```

Posted by telliott99 at 6/25/2020 11:20:00 AM

Labels: maps

Newer Post Home Older Post

to include Cocoa, R, simple math and assorted topics. As bbum says, it's so "google can organize my head." The programs here are developed on OS X using R and Python plus other software as noted. YMMV. I've had to turn comments off for the blog. Nothing but spam anymore. The intrepid reader will be able to find me. Hint: +"9" and I use gmail.

View my complete profile