Coordinate Systems: PROJ.4, EPSG and OGC WKT

Frank Warmerdam
FOSS4G 2006
Objectives

- Coordinate systems background
- PROJ.4 examples, and background
- EPSG examples and background
- WKT examples and background
- Variations in “Well Known Text”
- Example coordinate systems in MapServer, GRASS, GDAL/OGR and PostGIS
- Datum Dangers
PROJ.4 Background

- A coordinate system transformation library
- Used by GDAL/ OGR, MapServer, GRASS, Mapguide OS, and a variety of other programs.
- Supports over 100 projections
- Supports datum shifting with grid shift files and 3/7 parameter transforms.
- Simple “command line” format for describing coordinate systems.
- Hosted at http://www.remotesensing.org/proj
- Developed by Gerald Evenden (then of USGS), co-maintained by Frank Warmerdam now.
Geographic Coordinate Systems

e.g. “lat long, WGS84”
• Position as latitude (degrees north of equator) and longitude (degrees east of prime meridian)
• Ellipsoid (e.g. Clark 1880, or WGS84)
  - Semi-major axis (center to equator, in meters)
  - Semi-minor axis (center to pole, in meters)
• Prime Meridian (normally Greenwich)
• Units (degrees, radians, gradians)
• Datum ....
Datum

- Based on an ellipsoid
- Roughly, a name for a survey network
- Surveys accumulate error
- MAGIC!
- Conversions done with:
  - Grid shift files (ie. NAD27/83)
  - 3/7 parameter transformations
  - Polynomials (not supported by PROJ)
- Conversions often expressed relative to WGS84
- Is WGS84 the universal datum?
Projected Coordinate System

eg. “UTM zone 11 north, WGS84”
Location expressed in meters east/north of some reference location.

Needs:
• Projection method (ie. Transverse Mercator)
• Parameters (ie. Central Meridian, False Easting)
• Geographic Coordinate Systems (ie. WGS84)
• Linear units (ie. Meters, or feet)
**PROJ.4: Ellipsoid (Spheroid)**

Defined as:
- `+ellps=<name>`
- `+a=<semi_major_axis>`
- `+b=<semi_minor_axis>`
- `+a=<semi_major_axis>`
- `+rf=<inverse_flattening>`

Axis defined in meters.

Examples:
- "+ellps=WGS84"
- "+a=6378137.0 +rf=298.257223563"

Use "cs2cs -le" to get a list of known ellipsoids.
Geographic

Aka lat/long + proj=latlong

- Not really a projection!
- Still need datum or at least ellipsoid.
- Can include prime meridian.
- Units is implicitly degrees.
PROJ.4: Datums

Defined as:

- \([+\text{datum} = <\text{datum\_name}>]\)
- \([+\text{towgs84} = <x\_shift>,<y\_shift>,<z\_shift>]\)
- \([+\text{towgs84} = <xs>,<ys>,<zs>,<xr>,<yr>,<zr>,<s>]\)
- \([+\text{nadgrids} = <\text{list \ of \ grid \ shift \ files}>]\)

Examples:

- "+\text{datum} = \text{WGS84}"
- "+\text{towgs84} = -263.0, 6.0, 431.0 + \text{ellps} = \text{clark80}"
- "+\text{nadgrids} = \text{ntv1\_can.dat} + \text{ellps} = \text{clrk66}"  

Use "cs2cs -ld" to get a list of known datums.
**PROJ.4: Projection Parameters**

- +lon\_0 = <angle>
  - Central Meridian, Longitude of Origin, Center Long
- +lat\_0 = <angle>
  - Latitude of Origin, Center Latitude
- +k = <scale\_factor>
- +x\_0 = <false\_easting>
- +y\_0 = <false\_northing>

*Almost* all projections have +lon\_0, +x\_0, +y\_0.
Transverse Mercator

Aka Gauss-Kruger
+proj=tmerc +lon_0= <central meridian>
+lat_0= <latitude of origin> +k= <scale factor>
+x_0= <false easting> +y_0= <false northing>

Example (UTM 11 North):
+proj=tmerc +lon_0= -117 +lat_0= 0
+k= 0.9996
+x_0= 5000000 +y_0= 0
+datum= WGS84
**Lambert Conic Conformal (2SP)**

+ proj = lcc + lat_1 = \(1^{st}\) std. Parallel >
  + lat_2 = \(2^{nd}\) std. Parallel >
  + lat_0 = \(<\text{origin lat}\> + \text{lon}_0 = \(<\text{origin long}\> + x_0 = \(<\text{false easting}\> + y_0 = \(<\text{false northing}\>

**Example (Tennessee State Plane):**
+ proj = lcc + lat_1 = 35.25 + lat_2 = 36d25
  + lat_0 = 34d40 + lon_0 = -86
  + x_0 = 609601.2192024384
  + y_0 = 30480.06096012192
  + datum = NAD27 + units = ft
**Universal Transverse Mercator**

Aka UTM
\[+\text{proj}=\text{utm} + \text{zone}= < \text{zone}>>

Example (UTM zone in which Ottawa falls):
\[+\text{proj}=\text{utm} + \text{zone}=17 + \text{datum}=\text{WGS84}\]

Just an alias for:
\[+\text{proj}=\text{tmerc} + \text{lon}_0= -81 +k=0.9996 +\text{x}_0=500000 +\text{datum}=\text{WGS84}\]
**PROJ.4 Dictionaries**

- Common coordinate systems defined in dictionaries.
- **Format**: `+init=<dictionary>:<name>`
- **Example**: `+init=epsg:4326`
- Dictionaries are text files in `/usr/local/share/proj`
- Search them with a text editor!
- **Declarations look like**:
  
  ```
  # WGS 84
  <4326> +proj=longlat +datum=WGS84 +no_defs
  ```
Distributed Dictionaries:
• epsg: Definitions for EPSG GCS and PCS.
• nad27: State plane zones keyed on USGS zone#
• nad83: State plane zones keyed on USGS zone#
• esri: ESRI extended “EPSG” database
• other.extra: OGC WMS “EPSG” extensions
• world: assorted additional common projections
Open Geospatial Consortium “Well Known Text”

- OGC WKT is a “standard” for exchange of coordinate systems.
- Originally from Simple Features for SQL
- Variations used by ESRI “Projection Engine”, Oracle, AutoMap, Mapguide, GDAL/OGR and PostGIS
- Not to be confused with WKT geometries
OGR WKT Example

PROJCS["NAD27 / New York East",
    GEOGCS["NAD27",
        DATUM["North_American_Datum_1927",
            SPHEROID["Clarke 1866",6378206.4,294.9786982138982,
                AUTHORITY["EPSG","7008"]],
            AUTHORITY["EPSG","6267"]],
        PRIMEM["Greenwich",0,
            AUTHORITY["EPSG","8901"]],
        UNIT["degree",0.01745329251994328,
            AUTHORITY["EPSG","9122"]],
            AUTHORITY["EPSG","4267"]],
        PROJECTION["Transverse_Mercator"],
        PARAMETER["latitude_of_origin",40],
        PARAMETER["central_meridian",-74.33333333333333],
        PARAMETER["scale_factor",0.999966667],
        PARAMETER["false_easting",5000000],
        PARAMETER["false_northing",0],
        UNIT["US survey foot",0.3048006096012192,
            AUTHORITY["EPSG","9003"]],
        AUTHORITY["EPSG","32015"]]
Simplified OGR WKT Example

PROJCS["NAD27 / New York East",
    GEOGCS["NAD27",
        DATUM["North_American_Datum_1927",
            SPHEROID["Clarke 1866",6378206.4,294.9786982138982]],
        PRIMEM["Greenwich",0],
        UNIT["degree",0.01745329251994328]],
    PROJECTION["Transverse_Mercator"],
    PARAMETER["latitude_of_origin",40],
    PARAMETER["central_meridian",-74.33333333333333],
    PARAMETER["scale_factor",0.999966667],
    PARAMETER["false_easting",500000],
    PARAMETER["false_northing",0],
    UNIT["US survey foot",0.3048006096012192]]

- Striped down to “Simple Features” keywords.
-
ESRI WKT Example

PROJCS["NAD_1927_StatePlane_New_York_East_FIPS_3101",GEOGCS["GCS_North_American_1927",DATUM["D_North_American_1927",SPHEROID["Clarke_1866",6378206.4,294.9786982]],PRIMEM["Greenwich",0],UNIT["Degree",0.017453292519943295]],PROJECTION["Transverse_Mercator"],PARAMETER["False_Easting",500000],PARAMETER["False_Northing",0],PARAMETER["Central_Meridian",-74.33333333333333],PARAMETER["Scale_Factor",0.9999666666666667],PARAMETER["Latitude_Of_Origin",40],UNIT["Foot_US",0.30480060960121924]]

- ESRI .prj files are just one long line.
- ESRI uses very specific datum names
- ESRI has their own projection and parameter names in some cases
• Standard enumeration of widely used coordinate systems, datums, units, etc.
• Basis of the geotiff format.
• Used in WMS and many other web service requests.
• Used in many software packages
  eg. WGS84 is EPSG:4326
  UTM 11 North, WGS84 is EPSG:32611
Using EPSG

- Lookups can be tricky, I usually search the `/usr/local/share/gdal/pcs.csv` and `gcs.csv` files in a text editor!
- Sticky note: WGS84 (4326), NAD83 (4269), NAD27 (4267)
- If the code # is larger than 32767 then it isn't a real EPSG code
- Ask about WMS 1.3.0 and axis order later over a beer. Ugg.
PROJ.4 Command Usage

Command:
  cs2cs +proj=latlong +datum=WGS84
    +to +proj=utm +zone=11 +datum=WGS84

Input:
  -118.0 33.0

Output:
  406582.22  3651730.97  0.00
MapServer Projections

- Use PROJ.4 format:
  
  ```
  PROJECTION
  "+proj=utm +zone=11 +datum=WGS84"
  END
  ```

- Can also use PROJ.4 init files for epsg, etc
  
  ```
  PROJECTION
  "+init=epsg:4326"
  END
  ```

- Avoid using multiline format, or "+init=EPSG"
- No support for WKT
GRASS Projections

Usage:
g.proj [- pdjwefc] [georef= file] [wkt= file] [proj4= params] [location= name]

Flags:
- p  Print projection information (in conventional GRASS format)
- d  Verify datum information and print transformation parameters
- j  Print projection information in PROJ.4 format
- w  Print projection information in WKT format
- e  Use ESRI- style format (applies to WKT output only)
- f  Print 'flat' output with no linebreaks (applies to WKT and PROJ.4 output)
- c  Create new projection files (modifies current location unless 'location' option specified)
PostGIS

- Coordinates are referred to by SRID (Spatial Reference ID), a db-local identifier
- SRIDs often match EPSG codes for predefined
- SRIDs are related to proj.4 and WKT definitions in the spatial_ref_sys table
- WKT is just for user, not really used
PostGIS

- Register a newly created table:
  
  ```
  select AddGeometry('test', 'roads', 'geom', 4326, 'LINESTRING', 2);
  ```

- Create geometry with SRID:
  
  ```
  insert ... GeomFromEWKT(SRID=4326;POINT(5 7))...
  ```

- Reproject geometries: (ADD!)
epsg_tr.py - postgis 4326 4326
BEGIN;
---
--- EPSG 4326 : WGS 84
---
INSERT INTO "spatial_ref_sys"
("srid","auth_name","auth_srid","srtex","proj4text") VALUES
(4326,'EPSG',4326,'GEOGCS["WGS
84",DATUM["WGS_1984"],SPHEROID["WGS
84",6378137.298.257223563,AUTHORITY["EPSG","7030"],AUTHOR
RITY["EPSG","6326"],PRIMEM["Greenwich",0,AUTHORITY["EPSG","8901"],UNIT["degree",0.01745329251994328,AUTHORITY["EPSG","9122"],AUTHORITY["EPSG","4326"],'+ proj=longlat + ellps=WGS84 + datum=WGS84 + no_defs ');
COMMIT;
Gdal/ OGR

- Native format is OGC WKT (or OGR WKT!)
- Coordinate systems reported in WKT
- Includes transformation services to/from PROJ.4, ESRI WKT, and from EPSG.
- `-a_srs <srs>` to assign an SRS with gdal_translate and ogr2ogr
- `-t_srs` to select target SRS with gdalwarp and ogr2ogr commands to reproject
- Many formats of SRS supported
GDAL/ OGR

-a_srs EPSG:4326

-a_srs '+proj=utm +zone=11 +datum=WGS84'

-a_srs abc.wkt

-a_srs 'GEOCS...'

-a_srs ESRI::roads.prj
Datum Shifting – GridShift Files

- Uses a grid of offset values over region
- Gives best approximation of correction for irregular transformations
- Commonly used for NAD27 to NAD83
- PROJ.4 includes traditional US NAD27 to NAD83 files as well as Canadian NTv1
- Also supports Canadian NTv2 format now sometimes used in other countries
- Use +nadgrids= keyword.
- No explicit support in WKT.
Datum Shifting – 3/7 parameter

- 3 parameter – simple offset in 3 space
- 7 parameter – offset, rotate and scale
- Just an approximation
- Often different values in different regions for a single datum
- Often hard to find good values
- Use +towgs84= keyword
- TOWGS84[] in WKT
Datum Shifting – Examples

- **+ datum = WGS84** is
  + **ellps = WGS84 + towgs4 = 0,0,0**
- **+ datum = GRS87** is
  + **ellps = GRS80 + towgs84 = -199.87,74.79,246.62**
- **+ datum = NAD27** is
  + **ellps = clrk66 + nadgrids = @conus, @alaska, @ntv2_0.gsb, @ntv1_can.dat**
Gotchas

- PROJ.4 may default to WGS84 ellipsoid if not given, be explicit!
- Aea and lcc projections have default standard parallels for USA ... use +no_def.
- Longitude signs matter, Ottawa is west of greenwich which is a negative longitude.
- Alternate axis orientation not supported.
- Did you download grid shift files?
- False easting/northing always in meters.
- Europeans do +towgs84 signs backwards.
Tips

- Test a known point with command line tools.
- Use -v flag with cs2cs to see actual values used.
- Verify datum shift is doing something.
- Are grid shift files being found?
- Set PROJ_DEBUG environment variable to see files accessed.
- Don't trust the “epsg” dictionary, especially with regard to datum shifting and uncommon projections.
Conclusion

Details for PROJ.4 and WKT parameters for many common projections at:

www.remotesensing.org/geotiff/proj_list/

Questions?