
Subject: Any way to initialize mapping in IDL with WKT or a Proj.4 string?

Posted by [Gordon Farquharson](#) on Thu, 29 Sep 2016 22:20:42 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hi All

Can anybody think of a way that will allow me to initialize the IDL mapping routines with a Coordinate Reference System Well Known Text string [1] or a Proj.4 parameter string [2]. Examples include:

```
PROJCS["UTM Zone 10, Northern Hemisphere",GEOGCS["WGS
84",DATUM["WGS_1984",SPHEROID["WGS
84",6378137,298.257223563,AUTHORITY["EPSG","7030"]],TOWGS84[0,0,0,0,0,0,0],AUTHORIT
Y["EPSG","6326"]],PRIMEM["Greenwich",0,AUTHORITY["EPSG",
"8901"]],UNIT["degree",0.0174532925199433,AUTHORITY["EPSG
","9108"]],AUTHORITY["EPSG","4326"]],PROJECTION["Transverse_Mercator
"],PARAMETER["latitude_of_origin",0],PARAMETER["central_meridian
",-123],PARAMETER["scale_factor",0.9996],PARAMETER["false_easting
",500000],PARAMETER["false_northing",0],UNIT["Meter",1]]
```

and

```
+proj=utm +zone=10 +ellps=WGS84 +datum=WGS84 +units=m +no_defs
```

I want this process to work for any projections, i.e., not just UTM. I think the answer to my question lies in this last statement, namely that it is not possible, because the GCTP library does not implement all possible projections.

Sigh. Why didn't Harris/Exelis just listen to the god David Fanning, and convert IDL to use Proj.4 [3].

Probably time to write a script to parse Proj.4 strings for a few well-loved projections.

Gordon

[1] <http://docs.opengeospatial.org/is/12-063r5/12-063r5.html>

[2] <https://trac.osgeo.org/proj/wiki/GenParms>

[3] <https://groups.google.com/forum/#!topic/comp.lang.idl-pvwave/JrJmVhsYB8M>

Subject: Re: Any way to initialize mapping in IDL with WKT or a Proj.4 string?

Posted by [rawahranger](#) on Fri, 30 Sep 2016 04:22:43 GMT

[View Forum Message](#) <> [Reply to Message](#)

Gordon Farquharson writes:

> Sigh. Why didn't Harris/Exelis just listen to the god David Fanning, and convert IDL to use Proj.4 [3].

Sigh...

David

Subject: Re: Any way to initialize mapping in IDL with WKT or a Proj.4 string?

Posted by [Gordon Farquharson](#) on Fri, 30 Sep 2016 21:31:24 GMT

[View Forum Message](#) <> [Reply to Message](#)

OK, so here is a solution. It can be extended to deal with any projection IDL knows about. Thank you, Exelis/Harris for implementing the IDL-Python bridge. Very useful.

For this solution to work, you need to have the IDL-Python bridge working and have the Python API for GDAL/OGR installed.

```
FUNCTION map_proj_init_proj4, proj4_string
```

```
    compile_opt IDL2, LOGICAL_PREDICATE, STRICTARRSUBS
```

```
    osr = python.import('osgeo.osr')
```

```
    srs = osr.SpatialReference()
```

```
    IF osr.SpatialReference.ImportFromProj4(srs, proj4_string) NE 0 THEN BEGIN
        message, 'error importing from Proj.4 string'
```

```
    ENDIF
```

```
    CASE 1 OF
```

```
        proj4_string.matches('\+proj=utm'): BEGIN
```

```
            semimajor_axis = osr.SpatialReference.GetSemiMajor(srs)
```

```
            semiminor_axis = osr.SpatialReference.GetSemiMinor(srs)
```

```
            utm_zone = osr.SpatialReference.GetUTMZone(srs)
```

```
            map = map_proj_init(101, $
                                SEMIMAJOR_AXIS=semimajor_axis, $
                                SEMIMINOR_AXIS=semiminor_axis, $
                                ZONE=utm_zone)
```

```
        END
```

```
    ELSE: BEGIN
```

```
        message, 'projection not recognized'
```

```
    END
```

```
    ENDCASE
```

```
    return, map
```

```
END
```

Usage:

```
IDL> map_struct = map_proj_init_proj4('+proj=utm +zone=10 +ellps=WGS84 +datum=WGS84
+units=m +no_defs')
IDL> print, map_proj_forward(-122, 42, MAP=map_struct), FORMAT='(2(F20.9))'
582818.069248419 4650259.847758290
```

(The UTM/GCTP bug [1] seems to be fixed in IDL 8.5.1.)

Gordon

[1] http://www.idlcoyote.com/map_tips/utmwrong.php

Subject: Re: Any way to initialize mapping in IDL with WKT or a Proj.4 string?
Posted by [Gordon Farquharson](#) on Mon, 17 Oct 2016 02:41:48 GMT
[View Forum Message](#) <> [Reply to Message](#)

So, to follow up with this thread (for the approximately two people in the world who may be interested), below is an expanded IDL function that shows a case where the idea I proposed breaks with IDL. For the Equirectangular projection (#117 in IDL), the GCTP library allows only a sphere to be used for the ellipsoid, whereas Proj4 permits the use of any ellipsoid:

In Python (Equirectangular projection with the WGS84 ellipsoid):

```
>>> import osgeo.osr
>>> srs = osgeo.osr.SpatialReference()
>>> srs.ImportFromProj4('+proj=eqc +lat_ts=0 +ellps=WGS84')
0
>>> srs.ExportToWkt()
'PROJCS["unnamed",GEOGCS["WGS
84",DATUM["unknown",SPHEROID["WGS84",6378137,298.257223563]],PRIMEM[
"Greenwich",0],UNIT["degree",0.0174532925199433]],PROJECTION[
"Equirectangular"],PARAMETER["latitude_of_origin",0],PARAMETER[
"central_meridian",0],PARAMETER["false_easting",0],PARAMETER["false_northing",0]]'
```

This case is not supported by the GCTP library in IDL.

Here is the expanded IDL function that deals with this case:

```
FUNCTION map_proj_init_proj4, proj4_string, WKT=wkt

  compile_opt IDL2, LOGICAL_PREDICATE, STRICTARRSUBS

  osr = python.import('osgeo.osr')
  srs = osr.SpatialReference()
  IF osr.SpatialReference.ImportFromProj4(srs, proj4_string) NE 0 THEN BEGIN
    message, 'error importing from Proj.4 string'
```

ENDIF

wkt = osr.SpatialReference.ExportToWKT(srs)

CASE 1 OF

```
proj4_string.matches('\+proj=utm'): BEGIN
  semimajor_axis = osr.SpatialReference.GetSemiMajor(srs)
  semiminor_axis = osr.SpatialReference.GetSemiMinor(srs)
  utm_zone = osr.SpatialReference.GetUTMZone(srs)
  map = map_proj_init(101, $
    SEMIMAJOR_AXIS=semimajor_axis, $
    SEMIMINOR_AXIS=semiminor_axis, $
    ZONE=utm_zone)
```

END

```
proj4_string.matches('\+proj=eqc'): BEGIN
  IF osr.SpatialReference.GetInvFlattening(srs) NE 0 THEN BEGIN
    message, 'invalid combination: GCTP does not support a non-spherical ellipsoid with
the Equirectangular projection'
  ENDIF
  sphere_radius = osr.SpatialReference.GetSemiMajor(srs)
  center_longitude = osr.SpatialReference.GetProjParm(srs, 'central_meridian')
  true_scale_latitude = osr.SpatialReference.GetProjParm(srs, 'standard_parallel_1')
  false_easting = osr.SpatialReference.GetProjParm(srs, 'false_easting')
  false_northing = osr.SpatialReference.GetProjParm(srs, 'false_northing')
  map = map_proj_init(117, $
    SPHERE_RADIUS=sphere_radius, $
    CENTER_LONGITUDE=center_longitude, $
    TRUE_SCALE_LATITUDE=true_scale_latitude, $
    FALSE_EASTING=false_easting, $
    FALSE_NORTHING=false_northing)
```

END

```
ELSE: BEGIN
  message, 'projection not supported by this function (yet)'
END
```

ENDCASE

return, map

END

This example highlights the thought in my original message - namely, one is never going to be able to initialize the IDL mapping structure correctly from an arbitrary Proj4 string, because the Proj4 library supports far more projections than are supported in the GCTP library. However, for GCTP-supported projections, the the idea in the function above is useful.

