OctPROJ
Calling PROJ.4 from GNU Octave*

José Luis García Pallero†

February 6, 2010

Abstract

This is a small introduction to using the OctPROJ package. In this text, you can overview the basic usage of the functions in GNU Octave[1]. If you need a detailed description about the options and available projections, please visit the PROJ.4 website[2].

1 Overview

OctPROJ allows you to perform cartographic projections in GNU Octave using PROJ.4 library. You can take the power of PROJ.4 using the facilities that GNU Octave provides, without know the internals of the PROJ.4 C API. You can use the conversion programs coming with PROJ.4 distribution, but for use them in GNU Octave you must make system calls. With OctPROJ, PROJ.4 can be integrated in GNU Octave scripts in a natural way.

2 Installation

As several GNU Octave packages, OctPROJ installation consists in compiling the C++ kernel sources (see section 3), link them against GNU Octave library to generate *.oct functions and copy this *.oct executables and other *.m functions into a working directory.

The automatic procedure can be easily done by running the command:

```
octave:1> pkg install octproj-x.x.x.tar.gz
```

where x.x.x is the version number.

After that, the functions and documentation are installed in your machine and you are ready for use the package.

†Instituto de Astronomía y Geodesia, Fac. de CC Matemáticas, Universidad Complutense de Madrid. jlgpallero@pdi.ucm.es, jgpallero@gmail.com

[1] http://www.octave.org

[2] http://trac.osgeo.org/proj

3 Kernel wrapper

The main functions (the functions which make the actual computations) are programmed in a separate files: `projwrap.h` and `projwrap.c`.

The files contain three functions:

- `proj_fwd`: forward computation of geodetic to projected coordinates.
- `proj_inv`: inverse computation of projected to geodetic coordinates.
- `proj_transform`: general transformations. Is possible to make forward, inverse and other transformations using only one function (see `PROJ.4` documentation).

3.1 Error handling

Error handling in the kernel wrapper is based on error codes from `PROJ.4`. Functions in `projwrap` return the `PROJ.4` error code and the `PROJ.4` text string error message, which can be catched in order to work in this case.

The functions can stop the execution in presence of errors depending on the nature of the error.

- If exist an error involving a general parameter of the projection, the execution stops.
- If an error is found due to a wrong or out of domain input coordinate, the execution don't stops, but the error code is activated and the output transformed coordinate corresponding to the error position have a special value.

4 GNU Octave functions

Two types of functions are programmed for GNU Octave: `*.oct` functions and `*.m` functions.

4.1 `*.oct` functions

These functions are linked with `projwrap`. You can use it, but is no recommended because the input arguments are more strict (always column vectors) than `*.m` functions and don't check for errors.

The functions are:

- `_op_geod2geoc`: transformation between geodetic and cartesian geocentric coordinates. This function calls directly `PROJ.4`.
- `_op_geoc2geod`: transformation between cartesian geocentric and geodetic coordinates. This function calls directly `PROJ.4`.
- `_op_fwd`: forward projection (calls `proj_fwd`).
- `_op_inv`: inverse projection (calls `proj_inv`).
- `_op_transform`: general transformation (calls `proj_transform`).

4.2 `*.m` functions

These functions make the computations by calling the `*.oct` functions. You must call these functions because you can use various types of input (scalars, vectors or matrices) and checking of input arguments (data type, dimensions) is performed.

The functions are the same as in section 4.1 (without the _ at the beginning of the name):

- `op_geod2geoc`: calls `_op_geod2geoc`.
- `op_geoc2geod`: calls `_op_geoc2geod`.
- `op_fwd`: calls `_op_fwd`.
- `op_inv`: calls `_op_inv`.
- `op_transform`: calls `_op_transform`.

4.3 Error handling

`*.oct` and `*.m` functions can emit errors or warnings, some due to errors in input arguments and other due to errors in functions from `projwrap` kernel execution (see section 3).

Errors due to wrong input arguments (data types, dimensions, etc.) can be only given for `*.m` functions and this is the reason because the use of these functions are recommended. In this case, the execution is aborted and nothing is stored in output arguments.

Errors due to the execution of `projwrap` kernel can be emitted for both `*.oct` and `*.m` functions. If the error is due to an erroneous projection parameter, the execution is aborted and nothing is stored in output arguments; but if the error is due to a wrong or out of domain input coordinate, a warning is emitted and the execution has a normal end.

5 Examples

5.1 Geodetic to geocentric and vice versa

```
lon=-6*pi/180;lat=43*pi/180;h=1000;
[x,y,z]=op_geod2geoc(lon,lat,h,6378388,1/297)
x =  4647300.72326257
y = -488450.988568138
z =  4328259.36425774

[lon,lat,h]=op_geoc2geod(x,y,z,6378388,1/297);
lon*=180/pi,lat*=180/pi,h
lon = -6
lat =  43
h =  1000.00000000074
```

5.2 Forward and inverse projection

```
lon=-6*pi/180;lat=43*pi/180;
[x,y]=op_fwd(lon,lat,'+proj=utm +lon_0=3w +ellps=GRS80')
x =  255466.980547577
y =  4765182.93268401

[lon,lat]=op_inv(x,y,'+proj=utm +lon_0=3w +ellps=GRS80');
lon*=180/pi,lat*=180/pi
lon = -6.00000000003597
lat =  42.9999999999424
```

5.3 Forward and inverse projection: op_transform

5.3.1 With altitude

```
lon=-6*pi/180;lat=43*pi/180;h=1000;
[x,y,h]=op_transform(lon,lat,h,'+proj=latlong +ellps=GRS80',...
                     '+proj=utm +lon_0=3w +ellps=GRS80')
x =  255466.980547577
y =  4765182.93268401
h =  1000

[lon,lat,h]=op_transform(x,y,h,...
                         '+proj=utm +lon_0=3w +ellps=GRS80',...
                         '+proj=latlong +ellps=GRS80');
lon*=180/pi,lat*=180/pi,h
lon = -6.00000000003597
lat =  42.9999999999424
h =  1000
```

5.3.2 Without altitude

```
lon=-6*pi/180;lat=43*pi/180;
[x,y]=op_transform(lon,lat,'+proj=latlong +ellps=GRS80',...
                   '+proj=utm +lon_0=3w +ellps=GRS80')
x =  255466.980547577
y =  4765182.93268401

[lon,lat]=op_transform(x,y,'+proj=utm +lon_0=3w +ellps=GRS80',...
                       '+proj=latlong +ellps=GRS80');
lon*=180/pi,lat*=180/pi
lon = -6.00000000003597
lat =  42.9999999999424
```

5.4 Error due to an erroneous parameter

```
lon=-6*pi/180;lat=43*pi/180;
[x,y]=op_fwd(lon,lat,'+proj=utm +lon_0=3w +ellps=GRS8')
error:
        In function op_fwd:
```

```
In function _op_fwd:
Projection parameters
unknown elliptical parameter name
+proj=utm +lon_0=3w +ellps=GRS8
```

5.5 Error due to latitude too big

```
lon=[-6*pi/180;-6*pi/180];lat=[43*pi/180;43];
[x,y]=op_fwd(lon,lat,'+proj=utm +lon_0=3w +ellps=GRS80')
warning: _op_fwd:

warning: Projection error in point 2 (index starts at 1)
x =

   255466.980547577
                Inf

y =

   4765182.93268401
                Inf
```

References

[1] Eaton, John W.; Bateman, David, and Hauberg, Søren; GNU Octave. A high-level interactive language for numerical computations; Edition 3 for Octave version 3.2.3; July 2007; Permanently updated at http://www.gnu.org/software/octave/docs.html.

[2] Evenden, Gerald I.; Cartographic Projection Procedures for the UNIX Environment—A User's Manual; USGS Open-File Report 90-284; 2003; ftp://ftp.remotesensing.org/proj/OF90-284.pdf.

[3] Evenden, Gerald I.; Cartographic Projection Procedures Release 4, Interim Report; 2003; ftp://ftp.remotesensing.org/proj/proj.4.3.pdf.

[4] Evenden, Gerald I.; Cartographic Projection Procedures Release 4, Second Interim Report; 2003; ftp://ftp.remotesensing.org/proj/proj.4.3.I2.pdf.

[5] Snyder, John Parr; Map Projections: A Working Manual; USGS series, Professional Paper 1395; Geological Survey (U. S.), 1987; http://pubs.er.usgs.gov/usgspubs/pp/pp1395.