Line.Sta Algorithms

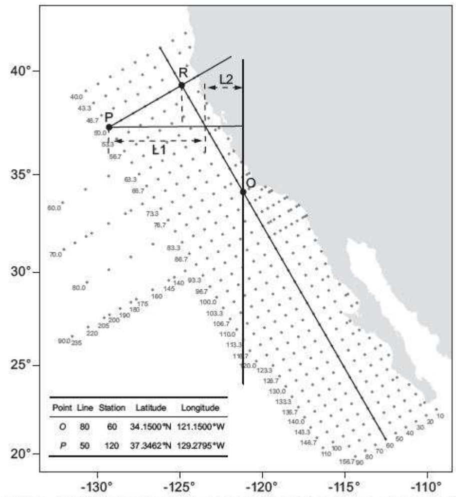
Parent Category: References (/references.html)
Category: Methods (/about-calcofi/methods.html)
C Last Updated: 02 September 2014

Corrected Conversion Algorithm for CalCOFI Station Grid And Their Implementation in Several Computer Languages (http://calcofi.org/publications/calcofireports/v54/Vol_54_Weber_97-106.pdf) by Edward Weber & Thomas Moorem, NOAA Fisheries, 2013 CalCOFI Report v 54.

Link to downloadable code (Java, Perl, Python & R) file presented in the CalCOFI Report vol 54, pages 102-106. (http://calcofi.org/downloads/software/LineSta/CalCOFICoordCode.zip)

UPDATED **Matlab scripts** (http://calcofi.org/files/code/CalCOFILineStationMatLab.zip) **by Robert Thombley, SIO-CalCOFI with Sep2014 error correction by Augusto Valencia, UABC**; based on Weber & Moor 2013.

Link to MatLab downloadable code authored by Robert Thombley, SIO-CalCOFI (http://calcofi.org/files/code/CalCOFILineStationMatLab.zip)



Point	Line	Station	Latitude	Longitude
O	80	60	34.1500°N	121.1500°W
P	50	120	37.3462°N	129.2795°W

Figure 1. CalCOFI station plan and schematic representation of the geometrical components used for CalCOFI coordinate conversions (after Eber and Hewitt 1979). Point *O* is located at line 80, station 60, 34.15°N, 121.15°W, and is the point about which the CalCOFI sampling grid is rotated –30° off of the meridian. Point *P* is located at line 50, station 120, 129.2795°W, 37.3462°N, and is used as an example for calculations. Reference point *R*, and line segments *L1*, and *L2* are intermediate products used in trigonometric calculations (equations 4–8 and 12–14).

(Abstract from pdf) "Converting between geographic coordinates in latitude and longitude and the line and station sampling pattern of the California Cooperative Fisheries Investigations (CalCOFI) program is a commonly required task for conducting research on the California Current ecosystem. This note presents several corrections and clarifications to the previously published algorithms for performing these conversions. We include computer code to implement the algorithms in Java™1, Perl, Python, and R. We note that freely available code to conduct the conversions in Fortran, Matlab®2, JavaScript™, and Visual Basic®6 has previously been published, and an online conversion tool is also available. A future version of the PROJ.4 cartographic projections library will also include support for CalCOFI conversions, thereby allowing for convenient conversions using the GRASS GIS, PostGIS, Python, Perl, R, and many other programs and programming languages."

For individual language - Java, Perl, Python, or R - implementations & downloads, see sections below

Java Implementation (**Java Download (http://calcofi.org/downloads/software/LineSta/CalCOFICoordCodeJava.zip)**): objects of the Java class CalcofiCoordConverter hold coordinates as either “longlat” or “calcofi.” Coordinates may be retrieved using the “getCoords” method. The current coordinate system may be retrieved using the “getProjection” method. The “reproject” method accepts either “longlat” or “calcofi ” as an argument and converts to the appropriate coordinate system.

To Convert Lat Lon to Line Sta

```
>java TestCalcofiCoordConverter -121.15 34.15 longlat
returns:
Original coordinates
x -121.15
y 34.15
projection longlat
New coordinates
x 79.99999886102664
y 60.0000059182792
projection calcofi
Backtransformed coordinates
x -121.15000054796204
y 34.15
projection longlat
```

To Convert Line Sta to Lat Lon:

```
> java TestCalcofiCoordConverter 50 120 calcofi
Original coordinates
x 50.0
y 120.0
projection calcofi
New coordinates
x -129.2795443042271
y 37.34615242270663
projection longlat
Backtransformed coordinates
x 49.999998777973516
y 120.00000634983587
projection calcofi
```

PERL Implementation (Perl Download (http://calcofi.org/downloads/software/LineSta/CalCOFICoordCodePerl.zip)): the Perl code is designed to read input data from a file (coords.in) and write the converted output coordinates (along with the original input coordinate data) to a file (coords.out). The section of the code that begins with “USER-MODIFIED SETTINGS” can be changed to use different filenames and identify the direction of the conversion (“cc2geo” for CalCOFI station grid coordinates to geographic coordinates and “geo2cc” for geographic coordinates to CalCOFI station grid coordinates).

Contents for **cc2geo_coords.in**:

```
LINE,STATION
80.0,60.0
50.0,120.0
```

Contents for **cc2geo_coords.out**:

```
LINE,STATION,LAT_DD,LON_DD
80.0,60.0,34.1500,-121.1500
50.0,120.0,37.3462,-129.2795
```

Contents for **geo2cc_coords.in:**

```
LATD,LATM,LATS,LATH,LOND,LONM,LONS,LONH
34,9,0,N,121,9,0,W
34.15,0,0,N,121.15,0,0,W
37,20.7692,0,N,129,16.7727,0,W
```

Contents for **geo2cc_coords.out:**

```
LATD,LATM,LATS,LATH,LOND,LONM,LONS,LONH,LAT_DD,
LON_DD,LINE,STATION
34,9,0,N,121,9,0,W,34.15,-121.15,80.0000,60.0000
34.15,0,0,N,121.15,0,0,W,34.15,-121.15,80.0000,60.0000
37,20.7692,0,N,129,16.7727,0,W,37.3461533333333,
-129.279545,50.0000,120.0000
```

Python Implementation (Python Download (http://calcofi.org/downloads/software/LineSta/CalCOFICoordCodePython.zip)): the Python functions perform the conversions on single coordinate pairs as x and y arguments, or multiple points by entering tuples, lists, or arrays from the numeric python module numpy. The functions always return numpy arrays.

```
> line, station = latlontostation(-121.15, 34.15)
> line
79.999998861026384
> station
60.000005918280479
> stationtolatlon(line, station)
array([-121.15000055, 34.15 ])

> lon, lat = stationtolatlon(50, 120)
> lon
-129.27954430422696
> lat
37.346152422706631
>latlontostation(lon, lat)
array([ 49.99999878, 120.00000635])
```

Multiple points may be converted equivalently using tuples:

```
> stationtolatlon((80, 50), (60, 120))
```

lists:

```
> stationtolatlon([80, 50], [60, 120])
```

or numpy arrays:

```
> arr = numpy.array((80, 60, 50, 120))
> arr.shape = (2, 2)
> arr
array([[ 80, 60],
[ 50, 120]])
> stationtolatlon(arr)
```

All three of these examples return:

```
array([[-121.15 , 34.15 ],
[-129.2795443 , 37.34615242]])
```

R Implementation (R Download (http://calcofi.org/downloads/software/LineSta/CalCOFICoordCodeR.zip)): the R functions listed in Appendix IV work similar.

```
> lineandstation <- latlon.to.station(c(-121.15,34.15))
> lineandstation
line station
[1,] 80 60.0000

> station.to.latlon(lineandstation)
lon lat
line -121.15 34.1
```

```
> latlon <- station.to.latlon(c(50, 120))
> latlon
lon lat
[1,] -129.2795 37.3461

> latlon.to.station(latlon)
line station
lon 50 120
```

The R functions will accept matrices of two columns to perform conversions on multiple points at the same time. For example,

```
> mat <- matrix(c(80, 50, 60, 120), 2, 2)
> mat
[,1] [,2]
[1,] 80 60
[2,] 50 120
> station.to.latlon(mat)
lon lat
[1,] -121.1500 34.15000
[2,] -129.2795 37.34615
```

By default, R uses fewer significant digits than Python or Java but this could be adjusted in the R options

MatLab Scripts (MatLab Download (http://calcofi.org/files/code/CalCOFILineStationMatLab.zip)) by Robert Thombley; gridlib.m

function [lat, lon] = cc2lat(li,st) DESCRIPTION: Use this function to convert from calcofi grid coordinates to latitude and longitude. This uses a slightly modified (because the original is wrong) version of the CalCOFI gridding algorithm (Eber and Hewitt 1979) (Also see errata 2006). INPUT: The calcofi line and station values OUTPUT: This function outputs a degree decimal latitude and longitude ASSUMPTIONS: All lat/long and station values are from the Northwestern hemisphere and within the middle latitudes. REFERENCE: Based on Weber & Moore 2013; Eber and Hewitt 1979, Conversion Algorithms for the CalCOFI Station Grid WRITTEN BY: Robert Thombley (2006), Scripps Institution of Oceanography, SIO-CalCOFI MODIFIED BY: Augusto Valencia (2014), Universidad de Baja California-UABC (02Sep2014).	function [line, sta] = lat2cc(la,lo) DESCRIPTION: Use this function to convert from degree decimal lat and long to calcofi grid coordinates. This uses the CalCOFI de-gridding algorithm (Eber and Hewitt 1979). This is the inverse of the function cc2lat. It is more complicated because this one requires some non-algebraic iteration. INPUT: latitude and longitude values in decimal degrees to be converted OUTPUT: This function outputs a line and station value ASSUMPTIONS: All lat/long and station values are from the Northwestern hemisphere and within the middle latitudes. REFERENCE: Based on Weber & Moore 2013; Eber and Hewitt 1979, Conversion Algorithms for the CalCOFI Station Grid WRITTEN BY: Robert Thombley (2006), Scripps Institution of Oceanography, SIO-CalCOFI MODIFIED BY: Augusto Valencia (2014), Universidad de Baja California-UABC (02Sep2014).