
Subject: Re: Met Data spatial interpolation
Posted by [dan](#) on Tue, 06 Dec 1994 19:04:42 GMT
[View Forum Message](#) <> [Reply to Message](#)

In article <1994Dec1.115119.8178@news.dkrz.de>, m211058@regen.dkrz.de (Wolfgang Knorr) writes:

> I was wondering whether anyone would have a program in IDL that can
> do any kind of spatial interpolation of meteorological data, either
> for filling gaps in gridded data sets, or to produce grid maps from
> station data.

> If nobody knows of any such routine, I will write one myself
> and make it available to anyone interested. (Are meteorologists
> using IDL at all?)

>
> Cheers to all.

>
> |-----|
> | Wolfgang Knorr Phone : +49 40 41173 214 |
> | Max-Planck-Institut f. Meteorologie Fax : +49 40 41173 298 |
> | Bundesstr. 55 |
> | 20146 Hamburg |
GERMANY knorr@dkrz.d400.de

>

I've written a routine which interpolates scattered data onto a regular latitude longitude grid. For each grid point where an interpolated value is desired, this routine weights the N (user defined) closest data points by great circle distance raised to a power (power is user defined, default is -1 which is an inverse distance weighting). Here it is. If you have questions, feel free to post or to email me.

; \$ID\$

;+

; Name:

; INTERP_SPHERE

;

; PURPOSE:

; This function maps scattered data defined by
; (longitude,latitude,value) onto a regular, but not
; necessarily evenly spaced, grid whose coordinates are
; also defined by longitude and latitude. The procedure searches
; for the N (default = 5) closest data points to each grid
; point and then averages these N data points weighted by
; distance^power from the grid point to the particular data point.
; Default is power=-1 which weights the points inversely by
; distance. All distances are along great circles on a sphere

```

; (the shortest distance between two points along the
; surface of a sphere).
;
; CATEGORY:
;   Interpolation?
;
; CALLING SEQUENCE:
;   grid = INTERP_SPHERE(lat,lon,data)
;
; INPUTS:
;
;   lat:   The latitudes on the grid where interpolated
;          values are desired (in degrees)
;
;   lon:   The longitudes on the grid where interpolated
;          values are desired (in degrees)
;
;   data:  An array (3,ndata) where ndata is the number of
;          data points, and can be any number larger than N.
;          each row of data should contain a longitude, a
;          latitude, and a value to be interpolated.
;
; KEYWORD PARAMETERS:
;
;   n:     The number of closest data points to be used
;          for each grid point interpolation. Default = 5
;
;   power: The exponent for the distance weighting function.
;          Default = -1 (weighting inversely by distance).
;          An input of power=-.5 would weight inversely by the
;          square root of the distance.
;
; OUTPUTS:
;
;   grid:  An array of interpolated data values. It has dimensions
;          (nlon,nlat) where nlon is the number of entries in the
;          input lon, and nlat is the number of entries in the input
;          lat.
;
; EXAMPLE:
;
; MODIFICATION HISTORY:
;
;   written by:  Dan Bergmann dbergmann@llnl.gov 11/10/94
;-

```

FUNCTION INTERP_SPHERE,lat,lon,data,n=n,power=power

```

nlat = (size(lat))(1)
nlon = (size(lon))(1)
grid = fltarr(nlon,nlat)

if (not(keyword_set(n))) then n = 5
if (not(keyword_set(power))) then power = -1

dtr = !pi / 180.

; convert lat and lon to radians

latr = dtr * lat
lonr = dtr * lon

; convert the lat and lon of the data to radians

dlatr = dtr * data(1,*)
dlonr = dtr * data(0,*)

; calculate the cartesian coordinates of the data points
; assuming a unit sphere.

xdata = cos(dlatr) * sin(dlonr)
ydata = cos(dlatr) * cos(dlonr)
zdata = sin(dlatr)

for x=0,nlon-1 do begin

    sinlonr = sin(lonr(x))
    coslonr = cos(lonr(x))

    for y=0,nlat-1 do begin

; calculate the cartesian coordinates of this particular
; grid point.

        xorig = cos(latr(y)) * sinlonr
        yorig = cos(latr(y)) * coslonr
        zorig = sin(latr(y))

; calculate the length squared of the cords connecting this grid
; point to all the data points and then sort the data points by
; these values.

        corddistsq = (xorig-xdata)^2+(yorig-ydata)^2+(zorig-zdata)^2

        sortdist = (sort(corddistsq))(0:n-1)

```

```

; if a data point lies directly on top of this grid point, then
; assign that value to the grid point.
; Otherwise calculate the n great circle distances and do a weighted
; average of the data values.

if ((corddistsq(sortdist))(0) eq 0) then begin

    grid(x,y) = data(2,(sortdist)(0))

endif else begin

    grcirdis = asin(sqrt(corddistsq(sortdist))/2.)

    grid(x,y) = (total(data(2,sortdist) * grcirdis^power)) / total(grcirdis^power)

endelse

endfor

return,grid

end

```

```

--
*****
** Dan Bergmann                dbergmann@llnl.gov **
** Global Climate Research      fax (510) 422-5844 **
** Lawrence Livermore National Lab human (510) 423-6765 **

```
