Subject: Re: Interpolation/gridding on a sphere?
Posted by sterner on Wed, 12 Jul 1995 07:00:00 GMT
View Forum Message <> Reply to Message

tmote@unlinfo.unl.edu (thomas mote) writes:

> I have a need to interpolate and grid climate data for the northern
> hemisphere. I wish to know if anyone has IDL code to perform
> interpolation across the surface of a sphere. Can the routines in the
> IDL user's library be modified to accomplish this?

You might be in luck if you have IDL Version 4. Here is an extract
from the online help:

SPH_SCAT
The SPH_SCAT function performs spherical gridding. Scattered samples on
the surface of a sphere are interpolated to a regular grid. This routine
is a convenient interface to the spherical gridding and interpolation
provided by TRIANGULATE and TRIGRID. The returned value of the function
is a regularly-interpolated grid.

I have not tried this yet.

Ray Sterner sterner@tesla.jhuapl.edu
The Johns Hopkins University North latitude 39.16 degrees.
Applied Physics Laboratory West longitude 76.90 degrees.
Laurel, MD 20723-6099
WWW Home page: ftp://fermi.jhuapl.edu/www/s1r/people/res/res.html

Subject: Re: Interpolation/gridding on a sphere?
Posted by dan on Wed, 12 Jul 1995 07:00:00 GMT
View Forum Message <> Reply to Message

In article <3tv5le$agd@crcnis3.unl.edu>, tmote@unlinfo.unl.edu (thomas mote) writes:
|> I have a need to interpolate and grid climate data for the northern
|> hemisphere. I wish to know if anyone has IDL code to perform
|> interpolation across the surface of a sphere. Can the routines in the
|> IDL user's library be modified to accomplish this?
|>
|> Thanks in advance.
|>
|> Thomas L. Mote
|> tmote@unlinfo.unl.edu
|>
|>
|>

I have a computationaly (is that a word?) intensive routine which weights grid points by inverse distance to each data point. Great circle distances on a sphere are used. I'll include the routine here. IDL 4.0 is supposed to have something better, but I haven't tried it out yet. Here is my routine ..

```
; $ID$

;+
; Name:
;       INTERP_SPHERE
;
; PURPOSE:
;       This function maps scattered data defined by
;       (longitude,latitude,value) onto a regular, but not
;       neccessarily 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.
;
;       latwt:   The weighting for the interpolation in the meridional
;               (North-South) direction. For negative power,
;               latwt > 1 produces a weighting with less latitude
;               influence. Default = 1
;
;       mask:    Mask for calculating grid values
;
;
; 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,latwt=latwt

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
if (not(keyword_set(latwt))) then latwt = 1
if (not(keyword_set(mask)))  then begin
  mask = intarr(nlon,nlat)
  mask(*,*) = 1
endif

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

;   check to see if this grid should be calculated

    if (mask(x,y) ne 0) then 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)*latwt)^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

  endif

 endfor

endfor

return,grid

end
```

--
 ** ***

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