
Subject: Re: Extract / interpolate in 3d atmospheric field
Posted by [Kenneth P. Bowman](#) on Wed, 03 Feb 2010 15:10:53 GMT
[View Forum Message](#) <> [Reply to Message](#)

In article

<f46dd771-47cd-44b6-805e-098d66c5ada1@c29g2000yqd.googlegroups.com>,
Dave <confused.scientist@gmail.com> wrote:

```
>  
> IDL> print, data.xlat[70]  
>      85.0000  
> IDL> help, reform(data.data[70,*,*])  
> <Expression>  FLOAT    = Array[96, 25]  
>  
> Unfortunately, the latitude circle of interest isn't always exactly  
> specified, say 83.5 or 84 deg N.  
> Re-running the model with higher grid resolution is not an option so  
> I'm stuck trying to interpolate  
> the data cube. I suspect I'm not the first person to encounter this in  
> atmospheric science and  
> was wondering if anyone had any routines or techniques they are  
> willing to share.  
>  
> I've looked at the routine 'extract_slice' but I'm not sure it is  
> appropriate here and I've not been  
> able to get sensible results from it.  
>  
> Thanks all,  
> Dave
```

A little work with INTERPOLATE will get you what you need,
assuming you are satisfied with tri-linear interpolation.

The function below will interpolate global gridded data
(NCEP reanalysis in this case) to an arbitrary list of
(x, y, z) points. These could be the grid points in your slice.
The x, y, and z arrays should have the same number of elements.

I store variables like NCEP data inside a structure, where data
is the structure, data.values is the 3-D data array, data.x is
the longitude coordinate, data.x.delta is the longitude grid spacing,
etc. x.delta and y.delta are scalars; z.delta is a 1-D array.

You can pass those quantities in as separate arguments if you want,
but I like the compactness of using structures.

This function assumes regular spacing in x and y (lon and lat) and
irregular spacing in z. It also handles the periodic boundary condition

in the longitude direction.

You might also want to look at the interpolation chapter in my book

http://csrp.tamu.edu/pdf/idl/sample_chapter.pdf

Ken Bowman

```
FUNCTION NCEP_P_INTRP_XYZ, data, x, y, z
```

```
;  
;+  
; Name:  
;   NCEP_P_INTRP_XYZ  
; Purpose:  
;   This procedure spatially interpolates the 3-D NCEP gridded data to the  
;   position(s) (x, y, z). The variables x, y, and z can be scalars or three  
;   arrays of the same size. Interpolation is done with the built-in  
;   function INTERPOLATE using trilinear interpolation. No time interpolation  
;   is done.  
;  
;   Use this function to interpolate one or more values from gridded data at a  
;   single time.  
; Category:  
;   NCEP data utility.  
; Calling sequence:  
;   f = NCEP_P_INTRP_XYZ(data, x, y, z)  
; Inputs:  
;   data : structure containing NCEP data  
;   x   : longitude(s)  
;   y   : latitude(s)  
;   z   : pressures(s)  
; Output:  
;   Scalar or array of data values interpolated to requested location(s).  
; Keywords:  
;   None.  
; Author:  
;   K. Bowman. 2005-11-06.  
;-
```

```
COMPILE_OPT IDL2                                ;Set compile options
```

```
values = [data.values, data.values[0,*,*]]        ;Copy first longitude slice
```

```
dxinv = 1.0/data.x.delta                          ;Inverse of delta-x  
dyinv = 1.0/data.y.delta                          ;Inverse of delta-y  
dzinv = 1.0/data.z.delta                          ;Inverse of delta-z
```

```

xi    = dxinv*x                                ;Longitude interpolation coordinate(s)
yi    = dyinv*(90.0 + y)                       ;Latitude interpolation coordinate(s)
k      = 0 > VALUE_LOCATE(data.z.values, z) < (data.z.n - 2) ;Find interval(s) containing z
value(s)
zi    = k + dzinv[k]*(z - data.z.values[k])      ;Altitude interpolation coordinate(s)

RETURN, INTERPOLATE(values, xi, yi, zi)          ;Return interpolated data

END

```
