Subject: Re: four corners for lat & long pixels
Posted by James Kuyper on Thu, 05 Jul 2007 16:26:31 GMT
View Forum Message <> Reply to Message

titan wrote:
> thank you for your answer but unfortunately i'm trying this procedure
> for the first time!! :o(
> could you please be more clear?
>
> thank you very much

Let's assume that 'latitude' is an array with dimensions (cols, rows). Most of the work that needs to be done can be done with the following commands:

```
ix = DINDGEN(cols+1)-0.5
iy = DINDGEN(rows+1)-0.5
corner_lat = BILINEAR(latitude, ix, iy)
```

Note: ix and iy have now been replaced by two-dimensional arrays. Unfortunately, when it goes outside of the bounds of the input array, BILINEAR switches over to a nearest-neighbor rather than using bilinear extrapolation. IDL has many routines for performing various kinds of interpolation, but a quick survey didn't turn up a single one that switched over to extrapolation beyond the boundaries of the original data. Therefore, you'll have to fix up all four edges. I'll show you how to handle the first row; the technique for the other three edges are similar:

```
top = latitude[0:cols-2,0:1]+latitude[1:*,0:1]
corner_lat[1:cols-1,0] = (2.0*top[*,0] - top[*,1])/6.0
```

Finally, special handling is also needed for the four outermost corners. For the first row and column, bilinear extrapolation to the corner point is equivalent to:

```
corner_lat[0,0] = 0.5625*latitude[0,0]-.
1875*(latitude[1,0]+latitude[0,1])+0.1875*latitude[1,1]
```

Similar calculations apply at the other three corners.

You shouldn't worry about the more sophisticated approaches I mentioned earlier, until you understand this simple approach well enough.