Subject: Re: Is it possible to speed up the Interpolate command?
Posted by Craig Markwardt on Fri, 08 Aug 2014 02:44:29 GMT
View Forum Message <> Reply to Message

On Thursday, August 7, 2014 6:11:29 PM UTC-4, Stephen Messenger wrote:
> I've been stuck on figuring out how to speed up an interpolation calculation and wondered if anyone has any suggestions?
>
>
>
> Here's the situation:
>
>
>
> I have a bunch (about 450,000) of 2d matrices that I need to interpolate within. Within each of the matrices, I'm looking to interpolate for 100 x/y combinations where I want values at points (x_1, y_1), (x_2, y_2), etc. (I am not looking to regrid the data, i.e., I don't need x_1,y_2). The matrices are currently stacked in a datacube (dimensions are 14 x 28 x 450000). Each of the matrices has the same x/y locations for the points to be interpolated. I thereby use "interpolate" to interpolate each matrix for the 100 values and then loop over the 3rd dimension. This utilizes the bilinear interpolation. Though, I have the matrices stacked in the data cube, I do not want a trilinear interpolation as the 3 dimension is independent. Here's the current code:
>
>
>
> for i=0, numlines-1 do begin
>
> Values(*,i)=interpolate(datacube(*,*,i),x_loc,y_loc)
>
> endfor
>
>
>
> Numlines is the n_elements(3rd dimension), which is the 450,000 referenced above. The x and y dimensions of the data cube are 14 and 28, respectively.
>
>
>
> The interpolation is taking about 2 seconds to run. I'm looking to find a way to trim it as much as possible...hopefully less than 0.1 seconds. This may be difficult given that the interpolation is calculating 45,000,000 values.
>
>
>
> Things I've tried:
>
> 1) I first removed the interpolation from the for loop. However, the combination of that interpolation with reforming the output result into the matrix I need requires this process to actually

take longer than the for loop above...this provides evidence the existence of the for loop is not the rate limiting step.
>
>
>
> 2) I rearranged the datacube into a very large 2d matrix (basically stacking in the 2nd dimension as opposed to creating the 3rd dimension). This lead to the same calculation time as the original way above, so no gains there.
>
>
>
> I need a bilinear interpolation due to the first two dimensions being linked so interpol will not work. I do not need regridded data, so I don't think that Krig2d or bilinear offer any help.
>
>
>
> I know that I can speed up the code by simply decreasing the size of the 3rd dimension and/or by interpolating for less than 100 values per matrix, but I'm trying to avoid this.

Turn your problem around. Right now you are interpolating 100 points in FOR loop with 450000 iterations. Instead, you should be doing a FOR loop of 100 iterations for 450000 points each. It will be much faster.

Bilinear interpolation involves some simple mathematics with linear weighting factors. See the wikipedia page. You don't need to use INTERPOLATE().

Something like this should do the trick...

```
for i = 0, 99 do begin
  x_loci = x_loc[i]
  y_loci = y_loc[i]

  i0 = floor(x_loci) & i1 = i0+1
  j0 = floor(y_loci) & j1 = j0+1

  u = x_loci - i0
  v = y_loci - j0
  values(i,*) = datacube(i0,j0,*)*(1-u)*(1-v) + datacube(i1,j0,*)*u*(1-v) + $
     datacube(i0,j1)*(1-u)*v + datacube(i1,j1)*u*v
endfor
```

Oh by the way if REFORM() is limiting you, then you should probably be using REFORM(...,/OVERWRITE).

Craig