Subject: Re: 2D interpolation with sparse data
Posted by cmancone on Wed, 23 May 2007 18:57:51 GMT
View Forum Message <> Reply to Message

Interesting solution. There's only one thing I'd be worried about if I were you, and that is: how sensitive is the result to your scaling factor? Do you have to change the scaling factor depending on how "bright" your data is? If the scaling factor is low enough, do you get non-x contributions? If you haven't already, I would make sure the answers to those questions are to your satisfaction.

On May 23, 10:42 am, Ken G <kagoldb...@gmail.com> wrote:
> Thank you both for the suggestions!
>
> I did try a strictly-x line-by-line interpolation and that certainly
> does work (and quickly) to fill in the missing points. Yet, a problem
> might occur when there's a non-x-dependence to the original data. But
> I came up with a different idea.
>
> I was originally been using the TRIANGULATE -> TRIGRID path to solve
> this interpolation and fill-in the missing data. TRIANGULATE computes
> the 'triangles' connectivity based on the simple distance between
> points, which means we can trick it to believe that points in x or y
> are closer or farther apart. I simply added a constant multiplier to
> the y positions in the input arguments to the TRIANGULATE procedure
> and it completely changes the triangulation being used. I got the idea
> when I looked at the matrix formulation in this article
> http://en.wikipedia.org/wiki/Delaunay_triangulation
>
> So the triangulate call looks like this:
> TRIANGULATE, x, y*scaling_factor, triangles, boundary
>
> and the whole interpolation looks more or less like this (pared down)
>
> w = where(img NE 0)
> wx = w mod Nx
> wy = w / Nx
> scaling = 4L
> TRIANGULATE, wx, wy*scaling, triangles, boundary
> img2 = TRIGRID(wx,wy,img(w), triangles, [1,1], [0,0,Nx-1,Ny-1], Nx=Nx,
> Ny=Ny)