Subject: Re: FFT's and images: or My 4th day on what I thought would take me only 6hrs
Posted by Craig Markwardt on Fri, 17 Aug 2001 05:08:59 GMT
View Forum Message <> Reply to Message

tbowers0@yahoo.com writes:

> Ok, 4 days is quite enough time to spend hacking over a problem before
> I post. So here goes...
> *All* I'm tryin' to do is MTF (modulation transfer function) blur an
> image then restore it with:
>
> blurredImage = fft(fft(originalImage,-1) * fft(pntSpreadFn,-1) ,1)
>
> then reverse it for check...
>
> reconstructedImage = fft(fft(blurredImage,-1) / fft(pntSpreadFn,-1)
> ,1)
>
> 1) How, oh HOW in the world do I create a *general* 2D gaussian to use
> as my PSF to start off with. I.e, if my image is [sizeX,sizeY] and
> sizeX does not necessarily equal sizeY, how do I create a symmetric 2D
> gaussian?? The best I can do is a 2D gaussian with (sizeX eq sizeY)
> always. Images are rarely the same in x and y.

Todd, you have asked a lot more questions than I think I can answer.
But, to get you started, how about this 100x50 2-d exponential
function.

```
nx = 100  & sx = 20   ;; Second number is gaussian sigma value
ny = 50   & sy = 20
psf = exp(-(findgen(nx)-nx/2.+0.5)^2/sx^2) # exp(-(findgen(ny)-ny/2.+0.5)^2/sy^2)
```

I'll leave you to stew on that one for a while, but the key is the "#"
operator which is the exterior matrix product operator. However, I
think you will need to modify this a little bit. I believe that the
2D FFT needs the centroid of the gaussian to be at the [0,0] pixel,
which means that you don't need the 0.5's that appear above, and that
you should shift the image by (-nx/2 ,-ny/2). I think.

> 2) Could anyone also give my the scoop on how to take any 1D psf(r or
> psi) and create a 2D psf from it? E.g. I have psf for all angles:
> angle=indgen(0,181)
> psf=[p0,p1,p2,...,p180]

Good question. This is really more of an interpolation question

though: given the measured value of the psf on the spokes of a wheel, interpolate into the pies between the spokes. You could probably go through, pixel by pixel in an array and then compute (r,theta). After that you could go to your list of psf values, find the pair of spokes that bracket the pixel, and perform bilinear interpolation. In fact it might be vectorizable, but start simple grasshopper. :-)

> 4) Lastly, if the psf has 0's in it anywhere, won't this screw up my
> division when I try to reconstruct the image? I ask because in my
> futile efforts at creating a gaussian psf I often manifest some at
> the edges. Of course, when I try to add a very small offset to
> where(psf eq 0.0), e.g. 1e-10, just to eliminate 'em, I sometimes get
> more strange results.

This is your real problem here Todd. You simply can't go backwards from your blurred image. You have removed information by blurring your image (ie taken away the high frequencies). When you try to restore it you probably found that any remaining high frequencies, ie noise, were amplified greatly.

This is probably a dead end, so don't spend too much more time on it, if all you are trying to do is verify if you got the equation right.
Why not blur some known functions like say, a delta function, some horizontal and vertical lines, etc.

Good luck,
Craig

--

---------- ----------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
---------- ----------