Subject: Re: I need a bit of help....Convol and functions
Posted by James Kuyper on Tue, 03 Oct 2006 23:17:32 GMT
View Forum Message <> Reply to Message

D.Kochman@gmail.com wrote:
> Hey you guys, once again thanks for the help. I'm still having fun
> learning IDL. ;)
>
> So heres my next problem is finding out how this was implemented so I
> can replicate the procedure in order to do my own convolution. Pretty
> much what this following function is is a multiple convolution. It
> inputs a system response file (confun) which is 1024 data points.
>
>
> FUNCTION func, P, X=x, Y=y, ERR=err, CONFUN=confun, $
> funcvals=f,binsizefactor=bsf
>
> fshift=floor(-P(4)/bsf)
> rshift=-P(4)/bsf-fshift
> fconfun=shift(confun,-fshift)
> sconfun=(1-rshift)*fconfun+rshift*SHIFT(fconfun,-1)
> sumex=P(0)*exp(-X / P(1))+P(2)*exp(-X / P(3))+P(7)*exp(-X / P(8))

I would strongly recommend use of the square bracket notation, such as
P[2], for array indices. The parenthesis notation, while legal, is
easily confused with a function call syntax. The results when you
mistype an array name get very confusing.

> blank=dblarr(1024)
> consum = convol([blank,sumex],sconfun,total(sconfun),CENTER=0)
> consumex=consum[1024:*]
> scatter= P(6)*sconfun
> f=P(5)+consumex+scatter
> err=sqrt(f)
> return,(y-f)/(err)
> end
>
> So my big question is what is going on in convol?

I'd recommend going to the online help for CONVOL(). It's very explicit
about what precisely is going on.

> If I'm guessing
> correctly it seems like this is creating an array [blank,sumex] in
> essence to force an array upon sumex - but its a function.

Yes, func() needs to create an array for CONVOL() to convolve. However,
sumex should already be an array, in order for this code to work as

intended, and this code does nothing to change any aspect of sumex. I'm not sure I understand what you mean by the comment "but its a function".

> Are
> specific data points being used in sumex (ie x=0->1024)?

Yes. Every single data point in sumex is being used to create an element in the temporary array which is created by the expression [blank,sumex]. Every single element in that temporary array contributes to the calculation of the consum array returned by CONVOL().

> ... I thought
> convol only took two parameters, which would mean sconfun would be the
> kernel, but I can't see that thats an array either?

sconfun should be an array, in order for this code to work correctly. It will be an array, if confun is an array. In other words, it's up to the caller of func() to make sure that it is.

> ... What is then the
> purpose of total(sconfun)?

The third argument to CONVOL() is an optional scale factor, which defaults to 1. The convolution is implemented as a sum which is divided by that factor. It's not very important for floating point values: CONVOL(array, kernel, S) is pretty much equivalent to CONVOL(array/S, kernal) or CONVOL(array, kernel/S) or CONVOL(array,kernel)/S; which one you use is your choice. However, for integer and byte arrays,

CONVOL(array, [[1,1],[1,1]], 4)

is faster than

CONVOL(array, [[0.25, 0.25],[0.25, 0.25]])

because it avoids the floating point conversion, and it is coded in such a way as to protected against the overflow problems that would occur if you tried

CONVOL(array, [[1,1],[1,1]])/4

Note: the /NORMALIZE option sets the scale factor to TOTAL(ABS(kernel)), which is often the most appropriate value, and is probably more appropriate than the value of TOTAL(sconfun) used in the code you're looking at. It is also recommended if you use the /INVALID or /NAN options, since it will scale the results only by the total over the kernel elements actually used.