Subject: Re: The Behavior of CONVOL
Posted by Kevin R. Turpie on Fri, 01 Nov 1996 08:00:00 GMT
View Forum Message <> Reply to Message

Ken Kump wrote:
>
>> First, CONVOL does not appear to perform a convolution by default;
>> rather it seems to do a correlation. They are similar, but give
>> different results if the kernel is asymmetric.
>
> No, the values are not normalized in any way. You need to be
> aware of truncation and padd accordingly.
>

I'm afraid you missed the point; normalization doesn't have anything
to do with it. It is the orientation of the kernel. In a convolution
the kernel "flipped" wrt to the function being convolved. In a
correlation is is not. With CENTER=1 (default), CONVOL uses the same
orientation as a correlation. With CENTER=0, it does not.

Consider the following 1-D example:

a = [0., 0., 1., 0., 0.]

b = [.4, .5, .1]

print, convol(a,b,/center),form='(9(F4.2,2x))' ; /center is default

0.00 0.10 0.50 0.40 0.00

print, convol(a,b,center=0),form='(9(F4.2,2x))'

0.00 0.00 0.40 0.50 0.10

Pretend this is an optics problem where _b_ is a point-spread
function and _a_ is a 1-D image. 40% of the energy of each pixel
is dumped into the pixel to the left and 10% to the right. The
result should be

0.00 0.40 0.50 0.10 0.00

That clearly doesn't happen in either application of CONVOL.

>> Second, when CENTER is set to 0, CONVOL does a convolution in a
>> strict sense *if* the input kernel function, say k(x), is defined
>> so that k(x) = 0 for all x < 0. The result is usually shifted to
>> the right.
>> To do a true convolution with CONVOL for any kernel, it seems that

>> CENTER must be set to 1 and REVERSE must be applied to each dimension
>> of the kernel prior to input.
>
> Yes, this is true. I like to **always** perform a strict
> convolution. I set center=0, reverse my convolution
> kernel in each direction, and padd zeros on
> either side of my input function, then after the convolution,
> I may truncate them to be repositioned correctly. I find this
> annoying and computationally inefficient. For a convolution
> of some magnitude (I think Numerical Recipies gave a number for
> a 1-D case of 60) Fourier convolution is **much** faster.

Yes, setting CENTER=0 and shifting "left" for each dimension is also
a valid workaround. Consider also this generalized example:

for i=1,(SIZE(b))(0) do b = REVERSE(b, i)
c = CONVOL(a, b, /EDGE_TRUNCATE).

If we have to workaround then I think this will work for you, and
with less effort. Yes, yes, I also use FFT to convolve things
like images and maps; the implementation is not difficult. But,
along with the benefits, there are caveats and limitations there
too.

But now I don't understand your point. Are you saying CONVOL is fine
as long as you do this, this, and that? My concern is that CONVOL
doesn't seem to behave conventionally and requires workarounds to
do common applications. I believe users should be made fully aware
of this before any serious mistakes are made (or worse, published).