Subject: Re: Does CONVOL convolute
Posted by Thomas Gutzler on Tue, 25 Feb 2003 01:26:07 GMT
View Forum Message <> Reply to Message

Big Bird wrote:
>
> Maybe I'm thinking something completely wrong here somewhere but if my
> array is
>
> 1 0 0
> 0 0 0 ...
> 0 0 0
> .
> .
>
> and my convolution kernel is
>
> a b c
> d e f
> g h i
>
> then I would expect the convolution to be
>
> e f 0
> h i 0 ...
> 0 0 0
> .
> .
>
> At least for a symmetric kernel (I'd have to think about an
> unsymmetric one). I expect that because the fourier transform of a
> delta function is a constant (one) which is thus the multiplicative
> neutral element. And the fourier transform of a convolution of
> functions is the multiplication of the transforms of the functions
> themselves.
>
> The longer I stare at the help, the more confused I get trying to
> figure out whether the [0,0] element of the result should be zero, as
> it seems to have a condition attached that reads (at least on my
> screen with the font the help uses) "if t >= 1-1 and u >= 1-1" . And I
> really don't think either of these "ones" could be 'lower-case ell'
> even though they use an 'ell' in the sum but without apparent
> motivation (or at least they don't seem to say what 'ell' *is*).
>
> /edge_truncate goes in the vague direction
>
> /edge_truncate,center=0 gives me something I don't understand at all
>

> /edge_wrap does what it sounds like (which is not what I would
> consider useful
> for most mathematical applications)
>
> /edge_wrap,center=0 comes closest to what I would have expected
>
> This is all rather mysterious to me - is the term "convolution" used
> differently in engineering than in math? Clearly I have to think about
> this some more ...

Maybe a bound mirror expansion combined with shrinking helps ?

```
IDL> a = [[1,0,0,0],[0,0,0,0],[0,0,0,0],[0,0,0,0]]
IDL> b = [[10,11,12],[13,14,15],[16,17,18]]
IDL> print, boundmirrorshrink(convol(boundmirrorexpand(a),b))
      14      13       0       0
      11      10       0       0
       0       0       0       0
       0       0       0       0
```

not exactly what you want, but closer, isn't it ? :)

```
FUNCTION BoundMirrorExpand, A
 ; Expand the matrix using mirror boundary condition
 ;
 ; A = [ 1  2  3  11
 ;       4  5  6  12
 ;       7  8  9  13 ]
 ;
 ; B = BoundMirrorExpand(A) =
 ;     [ 5  4  5  6  12  6
 ;       2  1  2  3  11  3
 ;       5  4  5  6  12  6
 ;       8  7  8  9  13  9
 ;       5  4  5  6  12  6 ]

 m = (size(A))[1]
 n = (size(A))[2]
 B = [A, A[0,*], A[0,*]]
 B = [[B], [B[*,0]], [B[*,0]]] ; B and A have same type
 B(1:m,1:n) = A
 B(0,0) = B(2,2)   ; mirror corners
 B(0,n+1) = B(2,n-1)
 B(m+1,0) = B(m-1,2)
 B(m+1,n+1) = B(m-1,n-1)
 B(1:m,0) = B(1:m,2)  ; mirror left and right boundary
 B(1:m,n+1) = B(1:m,n-1)
 B(0,1:n) = B(2,1:n)  ; mirror top and bottom boundary
```

```
 B(m+1,1:n) = B(m-1,1:n)
 RETURN, B
END

FUNCTION BoundMirrorShrink, A
 ; Shrink the matrix to remove the padded mirror boundaries
 ; for example
 ;
 ; A = [ 5  4  5  6  12  6
 ;       2  1  2  3  11  3
 ;       5  4  5  6  12  6
 ;       8  7  8  9  13  9
 ;       5  4  5  6  12  6 ]
 ;
 ; B = BoundMirrorShrink(A) =
 ;     [ 1  2  3  11
 ;       4  5  6  12
 ;       7  8  9  13 ]

 m = (size(A))[1]
 n = (size(A))[2]
 B = A(1:m-2,1:n-2)
 RETURN, B
END

just my 2 ce^H^Hfunctions,
   Tom
```