Subject: Bug in IDL CONVOL procedure
Posted by Ulrik Kjems on Fri, 14 Jul 1995 07:00:00 GMT
View Forum Message <> Reply to Message

There seems to be a (several) bugs in the CONVOL library routine.

IDL> a=[0,1,1,1,0]
IDL> b=[1,1,1]
IDL> print,convol(a,b)
      0   2   3   2   0
I believe, that the result should be
      1   2   3   2   1
according to the reference manual p. 1-55.

Further, when convolving in 3D:
IDL> a=replicate(1.0,5,5,5)
IDL> b=replicate(1.0,3,3,3)
IDL> c=convol(a,b,/edge_wrap)
IDL> print,c

Subject: Re: Bug in IDL CONVOL procedure
Posted by chase on Fri, 14 Jul 1995 07:00:00 GMT
View Forum Message <> Reply to Message

>>>> > "Ulrik" == Ulrik Kjems <kjems> writes:
In article <3u5rn6$kuf@news.uni-c.dk> Ulrik Kjems <kjems> writes:

Ulrik> There seems to be a (several) bugs in the CONVOL library routine.
IDL> a=[0,1,1,1,0]
IDL> b=[1,1,1]
IDL> print,convol(a,b)
Ulrik>   0   2   3   2   0
Ulrik> I believe, that the result should be
Ulrik>   1   2   3   2   1
Ulrik> according to the reference manual p. 1-55.

This is not a bug. It behaves exactly as the indicated in my manual.
Whenever the kernel extends beyond the array boundaries a 0 is the
result. Your manual must be different than my mine (Version 3.5,
November 1993 Edition) where convol is defined on page 1-53. As of
version 3.6 (?) convol can take two other keywords that effect its
behavior. From the release notes in file notes/rel_note.doc:

4/28/94
  Enhancements to CONVOL and CONTOUR:

Added to the CONVOL function the keywords: EDGE_WRAP and EDGE_TRUNCATE, which control how points on the edge are handled. The convolution formula, for the CENTER = 0 case, is and was:
R(i) = Sum over j=0,nk-1 of A(i-j) * K(j),
where A is the array, K is the convolution kernal, and nk is the number of points in the kernal. The default, as before, is to set R(i)=0 for the nk-1 points on the edge, i=0, ..., nk-2. If EDGE_WRAP is set, the array subscripts wrap around:
R(i) = Sum over j=0,nk-1 of A((i-j) mod na) * K(j),
where na is the number of elements in A. If EDGE_TRUNCATE is set, the array elements along the edge are repeated:
R(i) = Sum over j=0,nk-1 of A((i-j) > 0 < (na-1)) * K(j).
The multi-dimensional and centered cases are handled similarly.

You could use /edge_truncate to get your desired result. However, if your array is not zero padded, you might get results with /edge_truncate that are different than what you desire. The problem is how to define the convolution output at a particular array position when the kernel overlaid at that position does not lie completely within the array boundaries.

Chris
--
==========
Bldg 24-E188
The Applied Physics Laboratory
The Johns Hopkins University
Laurel, MD 20723-6099
(301)953-6000 x8529
chris.chase@jhuapl.edu