Subject: Re: Segfault when smoothing image
Posted by wlandsman on Mon, 15 Feb 2010 17:45:37 GMT
View Forum Message <> Reply to Message

On Feb 15, 6:05 am, thoeger <lusepus...@gmail.com> wrote:

> The code line I have used is:
>
> imgdata2 = filter_image(imgdata, FWHM_GAUSSIAN=300, /
> ALL_PIXELS)

> In real IDL, it simply stalls indefinitely after "%
> Compiled module: GAUSSIAN."

(filter_image.pro is from the IDL Astronomy Library
http://idlastro.gsfc.nasa.gov/ftp/pro/image/filter_image.pro)

Let's look at the number of arithmetic operations you are asking
for. By default, FILTER_IMAGE uses a kernel 3 times the size of the
FWHM, which in your case would be a 900 x 900 array. The /
ALL_PIXELS option reflects pads the main array to avoid edge effects,
so your initial 2200 x 2200 array will become a 3100 x 3100 array.
The number of multiplications for the convolution is then
900.^2 * 3100.^2 = 7.8e12

As Jeremy mentioned, one probably wouldn't use a direct convolution
for a problem like this. By default, FILTER_IMAGE computes the
convolution as a product of Fourier transforms, but it appears that
this is still too slow. A faster method is to approximate the
Gaussian convolution using iterated 3 x 3 SMOOTHing.

b = filter_image(a,smooth=300,/iterate)

(This will automatically take care of edge effects.)

There is a nice tutorial on the different smoothing at
http://www.jhlabs.com/ip/blurring.html (though it is in Java and they
call it blurring).

--Wayne

> On Feb 14, 6:51 pm, Gianguido Cianci <gianguido.cia...@gmail.com>
> wrote:
>

>> What is your input for all this, a bunch of x,y coords? And you want a
>> certain value at each coord in a "fake" 2200x2200 iamge?
>> Could you post examples of input and especially the code you are
>> using?
>
>> --Gianguido
>
>> On Feb 14, 4:54 am, thoeger <lusepus...@gmail.com> wrote:
>
>>> Hello newsgfroup;
>
>>> I hope this question isn't too basic.
>
>>> As part of my master thesis in astronomy, I have to make an image
>>> consisting of 2200x2200 pixels having the value zero except certain
>>> pixels, representing each the center of tan astronomical objects,
>>> having the value one. The goal is to get an idea of the number density
>>> of objects in the field, so I try to do a gaussian smoothing using
>>> the filter_image function, but end up with a segfault and IDL
>>> quitting due to floating underflow. (To be precise, this is GDL on my
>>> laptop. True IDL on the university computers just stalls
>>> indefinitely). So it seems I'm doing something wrong here. Does anyone
>>> have an idea how to implement the smoothing, if not by filter_image?
>
>