Subject: Re: strange behaviour of bytscl by large arrays
Posted by Lajos Foldy on Mon, 23 Apr 2012 16:14:21 GMT
View Forum Message <> Reply to Message

On Monday, April 23, 2012 4:07:00 PM UTC+2, Klemen wrote:
> Hi folks,
>
> is there any explanation of why I don't get the same or at least similar results using the code below by:
> a) using DINDGEN in line 3
> b) using FINDGEN in line 3
>
> pro test
>   s = 10000
>   a = sin(findgen(s, s)/100000.)
>   b = bytscl(a)
>   write_tiff, 'b.tif', b
> end
>
> The tif file I get using the DINDGEN function has waves all over the image. The option using FINDGEN produces strange results (a couple of waves and then wide bands of constant values). See the following link for the (resized) results.
> https://picasaweb.google.com/112572300011512591455/Eumetsat# 5734593216558178098
>
> I came across this problem as I tried to scale (using HIST_EQUAL and BYTSCL functions) 16-bit 5-band RapidEye data to 24-bit RGB image. Scaling the whole image produced results that were all black, smaller subsets seemed ok.
>
> Does anybody have a suggestion how to handle this issue?
>
> Cheers, Klemen

I think IDL's FINDGEN() implementation is wrong: it uses a float counter instead of an integer one. The following test shows the difference:

```
pro test
cpu, tpool_nthreads=1
n=10l^8
nn=n-1
a1=findgen(n)                  ; real FINDGEN()
a2=fltarr(n)
count=0.0
for j=0l, nn do a2[j]=count++   ; IDL's implementation
a3=fltarr(n)
count=0ll
for j=0l, nn do a3[j]=count++   ; better implementation
print, a1[nn], a2[nn], a3[nn], format='(3F15.3)'
end
```

(Multithreading must be disabled because the starting values for the threads are calculated as an integer. So the result of FINDGEN() depends on the number of your CPU cores, too :-)

regards,
Lajos