Subject: histogram produces extra bin in 64-bit IDL 8.0
Posted by Giuseppe Papa on Fri, 13 May 2011 13:45:11 GMT
View Forum Message <> Reply to Message

Consider the following code:
IDL> D=randomu(32,3200);
IDL> N=histogram(D,min=0.,max=1.,binsize=0.1,Locations=X)

In IDL 32-bit:
IDL> print, size(N)
          1   10   3   10
IDL> print, X
 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9
IDL> print, !version
{ x86 linux unix linux 8.0 Jun 18 2010   32   64}

In 64-bit IDL:
IDL> print, size(N)
          1   11   3   11
IDL> print, X
 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0
IDL> print, !version
{ x86_64 linux unix linux 8.0 Jun 18 2010   64   64}

As you can see, the 64-bit version produces an extra erroneous bin, contrary to the what is expected from the help pages for histogram.

Subject: Re: histogram produces extra bin in 64-bit IDL 8.0
Posted by Dick Jackson on Wed, 18 May 2011 15:56:10 GMT
View Forum Message <> Reply to Message

On May 18, 8:04 am, andry <andry_will...@hotmail.com> wrote:
> I just posted the following message but then I saw there just was a
> topic about more or less the similar problem. In my case, it is just
> about using different binsize and max.
>
> Can somebody explain why the following gives different number of
> elements with the HISTOGRAM function?
>
> IDL> hist_spd= HISTOGRAM(velocity[indb],binsize= .20,min=0.,max=1.40)
> IDL> print,hist_spd
>   6   16   23   0   0
> 0   0
> IDL> hist_spd= HISTOGRAM(velocity[indb],binsize= 20.,min=0.,max=140.)
> IDL> print,hist_spd
>   45   0   0   0   0
> 0   0   0

>
> (whatever velocity content is)
> My main concern is the number of elements returned by the function.
> The only difference between the 2 command is that I multiplied the
> binsize and max with 100. BUT they return 7 and 8 elements?
>
> I would expect they both return the same number of elements.

I will bet that floating point precision has something to do with it:

```
IDL> print,0.2*7 EQ 1.4
   1
IDL> print,0.2+0.2+0.2+0.2+0.2+0.2+0.2 EQ 1.4
   0
IDL> print,0.2+0.2+0.2+0.2+0.2+0.2+0.2
      1.40000
IDL> print,0.2+0.2+0.2+0.2+0.2+0.2+0.2, format='(F10.8)'
1.40000010
IDL> print,0.2*7, format='(F10.8)'
1.39999998
IDL> print,1.4, format='(F10.8)'
1.39999998
```

A fine write-up is here:
http://www.idlcoyote.com/math_tips/sky_is_falling.html

Cheers,
-Dick