Subject: Re: where function not finding value
Posted by Craig Markwardt on Wed, 11 Mar 2015 14:06:43 GMT
View Forum Message <> Reply to Message

On Wednesday, March 11, 2015 at 5:16:47 AM UTC-4, Jahvasc wrote:
> Thank you all for your replies.
>
> As I told you, I am aware of the problems caused by floating point precision and I also know that is not related to IDL itself. However, what has astonished me was the fact that different functions or ways to address the problem in IDL produced such discrepant results.
>
> Because of your help, I realised that the problem was in the way I was defining the intervals. In fact, they were too large. Then, I re-wrote the program introducing an eps=1.e-3 value. The loop turned-out to be:
>

```
>  mmod=[findgen(10)*0.01+0.01,findgen(3)*0.1+0.2]
>  eps=1.d-3
>  for j=0,12 do begin
>    a=where(massa ge mmod[j]-eps and massa le mmod[j]+eps,count)
>    print,mmod(j),count
>  endfor
```

>
> Now I get the right counts.
> Thank you again!

Jacqueline, from your description of the problem, your samples are right at the edges of the bins. I guess you've found a way to do this, but it doesn't look pretty and using a WHERE() function inside a FOR loop, well, it's kind of wierd.

But one will always get strange results if one tries to histogram/bin data when the sample values fall exactly on bin edges. (believe me, I goofed on this for data from a multi-million dollar space mission)

There are lots of ways to solve this, but they all rely on moving the bin edges away from your data samples. These techniques will always work, not do not rely on having special knowledge of how the data is quantized.

```
;; With HISTOGRAM (bin edges start at 0.005)
hh = histogram(massa, min=0.005, max=0.405, binsize=0.010)

;; I guess these are where your values are sampled, right?
mmod = [0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1, 0.2, 0.3, 0.4]
;; Then put the edges just to the left, plus one edge on the right hand side
edges = [sample_vals - 0.005, 0.5]

;; With HISTOGRAM and VALUE_LOCATE
hh = histogram(value_locate(edges, massa), min=0, max=12)
```

```
;; With a loop, no WHERE
for i = 0, 12 do begin
  hh[i] = total(massa GE edges[i] AND massa LT edges[i+1])
endfor

;; OK, you really want a loop with WHERE?
for i = 0, 12 do begin
  wh = where(massa GE edges[i] AND massa LT edges[i+1], count)
  hh[i] = count
  print, mmod[i], hh[i]
endfor
```

The expression,

```
  massa GE edges[i] AND massa LT edges[i+1]
```

does two things. Since EDGES[i+1] will be used as the right hand edge of the i-th bin, and then used again as the left edge of the (i+1)-th bin, one can never miss counts. There are *never* any holes due to floating point arithmetic. By the way, VALUE_LOCATE() does all this behind the scenes, that's what it was designed to do.

Also, look at the strategic use of GE on one bin edge, and LT on another bin edge. This guarantees that one sample can't fall into two bins, i.e. prevents double-counting.

Craig