Subject: Re: Box-Whisker plots in IDL
Posted by JD Smith on Mon, 27 Aug 2007 01:50:45 GMT
View Forum Message <> Reply to Message

On Mon, 20 Aug 2007 23:04:14 +0000, jschwab@gmail.com wrote:

> Pardon me if I'm mistaken, but I think these "quartiles with
> histogram" examples, including the one that's in JD's histogram
> tutorial are fundamentally incorrect.
>
> You are assuming "Equal bin widths" ==> "Equal #'s in each bin" !

I probably shouldn't have called them "quartiles", as they are really
quarter range bins (data quartiles). The data quartile is of course
as useful as the ordered quartile, but not for this problem.

One straightforward option for the ordered quartile is to use SORT,
picking out only the elements at nel/4 and 3*nel/4, e.g.:

```
 n=n_elements(data)
 s=sort(data)
 qval=data[s[3*n/4]]
```

Unfortunately, this is fairly slow for large data sets. Another
faster but approximate option is to form a cumulative total of
HISTOGRAM's output with an appropriate bin size, and find where it
reaches 25% and 75% of the total count of data points.

Depending on your needs and bin width, you may want to dive into the
individual bin using REVERSE_INDICES to find the *exact* quartile
value itself. This isn't as hard as it sounds:

```
 bs=0.05 ;something appropriate for bin size
 h=histogram(data,REVERSE_INDICES=r,BINSIZE=bs,OMIN=om)
 cum=total(h,/CUMULATIVE,/PRESERVE_TYPE)
 quart=3*n/4
 v=value_locate(cum,quart)
 vals=data[r[r[v+1]:r[v+2]-1]]
 qval=vals[(sort(vals))[quart-cum[v]]]
```

You'll find this is roughly 10x faster than using SORT by itself. And
if you only need the approximate value (good to the histogram bin
width), simply replace the last two lines with:

```
  qval=om+bs*(v+1.5)
```

for a modest additional speed-up. All the usual caveats with
HISTOGRAM apply (e.g. beware when dealt overly sparse data).

This problem reminds me of the one quote I always remember from
Numerical Recipes: "Selection is Sorting's austere sister."

JD