Subject: Re: Speed-up of code
Posted by JDS on Tue, 25 Aug 2009 22:30:48 GMT
View Forum Message <> Reply to Message

On Aug 25, 12:12 pm, Philip Elson <philipel...@googlemail.com> wrote:
>> h = HISTOGRAM(day, REVERSE_INDICES=ri)
>> nh = N_ELEMENTS(h)
>> sortData = value[ri[nh+1:*]]
>> totSortData = [0., TOTAL(sortData, /CUMULATIVE)]
>> vec8 = totSortData[ri[1:nh]-nh-1] - $
>> totSortData[ri[0:nh-1]-nh-1]
>> print,vec8/h ;May need to check for zero values first
>
> Aha! Thank you all very much!
>
> If I was to extend this further, any suggestions on the obtaining the
> maximum rather than the average of each bin?
>
> This would indeed provide some nice examples of gathering basic
> statistics using the histogram function!

The "dual histogram" method is useful for this. I'd use it like this:

```
h=histogram(day,REVERSE_INDICES=ri)
mx=fltarr(n_elements(h))
h2=histogram(h,OMIN=om,REVERSE_INDICES=ri2)
for k=long(om eq 0),n_elements(h2)-1 do begin
  if h2[k] eq 0 then continue
  bins=ri2[ri2[k]:ri2[k+1]-1]  ;bins of the histogram with this
repeat count

  if k+om eq 1 then begin
    mx[bins]=value[ri[ri[bins]]]
  endif else begin
    targ=[h2[k],k+om]
    points=ri[ri[rebin(bins,targ)]+rebin(lindgen(1,k+om),targ)]
    mx[bins]=max(value[points],DIMENSION=2)
  endelse
endfor
```

You'll note this does use a loop, but it's a loop over the number of
different bin occupation counts, i.e. a very small loop typically,
even with very large inputs. Please note that this is really only
useful compared to your 2nd example above only if you need speed at
the expense of clarity. You might want to set the array to NaN
originally, to notice any "missing" days in this example. Beware that
the cumulative total method of binning suffers greater round-off error
than the other methods, so you might be advised to use it with /

DOUBLE.

JD