Subject: Re: quick search of array
Posted by K. Bowman on Thu, 23 Jun 2005 19:33:49 GMT
View Forum Message <> Reply to Message

In article <1119544592.305102.257100@g49g2000cwa.googlegroups.com>,
 tanqian@hotmail.com wrote:

> Hi,
>
> I have four arraies
>
> all_time = fltarr(10000)
> all_location = fltarr(3, 10000)
>
> sel_time = fltarr(4000)
> sel_location = fltarr(3,4000)
>
> the 'sel_time' is a subset of 'all_time'. Is there a quick way I could
> find index 'j' of all_time for each sel_time(i) (when
> abs(all_time(j)-sel_time(i)) lt 1e-5)? So I could use that index to
> pick the sel_location from all_location for each sel_time(i). Since I
> have multiple(~20000) such tests, loop through both of them will be too
> time consuming.
>
> Should I sort both all_time and sel_time to descent/ascent order to
> speed it up?

If you sort your arrays, you can use VALUE_LOCATE (a binary search) instead of WHERE (an exhaustive search). Depending on circumstances, the speed difference can be substantial.

Ken Bowman

Subject: Re: quick search of array
Posted by R.G. Stockwell on Thu, 23 Jun 2005 20:21:44 GMT
View Forum Message <> Reply to Message

<tanqian@hotmail.com> wrote in message
 news:1119544592.305102.257100@g49g2000cwa.googlegroups.com.. .
> Hi,
>
> I have four arraies
>
> all_time = fltarr(10000)
> all_location = fltarr(3, 10000)
>
> sel_time = fltarr(4000)

> sel_location = fltarr(3,4000)
>
> the 'sel_time' is a subset of 'all_time'. Is there a quick way I could
> find index 'j' of all_time for each sel_time(i) (when
> abs(all_time(j)-sel_time(i)) lt 1e-5)? So I could use that index to
> pick the sel_location from all_location for each sel_time(i). Since I
> have multiple(~20000) such tests, loop through both of them will be too
> time consuming.
>
> Should I sort both all_time and sel_time to descent/ascent order to
> speed it up?
>
> Thanks,
>
> Qian
> Qian
>

How about the usual trick of expanding the 1D array to 2D arrays,
for instance:

a = all_time # (fltarr(4000)+1)

b = (fltarr(10000)+1) # sel_time

c = abs(a - b)

w = where(c lt 0.001, count)

print,count

Of course, you'll want to decode these where results into the two
dimensions,
and then take the row as the index with which to access the alltime array.

for instance, something like:

selecteddata = all_location(*,wcols)

Cheers,
bob