Subject: Re: Faster way ?
Posted by Craig Markwardt on Mon, 27 Sep 2004 17:44:30 GMT
View Forum Message <> Reply to Message

rats@mail.geog.uvic.ca (Rafael Loos) writes:
> Hi, I am trying to find the number of values that are within a range
> ...
> I have an Array that has 3 columns and 5 millions lines.
> Thats what I am doing ...
>
> number = WHERE((Array[1,*] GE Min) AND (Array[1,*] LE Max), geralX)
>
> I am storing the number inside the variable geralX ...
> It is taking 0.23 seconds ... but I want to know if there is a faster
> way to find that ...

If you are doing this many times in a loop and ARRAY is unchanging, it
may be worth extracting ARRAY[1,*] into its own variable. That way,
you will save the time of extracting each iteration.

If you just want the total number of elements that match your filter,
you can use total, as in:

```
filter = (Array[1,*] GE Min) AND (Array[1,*] LE Max)
geralX = total(filter)
```

Good luck,
Craig

--
-- --------------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@REMOVEcow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
-- --------------

Subject: Re: Faster way ?
Posted by Dick Jackson on Mon, 27 Sep 2004 20:46:14 GMT
View Forum Message <> Reply to Message

"Craig Markwardt" <craigmnet@REMOVEcow.physics.wisc.edu> wrote in message
news:onr7onip35.fsf@cow.physics.wisc.edu...
> rats@mail.geog.uvic.ca (Rafael Loos) writes:
>> Hi, I am trying to find the number of values that are within a range
>> ...
>> I have an Array that has 3 columns and 5 millions lines.
>> Thats what I am doing ...

>>
>> number = WHERE((Array[1,*] GE Min) AND (Array[1,*] LE Max), geralX)
>>
>> I am storing the number inside the variable geralX ...
>> It is taking 0.23 seconds ... but I want to know if there is a faster
>> way to find that ...
>
> If you are doing this many times in a loop and ARRAY is unchanging, it
> may be worth extracting ARRAY[1,*] into its own variable. That way,
> you will save the time of extracting each iteration.
>
> If you just want the total number of elements that match your filter,
> you can use total, as in:
>
>   filter = (Array[1,*] GE Min) AND (Array[1,*] LE Max)
>   geralX = total(filter)

Even with the two uses of Array[1,*], I got 30-40% time reduction with this:

array1 = Array[1,*]
number = WHERE((Array1 GE Min) AND (Array1 LE Max), geralX)

... and then splicing in your method gave a total of about 45% time
reduction:

array1 = Array[1,*]
geralX = Total((Array1 GE Min) AND (Array1 LE Max))

Hope this helps!

Cheers,
--
-Dick

Dick Jackson / dick@d-jackson.com
D-Jackson Software Consulting / http://www.d-jackson.com
Calgary, Alberta, Canada / +1-403-242-7398 / Fax: 241-7392

Subject: Re: Faster way ?
Posted by btt on Mon, 27 Sep 2004 21:12:56 GMT
View Forum Message <> Reply to Message

Dick Jackson wrote:
> "Craig Markwardt" <craigmnet@REMOVEcow.physics.wisc.edu> wrote in message
> news:onr7onip35.fsf@cow.physics.wisc.edu...
>
>> rats@mail.geog.uvic.ca (Rafael Loos) writes:

>>
>>> Hi, I am trying to find the number of values that are within a range
>>> ...
>>> I have an Array that has 3 columns and 5 millions lines.
>>> Thats what I am doing ...
>>>
>>> number = WHERE((Array[1,*] GE Min) AND (Array[1,*] LE Max), geralX)
>>>
>>> I am storing the number inside the variable geralX ...
>>> It is taking 0.23 seconds ... but I want to know if there is a faster
>>> way to find that ...
>>
>> If you are doing this many times in a loop and ARRAY is unchanging, it
>> may be worth extracting ARRAY[1,*] into its own variable. That way,
>> you will save the time of extracting each iteration.
>>
>> If you just want the total number of elements that match your filter,
>> you can use total, as in:
>>
>> filter = (Array[1,*] GE Min) AND (Array[1,*] LE Max)
>> geralX = total(filter)
>
>
> Even with the two uses of Array[1,*], I got 30-40% time reduction with this:
>
> array1 = Array[1,*]
> number = WHERE((Array1 GE Min) AND (Array1 LE Max), geralX)
>
> ... and then splicing in your method gave a total of about 45% time
> reduction:
>
> array1 = Array[1,*]
> geralX = Total((Array1 GE Min) AND (Array1 LE Max))
>

Hello,

You know, if your data is composed of integers, then you canuse histogram pretty effectively (unless you expect to need a gazillion bins.) The steps below give the following results...

Orginal Method (msec) 656.23403
Histogram Method (msec) 29.270887
Where Method (msec) 180.91989
Total Method (msec) 57.934046

As a bonus, you can use the REVERSE_INDICES keyword to get the locations of these values that fit your criteria. I suppose you could do this with floats,

but then you have to fuss with where the bin locations start and how wide they are. But, I suppose if you do have floating decimal data but your cutoffs are fairly coarse, you could pull a fast one by bumping them up a could of orders of magnitude and then converting to integers.

For example, if your cutoffs are known to the second decimal place then maybe...

```
bumpedarray = LONG(array * 100)
bumpedMin = LONG(minVal * 100)
bumpedMax = LONG(maxVal * 100)
```

Now the histogram function might be handy (again, depends on what you data are like.)

Ben

```
***** BEGIN HERE
array = LONG(RANDOMN(seed, 3, 5000000L))

array1 = Array[1,*]
bottom = MIN(array1, max = TOP)
minVal = -2L
maxval = 2L


t0 = systime(/sec)
number = WHERE((Array[1,*] GE MinVal) AND (Array[1,*] LE MaxVal), geralX)
print, 'Orginal Method (msec)', 1000*(systime(/sec) - t0)


t0 = systime(/sec)
h = HISTOGRAM(array1,min = bottom, max = top, location = loc)
a = where(loc GE minval AND loc LE maxval, cnt)
if cnt GT 0 then geralX = TOTAL(H[A]) else geralX = 0
print, 'Histogram Method (msec)', 1000*(systime(/sec) - t0)

t0 = systime(/sec)
geralX = Total((Array1 GE MinVal) AND (Array1 LE MaxVal))
number = WHERE((Array1 GE Minval) AND (Array1 LE MaxVal), geralX)
print, 'Where Method (msec)', 1000*(systime(/sec) - t0)

t0 = systime(/sec)
geralX = Total((Array1 GE MinVal) AND (Array1 LE MaxVal))
print, 'Total Method (msec)', 1000*(systime(/sec) - t0)


*****FINI HERE
```

Subject: Re: Faster way ?
Posted by JD Smith on Tue, 28 Sep 2004 15:49:36 GMT
View Forum Message <> Reply to Message

On Mon, 27 Sep 2004 20:46:14 +0000, Dick Jackson wrote:

>
> "Craig Markwardt" <craigmnet@REMOVEcow.physics.wisc.edu> wrote in message
> news:onr7onip35.fsf@cow.physics.wisc.edu...
>> rats@mail.geog.uvic.ca (Rafael Loos) writes:
>>> Hi, I am trying to find the number of values that are within a range
>>> ...
>>> I have an Array that has 3 columns and 5 millions lines. Thats what I
>>> am doing ...
>>>
>>> number = WHERE((Array[1,*] GE Min) AND (Array[1,*] LE Max), geralX)
>>>
>>> I am storing the number inside the variable geralX ... It is taking
>>> 0.23 seconds ... but I want to know if there is a faster way to find
>>> that ...
>>
>> If you are doing this many times in a loop and ARRAY is unchanging, it
>> may be worth extracting ARRAY[1,*] into its own variable. That way, you
>> will save the time of extracting each iteration.
>>
>> If you just want the total number of elements that match your filter,
>> you can use total, as in:
>>
>> filter = (Array[1,*] GE Min) AND (Array[1,*] LE Max) geralX =
>> total(filter)
>
> Even with the two uses of Array[1,*], I got 30-40% time reduction with
> this:
>
> array1 = Array[1,*]
> number = WHERE((Array1 GE Min) AND (Array1 LE Max), geralX)
>
> ... and then splicing in your method gave a total of about 45% time
> reduction:
>
> array1 = Array[1,*]
> geralX = Total((Array1 GE Min) AND (Array1 LE Max))

It may not be directly relevant to this problem, but if you only care
about whether *any* values match the filter (i.e. geralX gt 0) then you
can use:

geralX = ~array_equal((Array1 GE MinVal) AND (Array1 LE MaxVal),0b)

which offers some slight gains (though not as much as you'd think:
most the time is spent on the comparison operations). By the way,
it's not fair to precomute min/max for HISTOGRAM outside of the time
accounting. When you move it back in, I get:

Orginal Method (msec) 651.46804
Histogram Method (msec) 87.692976
Where Method (msec) 211.58504
Total Method (msec) 95.319033
Array_Equal method (msec) 86.041927

which depends somewhat on how quickly ARRAY_EQUAL finds a
non-complying value (and can therefore abort). Another testament to
the heavy internal optimization of HISTOGRAM.

JD

Subject: Re: Faster way ?
Posted by btt on Tue, 28 Sep 2004 16:05:50 GMT
View Forum Message <> Reply to Message

JD Smith wrote:
By the way,
> it's not fair to precomute min/max for HISTOGRAM outside of the time
> accounting.

oops!

Orginal Method (msec) 662.43482
Histogram Method (msec) 50.179005
Where Method (msec) 174.30687
Total Method (msec) 55.247068
Array Equal Method (msec) 49.051046
