Subject: N_ELEMENTS and WHERE: Scalar or Array ?
Posted by rmlongfield on Fri, 29 Jan 1999 08:00:00 GMT
View Forum Message <> Reply to Message

Hi All, I keep making the same mistake with N_ELEMENTS so I decided
to write and ask if anyone has found a solution. I use WHERE to find some
zeroes in a data set which I want to exclude in further processing. Problem
is that sometimes they are all zeroes. Using a simple :

non_zero_xvalues = WHERE (subarray1 GT 0)
IF(non_zero_array EQ -1) ...

gets me into trouble because, when it is an array, I get an error. If I use
N_ELEMENTS(non_zero_array) there is always at least one element, whether it
is -1 or something else. I don't like ignoring the 'something else' value
just because it is the only one. Is the answer another IF statement or some
sort of error control?

Thanks !
Rose

Subject: Re: N_ELEMENTS and WHERE: Scalar or Array ?
Posted by Mark Buckley on Fri, 29 Jan 1999 08:00:00 GMT
View Forum Message <> Reply to Message

rmlongfield@my-dejanews.com wrote in message
<78s23o$525$1@nnrp1.dejanews.com>...

> I use WHERE to find some zeroes in a data set which I
> want to exclude in further processing. Problem
> is that sometimes they are all zeroes. Using a simple :
>
> non_zero_xvalues = WHERE (subarray1 GT 0)
> IF(non_zero_array EQ -1) ...
>
> gets me into trouble because, when it is an array, I get an error. If I use
> N_ELEMENTS(non_zero_array) there is always at least one element, whether it
> is -1 or something else. I don't like ignoring the 'something else' value
> just because it is the only one. Is the answer another IF statement or
some
> sort of error control?

I suspect that you need to use the COUNT parameter to WHERE, to tell

you how many values were matched in the WHERE clause. Check COUNT,
and if it is equal to zero, you know that none passed the test...

cheers,

Mark

Rutherford-Appleton Laboratory
Chilton
Didcot
OXON

Subject: Re: N_ELEMENTS and WHERE: Scalar or Array ?
Posted by davidf on Fri, 29 Jan 1999 08:00:00 GMT
View Forum Message <> Reply to Message

Rose (rmlongfield@my-dejanews.com) writes:

> Hi All, I keep making the same mistake with N_ELEMENTS so I decided
> to write and ask if anyone has found a solution. I use WHERE to find some
> zeroes in a data set which I want to exclude in further processing. Problem
> is that sometimes they are all zeroes. Using a simple :
>
> non_zero_xvalues = WHERE (subarray1 GT 0)
> IF(non_zero_array EQ -1) ...
>
> gets me into trouble because, when it is an array, I get an error. If I use
> N_ELEMENTS(non_zero_array) there is always at least one element, whether it
> is -1 or something else. I don't like ignoring the 'something else' value
> just because it is the only one. Is the answer another IF statement or some
> sort of error control?

The second positional parameter of the WHERE function is
an output variable that holds the "count" of the number
of elements found that meet the WHERE criteria. So the
proper sequence of commands looks something like this:

```
index = WHERE( array GT 0, count)
IF count GT 0 THEN ....
```

Cheers,

David

David Fanning
Fanning Software Consulting

Phone: 970-221-0438
E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: N_ELEMENTS and WHERE: Scalar or Array ?
Posted by David Ritscher on Fri, 29 Jan 1999 08:00:00 GMT
View Forum Message <> Reply to Message

> non_zero_xvalues = WHERE (subarray1 GT 0)
> IF(non_zero_array EQ -1) ...
>
> gets me into trouble because, when it is an array, I get an error. If I use
> N_ELEMENTS(non_zero_array) there is always at least one element, whether it
> is -1 or something else. I don't like ignoring the 'something else' value
> just because it is the only one. Is the answer another IF statement or some
> sort of error control?

This is my old standby:

non_zero_xvalues = WHERE (subarray1 GT 0)
IF(non_zero_array(0) EQ -1) ...

David Ritscher
--
Cardiac Rhythm Management Laboratory
Department of Medicine
University of Alabama at Birmingham
B168 Volker Hall - 1670 University Boulevard
Birmingham AL 35294-0019
Tel: (205) 975-2122   Fax: (205) 975-4720
Email: david.ritscher@bigfoot.com

Subject: Re: N_ELEMENTS and WHERE: Scalar or Array ?
Posted by David Kastrup on Fri, 29 Jan 1999 08:00:00 GMT
View Forum Message <> Reply to Message

rmlongfield@my-dejanews.com writes:

> Hi All, I keep making the same mistake with N_ELEMENTS so I decided
> to write and ask if anyone has found a solution. I use WHERE to find some
> zeroes in a data set which I want to exclude in further processing. Problem
> is that sometimes they are all zeroes. Using a simple :
>
> non_zero_xvalues = WHERE (subarray1 GT 0)

> IF(non_zero_array EQ -1) ...
>
> gets me into trouble because, when it is an array, I get an error.

How about
IF((size(non_zero_xvalues))[0] EQ 0)

Namely, checking whether non_zero_xvalues is a scalar?

Or, less complicated,
IF (non_zero_values[0] EQ -1)
because you are allowed to index a scalar with 0?

--
David Kastrup Phone: +49-234-700-5570
Email: dak@neuroinformatik.ruhr-uni-bochum.de Fax: +49-234-709-4209
Institut fï¿½r Neuroinformatik, Universitï¿½tsstr. 150, 44780 Bochum, Germany

---

Subject: Re: N_ELEMENTS and WHERE: Scalar or Array ?
Posted by R.Bauer on Fri, 29 Jan 1999 08:00:00 GMT
View Forum Message <> Reply to Message

rmlongfield@my-dejanews.com wrote:

> Hi All, I keep making the same mistake with N_ELEMENTS so I decided
> to write and ask if anyone has found a solution. I use WHERE to find some
> zeroes in a data set which I want to exclude in further processing. Problem
> is that sometimes they are all zeroes. Using a simple :
>
> non_zero_xvalues = WHERE (subarray1 GT 0)

You should use
non_zero_xvalues = WHERE (subarray1 GT 0,count_non_zero_xvalues)
if count_non_zero_xvalues ne 0 then begin
endif

>
> IF(non_zero_array EQ -1) ...

IF non_zero_array[0] EQ -1 ..

>
>
> gets me into trouble because, when it is an array, I get an error. If I use

> N_ELEMENTS(non_zero_array) there is always at least one element, whether it
> is -1 or something else. I don't like ignoring the 'something else' value
> just because it is the only one. Is the answer another IF statement or some
> sort of error control?
>

Regards
R.Bauer

Subject: Re: N_ELEMENTS and WHERE: Scalar or Array ?
Posted by Craig Markwardt on Mon, 01 Feb 1999 08:00:00 GMT
View Forum Message <> Reply to Message

wmc@bas.ac.uk writes:
> ...
> But since this has come up, & its one of my pet peeves: why cannot where
> return a null array to indicate no-elements-match. And then array[null] would
> match to nothing. This would allow one to say
>
> array[where(wurble)]='stoat'
>
> instead of the ugly
>
> i=where(wurble,count)
> if (count gt 0) then array[i]='stoat'
>

Amen brother! It is sadly unfortunate that where(0) returns a number
at all. A null array -- if IDL had them! -- would be the best.

Of course, there is the trade-off between power user and newbie user.
Consider the following:

```
wh_one = where(x GT 1)
x[wh_ine] = 0             ; NOTE the type
```

Because of the typo, the second statement would have the mysterious
side effect of having no effect at all! (including no error message) I
would still love to have it, but newbies might go crazy.

Craig

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

Subject: Re: N_ELEMENTS and WHERE: Scalar or Array ?
Posted by wmc on Mon, 01 Feb 1999 08:00:00 GMT
View Forum Message <> Reply to Message

Stein Vidar Hagfors Haugan <steinhh@ulrik.uio.no> wrote:
> In article <36b57934.0@news.nwl.ac.uk> wmc@bas.ac.uk writes:
>> But since this has come up, & its one of my pet peeves: why cannot where
>> return a null array to indicate no-elements-match. And then array[null] would
>> match to nothing. This would allow one to say
>>
>> array[where(wurble)]='stoat'
>>

> I agree, it would be a nice feature... However, I have problems
> seeing how to implement this, without altering e.g. the way IDL
> allows you to index arrays with an array that is out of
> bounds.. (try help,(findgen(10))(findgen(20)-5))

I'm not sure this is so: indexing by nulls ("where" in the example above would return "null", not -1) can be distinguished from out-of-range.

But even so: I've always felt that allowing indexing by out of bounds indices is more a bug that a feature. Why is it possible? Can you think of an example where it is useful, or necessary?

If this is necessary for legacy reasons, it might be possible to make () and [] behave differently in this case? Possibly a missed opportunity when [] came in!

> Maybe using (float/double) NaN values? Hmm. That could work!
> Let's see,

> array[NaN] = 5 ; Would be allowed, but does nothing

This could well be possible as an easy-to-do work-around. In that case, where would have to return NaN not -1.

The other possibility (which would only work for this special case, but its quite a common special case) is that -1 would count as a "special" value & assigning to array[-1] would, as a special case, just do nothing rather than producing an error message.

Incidentally, I've just realised how dangerous the out-of-bounds stuff is:

array([where(array eq false)])='stoat'

assigns to the first element...

>> While I'm here: would RSI please put a decent regexp package into IDL?

> Take a look at http://www.uio.no/~steinhh/IDL/additions.html for an
> example of how to add it using dynamically loadable modules.
> It requires a Unix flavor with regex.h

I have that so I'll take a look. Thanks!

-W.

--
William M Connolley | wmc@bas.ac.uk | http://www.nbs.ac.uk/public/icd/wmc/
Climate Modeller, British Antarctic Survey | Disclaimer: I speak for myself

---

Subject: Re: N_ELEMENTS and WHERE: Scalar or Array ?
Posted by steinhh on Mon, 01 Feb 1999 08:00:00 GMT
View Forum Message <> Reply to Message

In article <36b57934.0@news.nwl.ac.uk> wmc@bas.ac.uk writes:
[...]
> But since this has come up, & its one of my pet peeves: why cannot where
> return a null array to indicate no-elements-match. And then array[null] would
> match to nothing. This would allow one to say
>
>   array[where(wurble)]='stoat'
>
> instead of the ugly
>
>   i=where(wurble,count)
>   if (count gt 0) then array[i]='stoat'
>
> I use Perl a lot, and the contrast is very striking there: perl handles null
> values quite happily and it simplifies a lot of things.

I agree, it would be a nice feature... However, I have problems
seeing how to implement this, without altering e.g. the way IDL
allows you to index arrays with an array that is out of
bounds.. (try help,(findgen(10))(findgen(20)-5))

Maybe using (float/double) NaN values? Hmm. That could work!
Let's see,

```
array[NaN] = 5        ; Would be allowed, but does nothing
array[ [NaN,0] ] = 5  ; Assigns 5 to element 0 only.
```

```
data = array[NaN]       ; data is set to NaN !
data = array[ [NaN,0] ] ; Data is set to array[0]
```

The only problem I can see is that the indices will sometimes have to be converted to floats/doubles in order to store the NaN values, thus some overhead will occur...

> While I'm here: would RSI please put a decent regexp package into IDL?

Take a look at http://www.uio.no/~steinhh/IDL/additions.html for an example of how to add it using dynamically loadable modules.

It requires a Unix flavor with regex.h, or alternatively the POSIX compliant inteface for the GNU regular expression library (not tested, though) - see http://sunland.gsfc.nasa.gov/info/regex/Top.html, especially the section "Programming with Regex".

Regards,

Stein Vidar

Subject: Re: N_ELEMENTS and WHERE: Scalar or Array ?
Posted by wmc on Mon, 01 Feb 1999 08:00:00 GMT
View Forum Message <> Reply to Message

rmlongfield@my-dejanews.com wrote:
> Hi All, I keep making the same mistake with N_ELEMENTS so I decided
> to write and ask if anyone has found a solution. I use WHERE to find some
> zeroes in a data set which I want to exclude in further processing. Problem
> is that sometimes they are all zeroes. Using a simple :

> non_zero_xvalues = WHERE (subarray1 GT 0)
> IF(non_zero_array EQ -1) ...

> gets me into trouble because, when it is an array, I get an error. If I use
> N_ELEMENTS(non_zero_array) there is always at least one element, whether it
> is -1 or something else. I don't like ignoring the 'something else' value
> just because it is the only one. Is the answer another IF statement or some
> sort of error control?

The temporay solution to this is to use where(wurble,count)...

But since this has come up, & its one of my pet peeves: why cannot where return a null array to indicate no-elements-match. And then array[null] would match to nothing. This would allow one to say

```
  array[where(wurble)]='stoat'
```

instead of the ugly

```
  i=where(wurble,count)
  if (count gt 0) then array[i]='stoat'
```

I use Perl a lot, and the contrast is very striking there: perl handles null values quite happily and it simplifies a lot of things.

While I'm here: would RSI please put a decent regexp package into IDL?

-W.

--
William M Connolley | wmc@bas.ac.uk | http://www.nbs.ac.uk/public/icd/wmc/
Climate Modeller, British Antarctic Survey | Disclaimer: I speak for myself

---

Subject: Re: N_ELEMENTS and WHERE: Scalar or Array ?
Posted by Jack Saba on Tue, 02 Feb 1999 08:00:00 GMT
View Forum Message <> Reply to Message

Stein Vidar Hagfors Haugan wrote:
>
:
> The problem is the "null" - it ought to be something other than an
> integer/long/long64. Ok, so maybe -2LL^63 would do... and of course
> you'd need to keep compatible, so you need WHERE(..,/null)
>
:
>
> How'bout {} ? :-) I'm not *just* kidding. [] work as both array
> constructors and indexing brackets, so {} could work as both
> structure constructors and indexing brackets..

Differences in function based on differences in types of brackets will lead
to confusion -- in part because of the difficulty of distinguishing {
and [
and (in some fonts.

The idea of a switch on the WHERE function sounds much better.

Subject: Re: N_ELEMENTS and WHERE: Scalar or Array ?
Posted by steinhh on Tue, 02 Feb 1999 08:00:00 GMT

In article <36b5d66b.0@news.nwl.ac.uk> wmc@bas.ac.uk writes:
[..]
> I'm not sure this is so: indexing by nulls ("where" in the example
> above would return "null", not -1) can be distinguished from out-
> of-range.

The problem is the "null" - it ought to be something other than an integer/long/long64. Ok, so maybe -2LL^63 would do... and of course you'd need to keep compatible, so you need WHERE(..,/null)

> But even so: I've always felt that allowing
> indexing by out of bounds indices is more a bug that a feature. Why
> is it possible? Can you think of an example where it is useful, or
> necessary?

Uh - no, *I* don't think it's a good thing. RSI does (did?) :-)

> If this is necessary for legacy reasons, it might be possible to make
> () and [] behave differently in this case? Possibly a missed
> opportunity when [] came in!

How'bout {} ? :-) I'm not *just* kidding. [] work as both array constructors and indexing brackets, so {} could work as both structure constructors and indexing brackets..

[..]
>> array[NaN] = 5 ; Would be allowed, but does nothing
>
> This could well be possible as an easy-to-do work-around. In that
> case, where would have to return NaN not -1.

(Yes - though with a WHERE(..,/nan) switch)

> The other possibility (which would only work for this special case,
> but its quite a common special case) is that -1 would count as a
> "special" value & assigning to array[-1] would, as a special case,
> just do nothing rather than producing an error message.
>
> Incidentally, I've just realised how dangerous the out-of-bounds stuff
> is:
>
> array([where(array eq false)])='stoat'
>
> assigns to the first element...

And you can *bet* some program(mer)s out there are counting on exactly this as a *feature*! Sorry to say so, but...that's why

you'd have to introduce a keyword switch in WHERE.

Regards,

Stein Vidar

Subject: Re: N_ELEMENTS and WHERE: Scalar or Array ?
Posted by davidf on Wed, 03 Feb 1999 08:00:00 GMT
View Forum Message <> Reply to Message

William Connolley (wmc@bas.ac.uk) writes:

> Well, they are very silly people then. Does anyone on this newsgroup want
> to confess to using this "feature"?

Uh, not me. But I saw Ray Sterner use it once very effectively. :-)

Cheers,

David

P.S. Unfortunately, the context escapes me. But I remember
being impressed. :-)

--
David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: N_ELEMENTS and WHERE: Scalar or Array ?
Posted by steinhh on Wed, 03 Feb 1999 08:00:00 GMT
View Forum Message <> Reply to Message

wmc@bas.ac.uk wrote:
>> And you can *bet* some program(mer)s out there are counting on
>> exactly this as a *feature*! Sorry to say so, but...that's why
>> you'd have to introduce a keyword switch in WHERE.
>
> Well, they are very silly people then. Does anyone on this newsgroup want
> to confess to using this "feature"?

After seeing *that* comment? :-)

Subject: Re: N_ELEMENTS and WHERE: Scalar or Array ?
Posted by wmc on Wed, 03 Feb 1999 08:00:00 GMT
View Forum Message <> Reply to Message

Stein Vidar Hagfors Haugan <steinhh@ulrik.uio.no> wrote:
> In article <36b5d66b.0@news.nwl.ac.uk> wmc@bas.ac.uk writes:

>> If this is necessary for legacy reasons, it might be possible to make
>> () and [] behave differently in this case? Possibly a missed
>> opportunity when [] came in!

> How'bout {} ? :-) I'm not *just* kidding. [] work as both array
> constructors and indexing brackets, so {} could work as both
> structure constructors and indexing brackets..

Hmm, having three different sorts of brackets to make arrays is
a bit of overkill. Anyway, {} might be needed for associative
arrays one day!

>>> array[NaN] = 5 ; Would be allowed, but does nothing
>>
>> This could well be possible as an easy-to-do work-around. In that
>> case, where would have to return NaN not -1.

> (Yes - though with a WHERE(..,/nan) switch)

Hmm, that would be acceptable. Or an nwhere function (a bit less typing).

>> Incidentally, I've just realised how dangerous the out-of-bounds stuff
>> is:
>>
>> array([where(array eq false)])='stoat'
>>
>> assigns to the first element...

> And you can *bet* some program(mer)s out there are counting on
> exactly this as a *feature*! Sorry to say so, but...that's why
> you'd have to introduce a keyword switch in WHERE.

Well, they are very silly people then. Does anyone on this newsgroup want
to confess to using this "feature"?

-W.

--

William M Connolley | wmc@bas.ac.uk | http://www.nbs.ac.uk/public/icd/wmc/
Climate Modeller, British Antarctic Survey | Disclaimer: I speak for myself

Subject: Re: N_ELEMENTS and WHERE: Scalar or Array ?
Posted by wmc on Wed, 03 Feb 1999 08:00:00 GMT
View Forum Message <> Reply to Message

Craig Markwardt <craigmnet@astrog.physics.wisc.edu> wrote:
> wmc@bas.ac.uk writes:
>> ...
>> But since this has come up, & its one of my pet peeves: why cannot where
>> return a null array to indicate no-elements-match. And then array[null] would
>> match to nothing. This would allow one to say
>>
>> array[where(wurble)]='stoat'

> Amen brother! It is sadly unfortunate that where(0) returns a number
> at all. A null array -- if IDL had them! -- would be the best.

> Of course, there is the trade-off between power user and newbie user.
> Consider the following:

> wh_one = where(x GT 1)
> x[wh_ine] = 0 ; NOTE the type

> Because of the typo, the second statement would have the mysterious
> side effect of having no effect at all! (including no error message) I
> would still love to have it, but newbies might go crazy.

No, we're not going completely perl here: there is a distinction between an undeclared variable (which would still cause an error in your example above) and a variable initialised with a null value (which is what I would like "where" to return). So your newbie would be safe.

In fact, would there be a problem with a "null" type, which would perhaps be variable type #11? Then you don't have to worry about the actual value it contains.

-W.

--
William M Connolley | wmc@bas.ac.uk | http://www.nbs.ac.uk/public/icd/wmc/
Climate Modeller, British Antarctic Survey | Disclaimer: I speak for myself

Subject: Re: N_ELEMENTS and WHERE: Scalar or Array ?

Posted by Annette Schloss on Wed, 03 Feb 1999 08:00:00 GMT
View Forum Message <> Reply to Message

Hi Rose,
you can do this:
non_zero_xvalues = WHERE (subarray1 GT 0, count)
if (count ne 0) then do something

Count will be zero if nothing satisfies the WHERE, otherwise it will be a positive number.

Does this help?
-Annette

rmlongfield@my-dejanews.com wrote:
>
> Hi All, I keep making the same mistake with N_ELEMENTS so I decided
> to write and ask if anyone has found a solution. I use WHERE to find some
> zeroes in a data set which I want to exclude in further processing. Problem
> is that sometimes they are all zeroes. Using a simple :
>
> non_zero_xvalues = WHERE (subarray1 GT 0)
> IF(non_zero_array EQ -1) ...
>
> gets me into trouble because, when it is an array, I get an error. If I use
> N_ELEMENTS(non_zero_array) there is always at least one element, whether it
> is -1 or something else. I don't like ignoring the 'something else' value
> just because it is the only one. Is the answer another IF statement or some
> sort of error control?
>
> Thanks !
> Rose
>
> ----------== Posted via Deja News, The Discussion Network ==----------
> http://www.dejanews.com/ Search, Read, Discuss, or Start Your Own

--
@ Annette Schloss annette.schloss@unh.edu
@ Complex Systems Research Center fax: 603-862-0188
@ 39 College Rd., U. of New Hampshire phn: 603-862-1792
@ Durham, NH 03824

Subject: Re: N_ELEMENTS and WHERE: Scalar or Array ?
Posted by David Ritscher on Thu, 04 Feb 1999 08:00:00 GMT

> wh_one = where(x GT 1)
> x[wh_ine] = 0 ; NOTE the type

> Because of the typo, the second statement would have the mysterious
> side effect of having no effect at all! (including no error message) I
> would still love to have it, but newbies might go crazy.

For questions regarding subscript handling, it's always worth looking at how MATLAB does things. They have a null array as a possible variable, and seem to handle well what happens with this (logically enough, when one indexes an array with a null array, the result is null):

```
>>  a = [1, 2]

a =

    1     2

>>  b = []

b =

    []

>>  a(b)

ans =

    []

>>
>>  whos
  Name      Size         Bytes  Class

  a         1x2             16  double array
  ans       0x0              0  double array
  b         0x0              0  double array

Grand total is 2 elements using 16 bytes
```

David Ritscher
--
Cardiac Rhythm Management Laboratory

Department of Medicine
University of Alabama at Birmingham
B168 Volker Hall - 1670 University Boulevard
Birmingham AL 35294-0019
Tel: (205) 975-2122 Fax: (205) 975-4720
Email: david.ritscher@bigfoot.com

Subject: Re: N_ELEMENTS and WHERE: Scalar or Array ?
Posted by Michael Werger on Thu, 04 Feb 1999 08:00:00 GMT
View Forum Message <> Reply to Message

Annette Schloss wrote:
>
> Hi Rose,
> you can do this:
> non_zero_xvalues = WHERE (subarray1 GT 0, count)
> if (count ne 0) then do something
>
> Count will be zero if nothing satisfies the WHERE, otherwise it will be a
> positive number.
>

And what about:

```
non_zero_xvalues = WHERE(subarray1 GT 0)
IF (min(non_zero_xvalues) NE -1) THEN
 ...
ENDIF
```

was this already proposed? Sorry then
--
Michael Werger ----------o
ESA ESTEC & Praesepe B.V. |
Astrophysics Division mwerger@astro.estec.esa.nl|
| Postbus 299 http://astro.estec.esa.nl |
| 2200 AG Noordwijk +31 71 565 3783 (Voice)
o---------- The Netherlands +31 71 565 4690 (FAX)
