Subject: Re: How to test for a vector/matrix of constants?
Posted by Allan Whiteford on Wed, 06 Dec 2006 18:40:38 GMT
View Forum Message <> Reply to Message

Mirko,

if (where(vector ne vector[0]))[0] eq -1 then print,'all same'

Thanks,

Allan

Mirko wrote:
> How can I quickly check if a vector/matrix is full of constants (all
> elements equal)?
> For example if a vector contained:
> [2.38,2.38,2.38,...,2.38]
> I want it flagged as a "constant" vector.
>
> I can think of finding differences between successive elements, and
> check for non-zero elements.
>
> Any faster options?
>
> Thanks,
>
> Mirko
>

Subject: Re: How to test for a vector/matrix of constants?
Posted by Bob[3] on Wed, 06 Dec 2006 18:43:21 GMT
View Forum Message <> Reply to Message

Mirko,

How about:
ARRAY_EQUAL(a, a[0])

Bob.

Subject: Re: How to test for a vector/matrix of constants?
Posted by Mirko on Wed, 06 Dec 2006 19:13:53 GMT
View Forum Message <> Reply to Message

Bob wrote:

> Mirko,
>
> How about:
> ARRAY_EQUAL(a, a[0])
>
> Bob.

Thanks Allan and Bob,

I think I will go with Bob's. It is "more intrinsic".

Mirko

Subject: Re: How to test for a vector/matrix of constants?
Posted by Braedley on Wed, 06 Dec 2006 19:38:24 GMT
View Forum Message <> Reply to Message

Mirko wrote:
> How can I quickly check if a vector/matrix is full of constants (all
> elements equal)?
> For example if a vector contained:
> [2.38,2.38,2.38,...,2.38]
> I want it flagged as a "constant" vector.
>
> I can think of finding differences between successive elements, and
> check for non-zero elements.
>
> Any faster options?
>
> Thanks,
>
> Mirko

Probably not as fast as the others, but

if n_elements(uniq(array, sort(array))) eq 1 then ;flag it

Braedley

Subject: Re: How to test for a vector/matrix of constants?
Posted by Foldy Lajos on Wed, 06 Dec 2006 19:59:45 GMT
View Forum Message <> Reply to Message

On Wed, 6 Dec 2006, Mirko wrote:

> How can I quickly check if a vector/matrix is full of constants (all
> elements equal)?
> For example if a vector contained:
> [2.38,2.38,2.38,...,2.38]
> I want it flagged as a "constant" vector.
>
> I can think of finding differences between successive elements, and
> check for non-zero elements.
>
> Any faster options?
>
> Thanks,
>
> Mirko
>

my solution would be:

mini=min(array, max=maxi)
if mini eq maxi then ...

regards,
lajos

Subject: Re: How to test for a vector/matrix of constants?
Posted by Paul Van Delst[1] on Wed, 06 Dec 2006 20:54:37 GMT
View Forum Message <> Reply to Message

Fï¿½LDY Lajos wrote:
>
> On Wed, 6 Dec 2006, Mirko wrote:
>
>> How can I quickly check if a vector/matrix is full of constants (all
>> elements equal)?
>> For example if a vector contained:
>> [2.38,2.38,2.38,...,2.38]
>> I want it flagged as a "constant" vector.
>>
>> I can think of finding differences between successive elements, and
>> check for non-zero elements.
>>
>> Any faster options?
>>
>> Thanks,
>>
>> Mirko
>>

>
> my solution would be:
>
> mini=min(array, max=maxi)
> if mini eq maxi then ...

I think Bob's solution is the safest since that is the only one that doesn't use the "EQ" operator. Insert the usual reasons here for not wanting to use "EQ" to compare floats.

cheers

paulv

p.s. Of course, one hopes the guts of ARRAY_EQUAL handles it correctly. :o)

--
Paul van Delst Ride lots.
CIMSS @ NOAA/NCEP/EMC Eddy Merckx
Ph: (301)763-8000 x7748
Fax:(301)763-8545

Subject: Re: How to test for a vector/matrix of constants?
Posted by Karl Schultz on Wed, 06 Dec 2006 20:58:35 GMT
View Forum Message <> Reply to Message

On Wed, 06 Dec 2006 20:59:45 +0100, =?ISO-8859-2?Q?F=D6LDY_Lajos?= wrote:

> On Wed, 6 Dec 2006, Mirko wrote:
>
>> How can I quickly check if a vector/matrix is full of constants (all
>> elements equal)?
>> For example if a vector contained:
>> [2.38,2.38,2.38,...,2.38]
>> I want it flagged as a "constant" vector.
>>
>> I can think of finding differences between successive elements, and
>> check for non-zero elements.
>>
>> Any faster options?
>>
>> Thanks,
>>
>> Mirko
>>
>
> my solution would be:

>
> mini=min(array, max=maxi)
> if mini eq maxi then ...
>
> regards,
> lajos

The ARRAY_EQUAL(a, a[0]) approach is probably the best, because it will
"short-circuit" or stop checking as soon as it finds an element that does
not match.

Karl

Subject: Re: How to test for a vector/matrix of constants?
Posted by Bob Crawford on Wed, 06 Dec 2006 21:04:23 GMT
View Forum Message <> Reply to Message

On Dec 6, 3:54 pm, Paul van Delst <Paul.vanDe...@noaa.gov> wrote:
> I think Bob's solution is the safest since that is the only one that doesn't use the "EQ"
> operator. Insert the usual reasons here for not wanting to use "EQ" to compare floats.

Paul,

Most of the other suggestions are actually using the EQ on integers
(array subscript values).
Lots of interesting approaches.

Bob.

Subject: Re: How to test for a vector/matrix of constants?
Posted by Foldy Lajos on Wed, 06 Dec 2006 21:05:18 GMT
View Forum Message <> Reply to Message

On Wed, 6 Dec 2006, Paul van Delst wrote:

> FÖLDY Lajos wrote:
>>
>> my solution would be:
>>
>> mini=min(array, max=maxi)
>> if mini eq maxi then ...
>
> I think Bob's solution is the safest since that is the only one that doesn't
> use the "EQ" operator. Insert the usual reasons here for not wanting to use
> "EQ" to compare floats.

>
> cheers
>
> paulv
>
> p.s. Of course, one hopes the guts of ARRAY_EQUAL handles it correctly. :o)

ARRAY_EQUAL is practically the same as EQ, only the number of comparisons
may differ. And it is available since IDL 5.4 only.

regards,
lajos

ps: Did you know, that ARRAY_EQUAL has the undocumented keywords /NE, /GE,
/GT, /LE, /LT? And they are reserved words :-)

>
> --
> Paul van Delst Ride lots.
> CIMSS @ NOAA/NCEP/EMC Eddy Merckx
> Ph: (301)763-8000 x7748
> Fax:(301)763-8545
>

Subject: Re: How to test for a vector/matrix of constants?
Posted by Paul Van Delst[1] on Wed, 06 Dec 2006 21:26:58 GMT
View Forum Message <> Reply to Message

Bob Crawford wrote:
> On Dec 6, 3:54 pm, Paul van Delst <Paul.vanDe...@noaa.gov> wrote:
>> I think Bob's solution is the safest since that is the only one that doesn't use the "EQ"
>> operator. Insert the usual reasons here for not wanting to use "EQ" to compare floats.
>
> Paul,
>
> Most of the other suggestions are actually using the EQ on integers
> (array subscript values).
> Lots of interesting approaches.

Well, apart from yours, only one other.

From Allan:
if (where(vector ne vector[0]))[0] eq -1 then print,'all same'
-> potential float comparison.

From Braedley
  if n_elements(uniq(array, sort(array))) eq 1 then ;flag it
-> No float comparison

From Lajos
  mini=min(array, max=maxi)
  if mini eq maxi then ...
-> Potential float comparison

Some of said constant, equal numbers could be more equal than others. :o)

cheers,

paulv

--
Paul van Delst Ride lots.
CIMSS @ NOAA/NCEP/EMC Eddy Merckx
Ph: (301)763-8000 x7748
Fax:(301)763-8545

Subject: Re: How to test for a vector/matrix of constants?
Posted by Foldy Lajos on Wed, 06 Dec 2006 21:31:30 GMT
View Forum Message <> Reply to Message

On Wed, 6 Dec 2006, Paul van Delst wrote:

> From Braedley
> if n_elements(uniq(array, sort(array))) eq 1 then ;flag it
> -> No float comparison

Do you think sort works without comparison? :-)

regards,
lajos

Subject: Re: How to test for a vector/matrix of constants?
Posted by JD Smith on Wed, 06 Dec 2006 22:10:23 GMT
View Forum Message <> Reply to Message

On Wed, 06 Dec 2006 22:05:18 +0100, Fï¿½LDY Lajos wrote:

> ps: Did you know, that ARRAY_EQUAL has the undocumented keywords /NE, /GE,
> /GT, /LE, /LT? And they are reserved words :-)

Whoa! Why aren't these documented? I use ARRAY_EWUAL all the time, and would love these. Was it a documentation mistake, or were they hidden for a reason? Karl?

JD

Subject: Re: How to test for a vector/matrix of constants?
Posted by Karl Schultz on Wed, 06 Dec 2006 23:23:26 GMT
View Forum Message <> Reply to Message

On Wed, 06 Dec 2006 15:10:23 -0700, JD Smith wrote:

> On Wed, 06 Dec 2006 22:05:18 +0100, FÖLDY Lajos wrote:
>
>
>> ps: Did you know, that ARRAY_EQUAL has the undocumented keywords /NE, /GE,
>> /GT, /LE, /LT? And they are reserved words :-)
>
> Whoa! Why aren't these documented? I use ARRAY_EWUAL all the time, and
> would love these. Was it a documentation mistake, or were they hidden for
> a reason? Karl?

I checked around and couldn't find a reason. I'll put in a change request and see if we can doc them for next release. There is also a DIFFERENT_LENGTH keyword that will allow comparing arrays of differing lengths, using only the number of elements of the smaller array. These keywords are even exercised in our test suite.

Karl

Subject: Re: How to test for a vector/matrix of constants?
Posted by JD Smith on Wed, 06 Dec 2006 23:53:04 GMT
View Forum Message <> Reply to Message

On Wed, 06 Dec 2006 22:31:30 +0100, Fï¿½LDY Lajos wrote:

>
> On Wed, 6 Dec 2006, Paul van Delst wrote:
>
>> From Braedley
>> if n_elements(uniq(array, sort(array))) eq 1 then ;flag it
>> -> No float comparison
>
> Do you think sort works without comparison? :-)

>
> regards,
> lajos

It's not SORT, but UNIQ which is the problem. Here's how UNIQ works
(see the code !DIR/lib/uniq.pro):

```
 indices = where(q ne shift(q,-1), count)
```

For every one of the mentioned methods, float equality is checked.

I know a lot has been said about the issue of floating point
representation (search "sky is falling float"), but I thought I'd have
a stab.

There's nothing special about float comparison, other than the fact
that it's hard to write down a floating point number uniquely, by hand
as a literal in your code, as ASCII in a text file, etc. This issue
just doesn't exist for integers, for which there is a one-to-one
mapping between integer representation in ASCII text and binary
representation on disk (as long as the number is within the integer
range).

A float is equal to another float only when their exact representation
in memory is equal. Note that the following is not an issue:

```
IDL> print,array_equal(replicate(4.923,100),4.923)
  1
```

It will *always* be true for any floating number specified like this,
even ones which cannot be represented at the given precision:

```
IDL>  print,array_equal(replicate(4.923098790790790879087908790879087, 100), $
       4.923098790790790879087908790879087)
  1
```

Problems only begin to occur when you start to rely on your
(incorrect) intuitive expectation of how a computer *should* handle a
floating point number:

```
IDL> print,array_equal(replicate(4.923,100),1.00 + 1.02 + 1.045 + 1.858)
  0
```

But, why not, you ask, since:

```
IDL> print, 1.00 + 1.02 + 1.045 + 1.858
     4.92300
```

Well, not all of those numbers (if considered as exact values drawn from the inifinite set of all real numbers) can actually be represented uniquely as a float. The classic example of this issue is the decimal number 0.9, which in binary representation is:

0.11100110011001100110011001100110011.....

and repeating so on forever. Unfortunately, you can only allocate so many bits for a float, so:

```
IDL> print, 0.9 eq (0.6 + 0.3)
   0
IDL> print,FORMAT='(F0.8)',0.9, 0.6 + 0.3
0.89999998
0.90000004
```

Note that this doesn't mean that 0.9 and numbers like it will *never* equal to the "sum of their parts". Sometimes you get lucky and the bit truncation works out the same::

```
IDL> print, 0.9 eq (0.5 + 0.4)
   1
```

Here's a nice example in C to get a better handle on this, which will let you peek behind the curtain and see the real bit representation of a given floating point number:

```
#include <stdio.h>
int main() {
  float f=0.9,f1;

  printf("Literal -- %f: %lx\n",f,*(unsigned int *) &f);

  f1=(float) 0.3 + (float) 0.6;
  printf("0.6+0.3 -- %f: %lx\n",f1,*(unsigned int *) &f1);

  printf("Floats are %s\n",f1==f?"Equal":"Not Equal");

}
```

For convenience it prints the floats as 4 bytes of hexadecimal instead of 32 bits of binary, and it uses a de-referencing trick to get at the actual binary data for the float. Here are the results

```
% ./compare_float
Literal -- 0.900000: 3f666666
0.6+0.3 -- 0.900000: 3f666667
Floats are Not Equal
```

Note how I had to cast the literal numbers 0.3 and 0.6 to floats, since by default most C compilers promote literals to double in arithmetic before assigning to float. IDL does this casting natively as well (which is why you aren't guaranteed to get the same answer with IDL and C).

Note also how the bit representation of the floats is not equal. Here they are in binary:

3f666666: 11111101100110011001100110011110
3f666667: 11111101100110011001100110011111

Just a single bit difference, but what a difference it is!

If you really want to test floats which arrived in a given array from various sources for "equality", you need to specify what "equality" means (if something other than the computer's quite naive definition of equality as "exact same representation in binary format"). You might try:

ARRAY_EQUAL(abs(a-a[0]) lt epsilon,1b)

where epsilon is your maximum allowed difference within which floats should be considered "equal". A good number might be (machar()).eps, i.e.:

```
IDL> print,abs(0.9 - (0.6+0.3)) lt (machar()).eps
   1
```

Ahh, all is right with the world.

JD

Subject: Re: How to test for a vector/matrix of constants?
Posted by JD Smith on Wed, 06 Dec 2006 23:54:29 GMT
View Forum Message <> Reply to Message

On Wed, 06 Dec 2006 23:23:26 +0000, Karl Schultz wrote:

> I checked around and couldn't find a reason. I'll put in a change request
> and see if we can doc them for next release. There is also a
> DIFFERENT_LENGTH keyword that will allow comparing arrays of differing
> lengths, using only the number of elements of the smaller array. These
> keywords are even exercised in our test suite.

Now there's a Christmas gift (with apologies to Ronn)!

Subject: Re: How to test for a vector/matrix of constants?
Posted by JD Smith on Thu, 07 Dec 2006 00:00:35 GMT
View Forum Message <> Reply to Message

On Wed, 06 Dec 2006 16:53:04 -0700, JD Smith wrote:

> ARRAY_EQUAL(abs(a-a[0]) lt epsilon,1b)

Or if you want to live on the edge with the newly unearthed ARRAY_EQUAL keywords:

```
IDL> a=[replicate(0.9,10),replicate(0.3+0.6,10)]
IDL> print,array_equal(a,a[0])
   0
IDL> print,array_equal(abs(a-a[0]), /LE, (machar()).eps)
   1
```

JD

Subject: Re: How to test for a vector/matrix of constants?
Posted by Paul Van Delst[1] on Fri, 08 Dec 2006 13:48:55 GMT
View Forum Message <> Reply to Message

Fï¿½LDY Lajos wrote:
>
> On Wed, 6 Dec 2006, Paul van Delst wrote:
>
>> From Braedley
>> if n_elements(uniq(array, sort(array))) eq 1 then ;flag it
>> -> No float comparison
>
> Do you think sort works without comparison? :-)

Well, I was going to qualify sort and uniq - but I figured people would grok that part since since they are documented IDL routines and, as such, I put them in the category of "Crikey, they better have done it right." I was specifically referring to user programmer written comparisons, not those of IDL commands (be they written in IDL or C).

cheers,

paulv

--
Paul van Delst Ride lots.
CIMSS @ NOAA/NCEP/EMC Eddy Merckx
Ph: (301)763-8000 x7748
Fax:(301)763-8545
