Subject: Re: newbie wants to enforce "array conservation"
Posted by Chris[6] on Tue, 22 Jul 2008 02:04:01 GMT
View Forum Message <> Reply to Message

On Jul 21, 2:51 pm, Tom Roche <tlro...@gmail.com> wrote:
> How to check that two arrays have the same totals, to some tolerance?
> and to throw an error if they don't? Especially if they are not the
> same size? (Apologies if these are FAQs, but I've googled and searched
> the online help and I'm not seeing it.) 3 more detailed questions
> below:
>
> I'm massaging netCDF files containing data on emissions over space and
> time. (Sometimes space is 2D, others 3D.) I want to ensure that I'm
> not corrupting the emissions, e.g. by conserving mass. I'm guessing a
> straightforward way to verify conservation is to check that, after
> each step in the overall process, the sum of emissions in the
> pre-massage file matches the sum of emissions in the post-massage
> file. I remember just enough of my undergraduate scientific-computing
> course to know that I want to match subject to some tolerance. I don't
> know IDL very well, but I can see
>
> http://idlastro.gsfc.nasa.gov/idl_html_help/ARRAY_EQUAL.html
>
> That should work for massages that don't change the size of the data:
> unfortunately I must also do regridding, which changes the size. For
> size-invariant massage I'm thinking I should do something like this:
>
> ; time is the first dimension in all these arrays
> timeIndex=1
> ; read pre-massage data into array "before"
> ; read post-massage data into array "after"
> ; total before
> before_total=TOTAL(before,timeIndex,/NAN)
> badval=WHERE(before_total eq 0, ct)
> IF ct ne 0 THEN before_total[badval]=0
> ; total after
> after_total=TOTAL(after,timeIndex,/NAN)
> badval=WHERE(after_total eq 0, ct)
> IF ct ne 0 THEN after_total[badval]=0
> ; check match including size
> IF not ARRAY_EQUAL(before_total, after_total, /NO_TYPECONV) THEN
> <throw error/>
>
> Does that look right? If so,
>
> 1 How does one typically throw a (non-GUI) error in IDL?
>
> 2 How does ARRAY_EQUAL handle tolerance? I was somewhat surprised that

> there was not, e.g., a keyword. Am I missing something?
>
> If not, how should size-invariant array matching be done?
>
> For size-variant massage (i.e. SIZE(input) ne SIZE(output)) one cannot
> use ARRAY_EQUAL, because it checks that array sizes match. (Or am I
> missing something?) So I'm thinking I should verify size-variant
> massages by just matching the scalar sums, e.g.
>
> ; read pre-massage data into array "before"
> ; read post-massage data into array "after"
> ; scalar total before
> before_total=TOTAL(before,/NAN)
> ; scalar total after
> after_total=TOTAL(after,/NAN)
> ; check match ignoring size
> diff=ABS(before_total-after_total)
> tolerance=<some small float/>
> IF diff gt tolerance THEN <throw error/>
>
> Does that look right? If so,
>
> 3 How does one determine a good tolerance value?
>
> If not, how should size-variant array matching be done?
>
> TIA, Tom Roche <Tom_Ro...@pobox.com>

Preliminary aside: lines like this

> badval=WHERE(after_total eq 0, ct)
> IF ct ne 0 THEN after_total[badval]=0

aren't necessary (you look to see if the array has any zeroes and, if it does, you set those zeroes to zero!)

I don't think you want array_equal as, like you mention, it checks for strict equality and not 'almost equality.' Your method of computing the scalar total before and after is a good approach- I would use this over array comparisons (unless you want to perform a spatially resolved check to see if flux is conserved).

I think the tolerance you use depends on the kinds of data massaging you are doing. Floating point operations should preserve calculations to at least 5-6 decimal places. So the error induced by summing n pixels after each has been corrupted by a floating point operation would be something like sqrt(n)*10^-5 or so. Anything smaller than

this may simply be due to finite machine precision. Errors much
greater than this might be a sign of a bug.

Also, you may be able to relax that restriction a bit if you know that
the uncertainty in your data is much larger than a part in 10^5.
Really, as long as your tolerance is some small fraction of the
uncertainty in the expected total, flux non-conservation (even if it
is due to a bug or sloppy calculation) doesn't matter.

chris

As far as error handling goes, read up on CATCH.

Subject: Re: newbie wants to enforce "array conservation"
Posted by Chris[6] on Tue, 22 Jul 2008 02:12:31 GMT
View Forum Message <> Reply to Message

On Jul 21, 4:04 pm, Chris <beaum...@ifa.hawaii.edu> wrote:
> On Jul 21, 2:51 pm, Tom Roche <tlro...@gmail.com> wrote:
>
>
>
>> How to check that two arrays have the same totals, to some tolerance?
>> and to throw an error if they don't? Especially if they are not the
>> same size? (Apologies if these are FAQs, but I've googled and searched
>> the online help and I'm not seeing it.) 3 more detailed questions
>> below:
>
>> I'm massaging netCDF files containing data on emissions over space and
>> time. (Sometimes space is 2D, others 3D.) I want to ensure that I'm
>> not corrupting the emissions, e.g. by conserving mass. I'm guessing a
>> straightforward way to verify conservation is to check that, after
>> each step in the overall process, the sum of emissions in the
>> pre-massage file matches the sum of emissions in the post-massage
>> file. I remember just enough of my undergraduate scientific-computing
>> course to know that I want to match subject to some tolerance. I don't
>> know IDL very well, but I can see
>
>> http://idlastro.gsfc.nasa.gov/idl_html_help/ARRAY_EQUAL.html
>
>> That should work for massages that don't change the size of the data:
>> unfortunately I must also do regridding, which changes the size. For
>> size-invariant massage I'm thinking I should do something like this:
>
>> ; time is the first dimension in all these arrays
>> timeIndex=1
>> ; read pre-massage data into array "before"

>> ; read post-massage data into array "after"
>> ; total before
>> before_total=TOTAL(before,timeIndex,/NAN)
>> badval=WHERE(before_total eq 0, ct)
>> IF ct ne 0 THEN before_total[badval]=0
>> ; total after
>> after_total=TOTAL(after,timeIndex,/NAN)
>> badval=WHERE(after_total eq 0, ct)
>> IF ct ne 0 THEN after_total[badval]=0
>> ; check match including size
>> IF not ARRAY_EQUAL(before_total, after_total, /NO_TYPECONV) THEN
>> <throw error/>
>
>> Does that look right? If so,
>
>> 1 How does one typically throw a (non-GUI) error in IDL?
>
>> 2 How does ARRAY_EQUAL handle tolerance? I was somewhat surprised that
>> there was not, e.g., a keyword. Am I missing something?
>
>> If not, how should size-invariant array matching be done?
>
>> For size-variant massage (i.e. SIZE(input) ne SIZE(output)) one cannot
>> use ARRAY_EQUAL, because it checks that array sizes match. (Or am I
>> missing something?) So I'm thinking I should verify size-variant
>> massages by just matching the scalar sums, e.g.
>
>> ; read pre-massage data into array "before"
>> ; read post-massage data into array "after"
>> ; scalar total before
>> before_total=TOTAL(before,/NAN)
>> ; scalar total after
>> after_total=TOTAL(after,/NAN)
>> ; check match ignoring size
>> diff=ABS(before_total-after_total)
>> tolerance=<some small float/>
>> IF diff gt tolerance THEN <throw error/>
>
>> Does that look right? If so,
>
>> 3 How does one determine a good tolerance value?
>
>> If not, how should size-variant array matching be done?
>
>> TIA, Tom Roche <Tom_Ro...@pobox.com>
>
> Preliminary aside: lines like this
>

>> badval=WHERE(after_total eq 0, ct)
>> IF ct ne 0 THEN after_total[badval]=0
>
> aren't necessary (you look to see if the array has any zeroes and, if
> it does, you set those zeroes to zero!)
>
> I don't think you want array_equal as, like you mention, it checks for
> strict equality and not 'almost equality.' Your method of computing
> the scalar total before and after is a good approach- I would use this
> over array comparisons (unless you want to perform a spatially
> resolved check to see if flux is conserved).
>
> I think the tolerance you use depends on the kinds of data massaging
> you are doing. Floating point operations should preserve calculations
> to at least 5-6 decimal places. So the error induced by summing n
> pixels after each has been corrupted by a floating point operation
> would be something like sqrt(n)*10^-5 or so. Anything smaller than
> this may simply be due to finite machine precision. Errors much
> greater than this might be a sign of a bug.
>
> Also, you may be able to relax that restriction a bit if you know that
> the uncertainty in your data is much larger than a part in 10^5.
> Really, as long as your tolerance is some small fraction of the
> uncertainty in the expected total, flux non-conservation (even if it
> is due to a bug or sloppy calculation) doesn't matter.
>
> chris
>
> As far as error handling goes, read up on CATCH.

Another potential pitfall:
Make sure you know what the units of emission are. If, for example, you regrid emission data on a grid that has 4x larger pixels, and the emission is something like power/solid angle, then you want the sum of the regridded array to be 4x smaller. the total power in each image is the value per pixel times the solid area of the pixel, summed up. Simply summing two arrays on different scales doesn't do the 'multiply by pixel size' step

chris

Subject: Re: newbie wants to enforce "array conservation"
Posted by Paul Van Delst[1] on Tue, 22 Jul 2008 13:47:11 GMT
View Forum Message <> Reply to Message

Tom Roche wrote:
> How to check that two arrays have the same totals, to some tolerance?

> and to throw an error if they don't? Especially if they are not the
> same size? (Apologies if these are FAQs, but I've googled and searched
> the online help and I'm not seeing it.) 3 more detailed questions
> below:
>
> I'm massaging netCDF files containing data on emissions over space and
> time. (Sometimes space is 2D, others 3D.) I want to ensure that I'm
> not corrupting the emissions, e.g. by conserving mass. I'm guessing a
> straightforward way to verify conservation is to check that, after
> each step in the overall process, the sum of emissions in the
> pre-massage file matches the sum of emissions in the post-massage
> file. I remember just enough of my undergraduate scientific-computing
> course to know that I want to match subject to some tolerance. I don't
> know IDL very well, but I can see
>
> http://idlastro.gsfc.nasa.gov/idl_html_help/ARRAY_EQUAL.html
>
> That should work for massages that don't change the size of the data:
> unfortunately I must also do regridding, which changes the size. For
> size-invariant massage I'm thinking I should do something like this:
>
> ; time is the first dimension in all these arrays
> timeIndex=1
> ; read pre-massage data into array "before"
> ; read post-massage data into array "after"
> ; total before
> before_total=TOTAL(before,timeIndex,/NAN)
> badval=WHERE(before_total eq 0, ct)
> IF ct ne 0 THEN before_total[badval]=0
> ; total after
> after_total=TOTAL(after,timeIndex,/NAN)
> badval=WHERE(after_total eq 0, ct)
> IF ct ne 0 THEN after_total[badval]=0
> ; check match including size
> IF not ARRAY_EQUAL(before_total, after_total, /NO_TYPECONV) THEN
> <throw error/>
>
> Does that look right? If so,
>
> 1 How does one typically throw a (non-GUI) error in IDL?

CATCH is what you want. Something like:

```
FUNCTION my_func, arg1, arg2, etc..

  ; Define error results
  SUCCESS = 0 ; You should stick these in an include
  FAILURE = 1 ; file and use it everywhere
```

```
  ; Define error handler
  CATCH, err_stat
  IF ( err_stat NE 0 ) THEN BEGIN
    CATCH, /CANCEL
    MESSAGE, !ERROR_STATE.MSG, /CONTINUE
    RETURN, FAILURE
  ENDIF

  ; Do stuff...sum array...difference array..etc....

  ; Check array difference and throw error if required
  IF ( diff GT tolerance ) THEN $
    MESSAGE, 'Array difference is > tolerance', /NONAME, /NOPRINT

  ; Test successful. Do some more stuff....

  RETURN, SUCCESS
END
```

> 2 How does ARRAY_EQUAL handle tolerance? I was somewhat surprised that
> there was not, e.g., a keyword. Am I missing something?

Dunno. I'd roll my own. As the other poster, Chris, said: your summation difference may be much less than your data uncertainty.

But, if it's not, you may want to look into first sorting your arrays and then using a compensated summation algorithm (like Kahan's) to sum the sorted numbers. That should minimise any summation error accumulation and, depending on the algorithm, may also give you an estimate of how large the accumulation is. You may be able to use those numbers as a tolerance of some sort.

Note that sorting, then summing in this fashion, can be slooow.

>
> If not, how should size-invariant array matching be done?
>
> For size-variant massage (i.e. SIZE(input) ne SIZE(output)) one cannot
> use ARRAY_EQUAL, because it checks that array sizes match. (Or am I
> missing something?) So I'm thinking I should verify size-variant
> massages by just matching the scalar sums, e.g.
>
> ; read pre-massage data into array "before"
> ; read post-massage data into array "after"
> ; scalar total before
> before_total=TOTAL(before,/NAN)
> ; scalar total after

> after_total=TOTAL(after,/NAN)
> ; check match ignoring size
> diff=ABS(before_total-after_total)
> tolerance=<some small float/>
> IF diff gt tolerance THEN <throw error/>
>
> Does that look right? If so,
>
> 3 How does one determine a good tolerance value?

That's probably the hardest thing. Depends on how big your numbers are, their dynamic range, etc. But, see #2 for an idea. Trial and error can work too. :o)

If the numbers must agree to within a certain precision, then you might want to look into using MACHAR() to get an estimate for your platform.

cheers,

paulv
