Subject: Problem with suppression of dimension
Posted by jeyadev on Thu, 14 Dec 2000 00:25:50 GMT
View Forum Message <> Reply to Message

This may be a FAQ candidate, but I cannot find anything in the PV-Wave manuals or in David's (fine) book.

I am having problem with the following code fragment:

```
......
......

nsteps = 5
  npts = 1    ; number of data points within each layer
 ncols = 5

dpt = {fullrow, cood:fltarr(4), colcode: ''}
fulldata = replicate(dpt, nsteps*npts*ncols)

avdata = fltarr(4,nsteps*ncols)

for i = 0,nsteps*ncols-1 do begin
    j = i*npts
    for k = 0,3 do avdata(k,i) = avg(fulldata(j:j+npts-1).cood(k))
    print, Format="(4(4x,f8.4), 5x, a1)", avdata(*, i),  cols(i/nsteps)
endfor

........
........
```

This basic job of the code is to take a 2-d array of dimensions (4, nsteps*npts*ncols) and then reduce this dataset to a another 2-d array of dimensions (4,nsteps*ncols) by averaging over blocks of 'npts' rows. (For arcane reasons, the 'colcode' variable is present in every row of the raw matrix and hence the structure.)

This works fine for npts > 1 (that is the usual case and I have been using the procedure for a while), but if, as I now need, npts = 1, I get an error from the AVG procedure as the argument being passed is not an array:

```
    Variable must be an array, name= ARRAY, routine AVG.
```

Now for some trouble shooting ....

With i = j = 0 and setting k = 0, for example

```
   WAVE> i = 0 & j = 0
```

```
WAVE> avdata = fltarr(4,nsteps*ncols)
WAVE> k = 0
WAVE> info, fulldata(j:j+npts-1).cood(k)
<Expression>    FLOAT     =       28.3790
```

On the other hand with npts = 2, the same gives

```
WAVE> npts = 2
WAVE> info, fulldata(j:j+npts-1).cood(k)
<Expression>    FLOAT     = Array(2)
```

So the question is, how do I force the float to be an array of length 1 when npts = 1?

I would like to avoid using an

```
if(npts eq 1)then avdata = fulldata.cood(0:2) else ....
```

if I can.

I seem to remember reading about this issue of avoiding the suppression of trailing degenerate dimesions, but I cannot find it in my private hints database (collected from this NG), online anywhere, in the manual and not even in David's superb tome.

Any pointers will be appreciated.

I am using PV-WAVE CL Version 6.01 (sun4 solaris sparc).

Needless to add, if there is a completely different way of accomplishing the row-wise averaging, I will gladly throw my method in the dustbin. :-)

thanks

--

Surendar Jeyadev jeyadev@wrc.xerox.com

Subject: Re: Problem with suppression of dimension
Posted by Craig Markwardt on Fri, 15 Dec 2000 16:31:16 GMT

Hi Surendar--

Your post brings up several points.

1) When you extract a one-element field from a structure, no matter how many dimensions it has, it will come back as a scalar. I have argued for quite a while that this is a big fat bug in IDL.

   This is a little different than IDL's "standard" and documented behavior. IDL is *supposed* to drop any trailing dimensions of 1, which can be aggravating enough, but in this case it drops *all* dimensions!

   It would be interesting to know if this has changed in IDL 5.4.

2) Functions like AVG, REFORM and SMOOTH do not accept scalars, even though this is a trival case. On the other hand TOTAL does accept scalar arguments. Again, why such inconsistency?

I thought I had another point, but enough negativism. RSI/Kodak will obviously never change :-)

To be constructive, perhaps the simplest change to your code would be to convert from AVG(...) to TOTAL(...)/N_VALUES. This requires the least amount of change, and as I point out, TOTAL *does* accept scalar arguments.

It should be fairly easy to vectorize this whole thing however. Try this one on for size:

```
array = reform(fulldata.cood, 4, npts, nsteps*ncols)
avdata = rebin(array, 4, 1, nsteps*ncols)
avdata = reform(avdata, 4, nsteps*ncols, /overwrite)
```

No FOR loops are required. The key is to reform the array so that the NPTS dimension becomes explicit, and then use the inherent averaging property of REBIN to reduce that dimension. If memory is critical then judicious use of temporary can help.

Despite the fact that a solution exists here, I still argue my points (1) and (2) above are important for the internal consistency of IDL.

Craig

jeyadev@wrc.xerox.com (Surendar Jeyadev) writes:

> This may be a FAQ candidate, but I cannot find anything in the PV-Wave
> manuals or in David's (fine) book.
>
> I am having problem with the following code fragment:
>
> ......
> ......
>
> nsteps = 5
> npts = 1 ; number of data points within each layer
> ncols = 5
>
> dpt = {fullrow, cood:fltarr(4), colcode: ''}
> fulldata = replicate(dpt, nsteps*npts*ncols)
>
> avdata = fltarr(4,nsteps*ncols)
>
> for i = 0,nsteps*ncols-1 do begin
> j = i*npts
> for k = 0,3 do avdata(k,i) = avg(fulldata(j:j+npts-1).cood(k))
> print, Format="(4(4x,f8.4), 5x, a1)", avdata(*, i), cols(i/nsteps)
> endfor
>
> ....
> ....
>
> This basic job of the code is to take a 2-d array of dimensions
> (4, nsteps*npts*ncols) and then reduce this dataset to a another
> 2-d array of dimensions (4,nsteps*ncols) by averaging over blocks
> of 'npts' rows. (For arcane reasons, the 'colcode' variable is
> present in every row of the raw matrix and hence the structure.)
>
> This works fine for npts > 1 (that is the usual case and I have
> been using the procedure for a while), but if, as I now need,
> npts = 1, I get an error from the AVG procedure as the argument
> being passed is not an array:
>
> Variable must be an array, name= ARRAY, routine AVG.
>
> Now for some trouble shooting ....
>
> With i = j = 0 and setting k = 0, for example
>
> WAVE> i = 0 & j = 0
> WAVE> avdata = fltarr(4,nsteps*ncols)
> WAVE> k = 0
> WAVE> info, fulldata(j:j+npts-1).cood(k)
> <Expression> FLOAT = 28.3790

>
>
>
> On the other hand with npts = 2, the same gives
>
> WAVE> npts = 2
> WAVE> info, fulldata(j:j+npts-1).cood(k)
> <Expression> FLOAT = Array(2)
>
>
>
> So the question is, how do I force the float to be an array of
> length 1 when npts = 1?
>
> I would like to avoid using an
>
> if(npts eq 1)then avdata = fulldata.cood(0:2) else ....
>
> if I can.
>
> I seem to remember reading about this issue of avoiding the
> suppression of trailing degenerate dimesions, but I cannot find
> it in my private hints database (collected from this NG), online
> anywhere, in the manual and not even in David's superb tome.
>
> Any pointers will be appreciated.
>
> I am using PV-WAVE CL Version 6.01 (sun4 solaris sparc).
>
> Needless to add, if there is a completely different way of
> accomplishing the row-wise averaging, I will gladly throw
> my method in the dustbin. :-)
>
> thanks
>
>
> --
>
> Surendar Jeyadev jeyadev@wrc.xerox.com

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