Subject: Re: What are the rules for automatic removal of singleton dimensions, and can I have a way of disabling them, please?
Posted by tom.grydeland on Wed, 16 Jan 2013 11:09:45 GMT
View Forum Message <> Reply to Message

Hi again,

On Monday, December 24, 2012 12:01:24 AM UTC+1, Tom Grydeland wrote:
> I was trying to visualize subsections of a windowing function when this bit me.

Today, it bit me again.

I was fiddling with a small routine to create dense grid index arrays (akin to those the 'mgrid' object from NumPy create), using a combination of adding extra indices and transpose() to create my matrices. If I could trust IDL not to strip dimensions, I could write this cleanly as:

(ax, ay and az are all vectors of length >= 1 at this point)

```
xout = ax[*,ay,az]
yout = transpose(ay[*,ax,az], [1,0,2])
zout = transpose(az[*,ax,ay], [1,2,0])
```

Unfortunately, the gratuitous stripping of dimensions even inside the indexing expression means that the array given to transpose() doesn't have three dimensions anymore, and I have to resort to this mess, which is much less readable and much harder to maintain.

```
out_dims = [n_elements(ax), n_elements(ay), n_elements(az)]
xout = reform(ax[*,ay,az], out_dims)
yout = transpose(reform(ay[*,ax,az], out_dims[[1, 0, 2]]), [1, 0, 2])
zout = transpose(reform(az[*,ax,ay], out_dims[[2, 0, 1]]), [1, 2, 0])
```

(Of course, in line with Mats Löfdahl's suggestion earlier, TRANSPOSE could also be modified to add back singleton dimensions to its input array as necessary.)

--Tom Grydeland