Subject: Re: What are the rules for automatic removal of singleton dimensions, and can I have a way of disabling them, please?
Posted by Jeremy Bailin on Thu, 17 Jan 2013 14:58:34 GMT
View Forum Message <> Reply to Message

On 1/17/13 2:11 AM, Tom Grydeland wrote:
> On Wednesday, January 16, 2013 12:09:45 PM UTC+1, Tom Grydeland wrote:
>
> Since I'm talking to myself here, I might as well reply to myself also.
>
>> (Of course, in line with Mats Lï¿½fdahl's suggestion earlier, TRANSPOSE could also be modified to add back singleton dimensions to its input array as necessary.)
>
> Having written this, I decided to go ahead and do it that way also, just to see. It feels a little bit like I'm fixing IDL's bugs for them instead of working on the problems I'm actually paid to solve. I tried writing MY_TRANSPOSE so that it can be a drop-in replacement of TRANSPOSE, i.e. it shall give exactly the same results as TRANSPOSE for every input that TRANSPOSE will accept.
>
> The tedium of adding singleton dimensions was factored out to a function ATLEAST_N_DIM, which modifies and returns its first argument.
>
> I'm putting these functions out there for anyone to see or use as they see fit.
>
> I am still hopeful that I can get a COMPILE_OPT that would (within a certain scope) disable stripping of trailing singleton dimensions, so I wouldn't have to resort to such hackery or the replacement of base IDL functionality.
>
> Stylistic comments, suggestions, etc?
>
>
> ;+
> ; Small utility that ensures its input array has dimensionality at least N.
> ; Modifies _and_ returns its input, without making copies.
> ;
> ; :Params:
> ; a: in, out
> ; array to modify dimensionality of
> ; n: in, type=integer
> ; minimum number of dimensions for result
> ;
> ; :Returns:
> ; a, with zero or more singleton dimensions appended
> ;
> ; :See also:
> ; MY_TRANSPOSE, DENSEGRID
> ;-
> function atleast_n_dim, a, n
> adim = size(a, /dimensions)

```
>       if n_elements(adim) ge n then return, a
>
>       adim = [adim, make_array(n-n_elements(adim), value=1)]
>       a = reform(a, adim, /overwrite)
>       return, a
>  end
>
>  ;+
>  ; Version of TRANSPOSE that will add singleton dimensions if needed
>  ;
>  ; :See also:
>  ;   ATLEAST_N_DIM, DENSEGRID
>  ;-
>  function my_transpose, arr, perm
>       ;; defer simple cases to native TRANSPOSE
>       if n_elements(perm) eq 0 then return, transpose(arr)
>
>       adims = size(arr, /dimensions)
>       np = max(perm)+1
>       na = n_elements(adims)
>       if np le na then return, transpose(arr, perm)
>
>       narr = transpose(atleast_n_dim(arr, np), perm)
>       ;; set dimensions of arr back to what they were
>       arr = reform(arr, adims, /overwrite)
>       return, narr
>  end
>
```

Looks good to me - I will definitely use them (at least the utility function).

-Jeremy.
