
Subject: Re: Automatic truncation of trailing dimension.....

Posted by [R.Bauer](#) on Wed, 28 Nov 2001 01:05:12 GMT

[View Forum Message](#) <> [Reply to Message](#)

"Pavel A. Romashkin" wrote:

```
>
> Oh, of course should be
>
> s = size(x)
> help, reform(x[*,*,1], s[1], 1)
>
> Duh!
>
> "Pavel A. Romashkin" wrote:
>>
>> Of course,
>>
>> s = size(x)
>> help, reform(x, s[1], 1)
```

Dear Pavel,

I have the same problems as you.

Therefore I have written a set routine this is attached.

If RSI won't change the state of removing last dimension 1 I like to
have a
compile option to extend these rule.

Everytime I have defined a three dimensional data set e.g. O3 in LAT and
LON and 1 time
the time is missing and this gives problems if I write HDF or netCDF
datafiles or
if I like to concatinate on the last dimension.

This was one of the reasons I have written dref and set.

It is so terrible if I have once changed my variable by reform and then
all is lost
if I initialize another variable like a=b then a is wrong while b is ok.

regards

Reimar

--
Reimar Bauer

Institut fuer Stratosphaerische Chemie (ICG-1)

Forschungszentrum Juelich

email: R.Bauer@fz-juelich.de

<http://www.fz-juelich.de/icg/icg1/>

=====

a IDL library at ForschungsZentrum Juelich

http://www.fz-juelich.de/icg/icg1/idl_icglib/idl_lib_intro.h.html

<http://www.fz-juelich.de/zb/text/publikation/juel3786.html>

=====

read something about linux / windows

<http://www.suse.de/de/news/hotnews/MS.html>

; Copyright (c) 2001, Forschungszentrum Juelich GmbH ICG-1

; All rights reserved.

; Unauthorized reproduction prohibited.

; This software may be used, copied, or redistributed as long as it is not

; sold and this copyright notice is reproduced on each copy made. This

; routine is provided as is without any express or implied warranties

; whatsoever.

;

;

;

; NAME:

; set

;

; PURPOSE:

; This functions dereferences pointers. If last dimension is 1 this is returned and not the
; standard IDL Array without last dimension 1.

; If value is a array of pointer the values are concatenated by concatenate_arrays at the last
dimension

; If value isn't a pointer this value is returned but with the right dimensions.

;

;

;

; CATEGORY:

; PROG_TOOLS

;

; CALLING SEQUENCE:

; result=set(value,[/free])

;

; INPUTS:

; value: the pointer to exchange to it's value

;

;

```

; KEYWORD PARAMETERS:
; free: if set the pointer is purged from memory
;
; PROCEDURE:
; This routine should be used if it could be possible that's a value is a value or a pointer
; of the value. In difference to the idl dereference operator * this routine returns the
; right dimensions back if last dimension are 1.
; This function returns concatenated array values if a vector of pointer is submitted.
; Therefore the dimensions should be conform to do this.
;
; EXAMPLE:
; a=10
; help,set(a) & print,set(a,/free)
; A           INT      =      10
; 10
;
;
;
; a=ptr_new(10)
; help,set(a) & print,set(a,/free)
; A           INT      =      10
; 10
;
;
;
;
; a=reform(indgen(10),10,1)
; X=set(a)
; y=a
; help,set(a) & print,set(a,/free) & help, x & help,y
; A           INT      = Array[10, 1]
; 0   1   2   3   4   5   6   7   8   9
; X           INT      = Array[10,1]
; Y           INT      = Array[10]
;
;
;
; a=ptr_new(reform(indgen(10),10,1))
; X=set(a)
; Y=a
; help,set(a) & print,set(a,/free) & help, x & help,y
; A           INT      = Array[10, 1]
; 0   1   2   3   4   5   6   7   8   9
; X           INT      = Array[10, 1]
; Y           POINTER  = <PtrHeapVar1388>
;
;
;
;
;
;
; MODIFICATION HISTORY:

```

```
; Written by: R.Bauer (ICG-1), 2000-Jul-09
; 2001-11-18 : feature with last dimension eq 1 added
;           : feature concatenate_arrays added: always on last dimension concatenation is done
;           : feature added : if value is no pointer this value is returned with the right dimensions
;-
```

```
FUNCTION set,value,free=free,_extra=p_key
```

```
IF SIZE(value[0],/TName) EQ 'POINTER' THEN BEGIN
```

```
n_ptr=SIZE(value,/N_ELEMENTS)
```

```
sz=SIZE(*value,/struct)
```

```
IF sz.n_dimensions LE 1 THEN BEGIN
```

```
  IF N_ELEMENTS(result) EQ 0 THEN result=(*value)
```

```
  ENDIF ELSE BEGIN
```

```
    IF sz.dimensions[sz.n_dimensions-1] EQ 1 THEN BEGIN
```

```
      IF N_ELEMENTS(result) EQ 0 THEN
```

```
result=REFORM((*value),sz.dimensions[0:sz.n_dimensions-1])
```

```
    ENDIF ELSE IF N_ELEMENTS(result) EQ 0 THEN result=(*value)
```

```
  ENDELSE
```

```
  IF KEYWORD_SET(free) THEN PTR_FREE,value
```

```
  RETURN,result
```

```
ENDIF ELSE BEGIN
```

```
sz=SIZE(value,/struct)
```

```
IF sz.n_dimensions GE 2 THEN IF sz.dimensions[sz.n_dimensions-1] EQ 1 THEN BEGIN
```

```
  RETURN,REFORM(value,sz.dimensions[0:sz.n_dimensions-1])
```

```
ENDIF ELSE RETURN,value
```

```
RETURN,value
```

```
ENDELSE
```

```
END
```

File Attachments

- 1) [set.pro](#), downloaded 89 times
-