Subject: Re: Concatenating arrays across chosen dimension
Posted by R.Bauer on Wed, 26 Jun 2002 07:18:04 GMT
View Forum Message <> Reply to Message

Dick Jackson wrote:
>
> "Randall Skelton" <rhskelto@atm.ox.ac.uk> wrote in message
> news:Pine.LNX.4.33.0206251749570.28170-100000@mulligan.atm.o x.ac.uk...
>
>> Ok... I have to ask. Is there actually a nice, clean way to concatenate
>> multidimensional arrays in IDL?
>>
>> a = make_array(2,2,2,2)
>> b = make_array(2,2,2,5)
>>
>> data1 = [[[[a]]] , [[[b]]]]
>>
>> Obviously the above fails, but what is the solution? Surely some
>> combination of rebin/reform...

Dear Dick,

we have written a general routine to concatinate on each dimension you want.

http://www.fz-juelich.de/icg/icg-i/idl_icglib/idl_source/idl_html/dbase/download/concatenate_arrays.tar.gz
http://www.fz-juelich.de/icg/icg-i/idl_icglib/idl_source/idl_html/dbase/download/concatenate_arrays.sav

```
; NAME:
;       concatenate_arrays
;
; PURPOSE:
;       This function concatenates two arrays with arbitrary but similar
dimensions.
;
; CATEGORY:
;   PROG_TOOLS
;
; CALLING SEQUENCE:
;   result=CONCATENATE_ARRAYS(a,b,dim)
;
; INPUTS:
;   a,b : arrays of similar structure
```

```
;   dim : dimension number for concatenation; first dimension has number
0
;
; OUTPUTS:
;   concatenated array
;
; RESTRICTIONS:
;   !No error tests are performed at the current stage!
;
; EXAMPLE:
;   a=indgen(1,2,3,4,5)
;   b=indgen(1,4,3,4,5)
;   help,a,b,concatenate_arrays(a,b,1)
;
;   A               INT       = Array[1, 2, 3, 4, 5]
;   B               INT       = Array[1, 4, 3, 4, 5]
;   <Expression>    INT       = Array[1, 6, 3, 4, 5]
```

For further routines and licensing please have a look at

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

regards

Reimar

> Well, I have to say I don't know *why* that one fails, since this works
> fine:
>
> IDL> a = make_array(2,2,2)
> IDL> b = make_array(2,2,5)
> IDL> help, [[[a]], [[b]]]
> <Expression>   FLOAT   = Array[2, 2, 7]
>
> ... and we're a long way from the 8-dimension limit on arrays.
>
> In any case, to concatenate on the *last* dimension in this case:
>
> c = Reform([a[*], b[*]], [2,2,2,7])
>
> To do this in general takes a little more hacking, dimension juggling in
> particular. I couldn't resist the challenge, so I whipped up the attached
> Concat.pro.
>
> Examples:
> IDL> Help, Concat(LIndGen(2,3,4), -LIndGen(5,3,4), Dim=0)

```
> <Expression>    LONG      = Array[7, 3, 4]
> IDL> Help, Concat(LIndGen(2,3,4), -LIndGen(2,5,4), Dim=1)
> <Expression>    LONG      = Array[2, 8, 4]
> IDL> Help, Concat(LIndGen(2,3,4), -LIndGen(2,3,5)) ; assumes last dimension
> <Expression>    LONG      = Array[2, 3, 9]
>
> Code is copied here for convenience. Any comments or corrections are
> welcome!
>
> =====
>
> FUNCTION Concat, $                  ; Return concatenation of two arrays
>    a, $                             ; First array
>    b, $                             ; Second array
>    Dimension=dim                    ; Dimension along which to
> concatenate
>                                     ; (counting from 0, defaults to
> *last*
>                                     ; dimension of arrays)
> ;;    Examples:
> ;;    IDL> Help, Concat(LIndGen(2,3,4), -LIndGen(5,3,4), Dim=0)
> ;;    <Expression>    LONG      = Array[7, 3, 4]
> ;;    IDL> Help, Concat(LIndGen(2,3,4), -LIndGen(2,5,4), Dim=1)
> ;;    <Expression>    LONG      = Array[2, 8, 4]
> ;;    IDL> Help, Concat(LIndGen(2,3,4), -LIndGen(2,3,5))
> ;;    <Expression>    LONG      = Array[2, 3, 9]
> ;;    (use Print with these to see actual results)
>
> ;;    Assuming here that a and b are of same dimensions except perhaps for
> ;;    dimension 'dim', which is assumed to be within range.
> ;;    Testing and error handling for this is left as an exercise for the
> ;;    reader. :-)
>
> nDims = Size(a, /N_Dimensions)
> IF N_Elements(dim) EQ 0 THEN dim = nDims-1
>
> aDims = Size(a, /Dimensions)
> bDims = Size(b, /Dimensions)
>
> ;;    Figure out desired dimensions of result
>
> resultDims = aDims
> resultDims[dim] = aDims[dim]+bDims[dim]
>
> ;;    Make a vector of dimension indices with concatenation dimension *last*
>
> transposeDimOrder = [Where(IndGen(nDims) NE dim), dim]
>
```

> ;; Juggle dimensions by transposing a and b to put desired concatenation
> ;; dimension last, then take all elements together into one vector
>
> joinedVector = [(Transpose(a, transposeDimOrder))[*], $
> (Transpose(b, transposeDimOrder))[*]]
>
> ;; Reform the vector to an array of the right size with juggled
> dimensions
>
> juggledResult = Reform(Temporary(joinedVector),
> resultDims[transposeDimOrder])
>
> ;; Un-juggle the dimensions to give final result
>
> Return, Transpose(Temporary(juggledResult), Sort(transposeDimOrder))
>
> END
>
> =====
>
> Cheers,
> --
> -Dick
>
> Dick Jackson / dick@d-jackson.com
> D-Jackson Software Consulting / http://www.d-jackson.com
> Calgary, Alberta, Canada / +1-403-242-7398 / Fax: 241-7392

--
Reimar Bauer

Institut fuer Stratosphaerische Chemie (ICG-I)
Forschungszentrum Juelich
email: R.Bauer@fz-juelich.de
 ---------- -------
  a IDL library at ForschungsZentrum Juelich
 http://www.fz-juelich.de/icg/icg1/idl_icglib/idl_lib_intro.h tml
 ========== =======