
Subject: Skeleton of a new array slicing function

Posted by [Martin Schultz](#) on Thu, 20 May 1999 07:00:00 GMT

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

Liam Gumley wrote:

>
> [...]

I thought this would be something to get started on right away. Before I rush home now to celebrate my little daughter's birthday, I'll just "throw" my version of arrex at you. lease find it attached. It doesn't do anything useful yet, but it should have all of the error checking and setting of default parameters. Whoever finishes, please let me know, i.e. post it here.

Thanks,
Martin.

|||||\\\-----//|
Martin Schultz, DEAS, Harvard University, 29 Oxford St., Pierce 109,
Cambridge, MA 02138 phone (617) 496 8318 fax (617) 495 4551
e-mail mgs@io.harvard.edu web http://www-as/people/staff/mgs/

```
function arrex,Array,Starti,Endi,Stride,Reform=DoReform
```

```
    ; return quietly if array is undefined or a scalar  
    if (n_elements(Array) lt 2) then $  
        return,Array
```

```
    ; get array dimensions  
    NDims = size( Array, /N_Dimensions )  
    Dims = size( Array, /Dimensions )
```

```
    ; ### DEBUG  
    help,Array
```

```
    ; check parameters and set defaults  
    ; if parameter is not present use  
    ; 0 as default for Starti  
    ; Dim[i] as default for Endi
```

```

; 1 as default for Stride
; if parameter has less dimensions than Array fill
; remainder with defaults
; if Starti or Endi are negative (-1) use defaults
; A Stride of -1 reverses the respective array dimension
; NOTE that Start indices may be smaller than end indices
; in this case.

```

```

DStart = lonarr(NDims) ; array with 0
DEnd   = Dims - 1L    ; last elements
DStride = lonarr(NDims) + 1L ; every element

```

```

if (n_elements(Starti) gt 0) then begin
  if (n_elements(Starti) gt NDims) then begin
    message,'START contains more numbers than ARRAY dimensions!', $
      /Continue
    return, Array
  endif
  ; since default is 0 anyway, only copy values in Starti that
  ; are greater zero
  ind = where(Starti gt 0L)
  if (ind[0] ge 0) then $
    DStart[ind] = ( Starti[ind] < Dims[ind] )
endif
; ### DEBUG
print,'Start indices: ',fix(DStart)

```

```

if (n_elements(Endi) gt 0) then begin
  if (n_elements(Endi) gt NDims) then begin
    message,'END contains more numbers than ARRAY dimensions!', $
      /Continue
    return, Array
  endif
  ; Only copy values in Endi that are at least zero
  ind = where(Endi ge 0L)
  if (ind[0] ge 0) then $
    DEnd[ind] = ( Endi[ind] < Dims[ind] )
endif
; ### DEBUG
print,'End indices: ',fix(DEnd)

```

```

if (n_elements(Stride) gt 0) then begin
  if (n_elements(Stride) gt NDims) then begin
    message,'STRIDE contains more numbers than ARRAY dimensions!', $
      /Continue
    return, Array
  endif
endif

```

```

endif
; Only copy values in Endi that are not zero
ind = where(Stride ne 0L)
if (ind[0] ge 0) then $
    DStride[ind] = Stride[ind]
endif else begin ; change default to include negative strides
    ; where ENDI It STARTI
    ind = where(DEnd lt DStart)
    if (ind[0] ge 0) then $
        DStride[ind] = -1L
    endelse

; ### DEBUG
print,'Stride: ',fix(DStride)

; For extraction, change Stride to all positive, but save
; info which dimensions shall be reversed
NeedReverse = intarr(NDims)
ind = where(DStride lt 0)
if (ind[0] ge 0) then $
    NeedReverse[ind] = 1
DStride = abs(DStride)

; ***** actual extraction goes here *****
Result = Array

; Reverse as needed
; ** NOTE: just discovered that IDL's reverse function only handles
; ** up to 3-dimensional arrays !!
if (NDims gt 3) then begin
    message,'More than 3 dimensions! Cannot reverse!','/Continue
    return,Result
endif

; Apply reform if requested
if (keyword_set(DoReform)) then $
    Result = reform(temporary(Result))

return,Result

```

end

File Attachments

1) [arrex.pro](#), downloaded 85 times
