Subject: Re: printing an array from pointers
Posted by Paul Van Delst[1] on Wed, 29 Mar 2006 16:06:48 GMT
View Forum Message <> Reply to Message

bressert@gmail.com wrote:
> Hi Peter,
>
> Another question, since I have ran into a new bump. Is there a way to
> say
>
> arr = fltarr(A,8)
>
> where A is a number that fluctuates? So rather than stating that arr is
> A rows long, it is a number determined by the total output of the for
> loop? For example,
>
> arr = fltarr(150,8)
>
> will be sufficient in gathering all the 'for' outputs, but I will have
> trailing zeros that have not been assigned an output value. Using UNIQ
> or an 'if' to get rid of the zeros in the array does not work, since
> some of the output from the 'for' loop is zero. This was the original
> reason why I used the pointers, since there was no requirement of
> predetermination of the number of rows. Any suggestions or ideas would
> be greatly appreciated. Thanks again for the help.

Keep in mind that the first index is the one that increments in contiguous memory (opposite to C) so maybe arr=fltarr(8,a) is required.

But, regardless, you must know the maximum limit of the for loop in advance, no? In other examples posted in this thread, we';ve seen:

```
arr = fltarr(10,8)
for i = 0,9 do begin
    ...
   arr[i,*] = (some 1 by 8 vector)
endfor
```

That can be re-written as:

```
loop_limit = func_to_compute_loop_limit()
arr = fltarr(8,loop_limit)  ! More efficient that (loop_limit,8)
for i=0,loop_limit-1 do begin
   ....
   arr[*,i] = (some 1 by 8 vector)
endfor
```

If you don;t know the loop limit in advance you can do two things:

1) Concatenation

```
i=-1
WHILE (some condition) DO BEGIN
  i++
  .....
  if(i eq 0) then $
    arr = (some 1 by 8 vector) $
  else $
    arr = [ [arr],[(some 1 by 8 vector)] ]
ENDWHILE
```

but this can be very slow if you concatenate a lot of stuff. For small arrays (low values of i) it's great. For large values of i, not so good.

2) Truncation

```
arr=fltarr(8,big_enough)
FOR i=0,big_enough-1 DO BEGIN
  ...
  arr[*,i] = (some 1 by 8 vector)
  if ( some condition ) then begin
    arr = arr[*,0:i]
    BREAK
  endif
ENDFOR
```

this is nearly always faster.

Note: all the above typed off top of head, so no guarantees. Test. Especially the concatenate stuff. I can never remember how many groups of [[][]]'s to use for multi-D arrays.

paulv

--
Paul van Delst
CIMSS @ NOAA/NCEP/EMC