Subject: Re: Selecting groups of 5 coords from a set of n (nC5)
Posted by Paolo Grigis on Fri, 03 Mar 2006 17:19:34 GMT

View Forum Message <> Reply to Message

Some time ago I needed combinations and I came up with
this function. Example:

```
IDL> print,pgcomb(5,3)
       0       1       2
       0       1       3
       0       1       4
       0       2       3
       0       2       4
       0       3       4
       1       2       3
       1       2       4
       1       3       4
       2       3       4
```

It might not be the most efficient solution, but at least
I understand the (quite simple) underlying algorithm.

Be careful that no check is made for bad inputs, it's
up to the user to make sure that 1<=j<=n.

Ciao,
Paolo

```
FUNCTION pgcomb,n,j
;;number of combinations of j elements chosen from n
nelres=long(factorial(n)/(factorial(j)*factorial(n-j)))

res=intarr(j,nelres);array for the result
res[*,0]=indgen(j);initialize first combination

FOR i=1,nelres-1 DO BEGIN;go over all combinations
   res[*,i]=res[*,i-1];initialize with previous value

   FOR k=1,j DO BEGIN;scan numbers from right to left

    IF res[j-k,i] LT n-k THEN BEGIN;check if number can be increased

      res[j-k,i]=res[j-k,i-1]+1;do so

      ;if number has been increased, set all numbers to its right
      ;as low as possible
```

```
        IF k GT 1 THEN res[j-k+1:j-1,i]=indgen(k-1)+res[j-k,i]+1

        BREAK;we can skip to the next combination

       ENDIF

    ENDFOR

ENDFOR

RETURN,res

END
```

Olivia wrote:
> Dear All,
>
> I am trying to write a loop to perform a calculation on all possible
> sets of 5 coordinates from a group of n. The test case I am working on
> has a total number of coordinates of 8, so there will be 8c5=56 unique
> solutions. At the moment, I am thinking of using 5 for loops as an
> extension of a similar problem I worked on choosing 3 points. The 3
> point code ran like this:
>
```
> ;Select groups of 3 boundary points
> ;for p=1, n, 1 do begin
> ; for k=1, n, 1 do begin
> ;  for m=0, n-1, 1 do begin
> ;   ellipse_points=[[bx[m], by[m]],$
> ;                 [bx[m+k], by[m+k]],$
> ;                 [bx[m+k+p], by[m+k+p]]]
```
>
> On reflection, I was wondering if there might be a better way of doing
> this. This is probably only a fear as my code is already looking really
> complicated and I am worried about putting rubbish in and getting
> rubbish out whilst being completely unaware. If anyone has any ideas I
> would be really grateful to hear them. Thank you very much,
>
> Olivia
>