Subject: Re: Search routines
Posted by R. Bauer on Sat, 26 Sep 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Kenneth P. Bowman wrote:

> In article <3602D89B.15BF8C2D@ssec.wisc.edu>, Liam Gumley
> <Liam.Gumley@ssec.wisc.edu> wrote:
>
>> Kenneth P. Bowman wrote:
>>> IDL has a pretty good SORT routine, but no SEARCH routine that I have been
>>> able to find (that is, a procedure to find the index of the closest/first
>>> match in an ordered list). Once again, this can be done with loops, but
>>> such an implementation would almost certainly be much slower than a
>>> built-in function. Since searching and sorting are such basic operations,
>>> does anyone know why there is no SEARCH in IDL?
>>
>> How about the MIN function, e.g.
>>
>> array = findgen(100)
>> value = 37.2
>> result = min(abs(value - array), location)
>> help, location
>
> Again, I'm sure this is an order-N operation, as MIN has to check every
> element, just like WHERE. It has no knowledge that the list is ordered.
>
> Ken
>

Hi Ken,

look at this routine.

The name should mean find_middling_indices. It was written in the past where we
have not the possibilty to use long names.
The help part could be added if you are interested.
In addition to a normal search this routine is able to use a window which is
usefull for some algorithm we have written.

We are thinking that's this routine is very fast, but any ideas to get it faster
are welcome.

Reimar

--

R.Bauer

Institut fuer Stratosphaerische Chemie (ICG-1)
Forschungszentrum Juelich
email: R.Bauer@fz-juelich.de

```
;
; Copyright (c) 1996, 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:
;       fi_mi_in
;
; PURPOSE:
;   The result of this function is a two dimensional indexfield.
;   It is used to find the indices which overlaps in client time depending on
master time and time_window
;   The first index is the start and the second the end index of the overlapping
values from client_time.
;
; CATEGORY:
;   MATH
;
; CALLING SEQUENCE:
;   Result=fi_mi_in(client,master,master_time_window,[/help])
;
; INPUTS:
;       client:            The client time
;       master:              The master time
;   master_time_window: The time_window must have same size as master
;
; KEYWORD PARAMETERS:
;  help: gives a short description
;
; OUTPUTS:
;   This function returns a two dimensional indexfield
;   The first index is the start and the second the end index of the used time
window
;   -1 at an index means there were no results
;
```

```
; EXAMPLE:
;      client=[1,2,3,4,5]
;      master=[2,4]
;      time_window=[0,0]
;      Result=fi_mi_in(client,master,time_window)
;      help,result
;      RESULT          LONG      = Array[2, 2]
;      print,result
;           1           1
;           3           3
;
;
; MODIFICATION HISTORY:
;   Written by: R.Bauer (ICG-1),  1996-May-06
;   1998-Mar-02 much more efficiency by combining with suche.pro (F.Rohrer)
;
;-

FUNCTION fi_mi_in,client,master,time_window,help=help

 if keyword_set(help) then begin
      help,call=call
      help_of_interest=within_brackets(call[0],brackets=['<','('])
      message,help_calling_sequence(help_of_interest),/cont
      return,-1
   endif


   laenge_master=(size(master))(1)
   lstx=(size(client))(1)-1


   remember=0
   index=MAKE_ARRAY(2,laenge_master,type=3,value=-1)



   FOR i=0L, laenge_master-1 DO BEGIN
      k = LONG(remember)
      WHILE client(k) LT master(i) -time_window(i)  AND k LT lstx DO k=k+1
      IF client(k) GE master(i) -time_window(i)  and client(k) LE
master(i)+time_window(i) THEN BEGIN
         index(0,i)=LONG(k)
         remember=(index(0,i))(0)
      ENDIF

      if index(0,i) ne -1 then begin
         mx = remember
```

```
        WHILE  client(mx) LE master(i)+time_window(i)  and mx LT lstx DO mx=mx+1

        if client(mx) LE master(i)+time_window(i) then index(1,i)=LONG(mx) else
index(1,i)=LONG(mx)-1
      endif

   ENDFOR

   RETURN,index
END
```