Subject: Need look-up-table routine
Posted by cri on Tue, 26 Nov 1996 08:00:00 GMT
View Forum Message <> Reply to Message

Hello,

I was about to try to write a routine which will, upon input of a value or array of values, give a corresponding output value or array of values according to a look-up table (also provided as input). I'm not sure yet whether I need to use interpolation methods, or just nearest-value, but in any case, I thought I should ask around here first, to see if such a routine already exists, either in IDL or somewhere else. Can anyone help?

Thanks,

Adam Ross
CRI, Inc.
cri@world.std.com

Subject: Re: Need look-up-table routine
Posted by dutch on Wed, 27 Nov 1996 08:00:00 GMT
View Forum Message <> Reply to Message

A follow-up to my previous post and that of Tom McGlynn

I just realised that I have largely phased out my usage of Table1 here, and replaced it with the equivalent routine provided by IDL:
INTERPOL.PRO (look under MATH routines in the IDL online help)
which handles irregular grids (i.e. x-values)
not to be confused with INTERPOLATE.PRO (IDL help - PROG routines)

Table2 may still be of some use for 2D table lookup on an irregular grid?

-- Michael

###
Dr. Michael Dutch email: dutch@eltcv1.epfl.ch
Centre de Recherches en Physique des Plasmas
Ecole Polytechnique Federale de Lausanne
21 Ave des Bains Aussie.Abroad
CH-1007 Lausanne, Switzerland ,--_|\
#--------------------------------------- / \ #
Stand clear of the doors, please! _.-X._/
Mind...The Gap, Mind...The Gap v
###

Subject: Re: Need look-up-table routine
Posted by dutch on Wed, 27 Nov 1996 08:00:00 GMT
View Forum Message <> Reply to Message

Hope these help:

Here are 1D and 2D table lookup routines which I wrote some time ago.
The procedures are heavily based on the MATLAB functions of the same
name.

Table1 can run into memory problems on our VMS Alphas here, for very large
tables, so two different techniques are used, based on table size.
Table2 calls table1.

These seem to work well for us, but let me know if you find bugs or improvements

--Enjoy Michael.


```
################################################################
# Dr. Michael Dutch          email: dutch@eltcv1.epfl.ch #
# Centre de Recherches en Physique des Plasmas           #
# Ecole Polytechnique Federale de Lausanne              #
# 21 Ave des Bains                      Aussie.Abroad #
# CH-1007 Lausanne, Switzerland              ,--_|\     #
#---------------------------------------    /     \    #
# Stand clear of the doors, please!         \_.-X._/    #
# Mind...The Gap,   Mind...The Gap               v      #
################################################################
```


----------8< start of table1.pro >8----------

```
FUNCTION table1,tab,x0
;+
; *** streamlined and speeded up 31/3/93 ***
;
; PURPOSE: Table look-up.
;
; USAGE:   Y = TABLE1(TAB,X0) returns a table of linearly interpolated
;    rows from table TAB, looking up X0 in the first column of TAB.
;    See also TABLE2.PRO
;         NOTE:  TAB's 1st column is checked for monotonicity.
;         It is an error to request a value outside the range of the first
;         column of TAB for X0.
;
; NOTES:   Modelled on the MATLAB routine of the same name!
;
; AUTHOR:  M.J.DUTCH 3/MAR/1992
; MODIFS:  M.J.DUTCH 1/APR/1992   Various dramatic  speed-ups.
```

```
;
;-

if (n_params() ne 2) then message,'two parameters are required.'

tabsize=size(tab)
x0size=size(x0)
xsize=x0size
if (tabsize(0) ne 2) then message,'TAB must be an array'
if (x0size(0) ge 2) then begin
   x0=reform(x0)
   xsize=size(x0)
   if (xsize(0) ge 2) then message,'X0 must be scalar or vector'
endif

m=tabsize(1) & n=tabsize(2)
if (xsize(0) eq 0) then nx=1         ; scalar
if (xsize(0) eq 1) then nx=xsize(1)  ; vector

;-- check whether X0 is in range --
   x0min=min(x0,max=x0max)
   tabmin=min(tab(0,*),max=tabmax)
   if ((x0min lt tabmin) or (x0max gt tabmax)) then $
      message,'x0 outside range of table'

;-- difference table --
dx=tab(*,1:n-1)-tab(*,0:n-2)
dx=[[dx],[dx(*,n-2)]] & sig=sign(dx(0,0))
ind=where(sign(dx(0,*))-sig)
if ind(0) ne -1 then $
   message,'First column of table must be monotonic.'

;-- find the indices --
;create a vector of ones (as fast as possible)
;cannot use replicate as it consumes memory (bug!?)
ones_nx=(x0 ne -999.99)   ;byte array!
ones_n =reform(tab(0,*) ne -999.99) ;byte array!
if (sig gt 0) then begin
   ind=(ones_nx#tab(0,*) le x0#ones_n) # ones_n
endif else begin
   ind=(ones_nx#tab(0,*) ge x0#ones_n) # ones_n
endelse
ind=ind-1

;-- interpolate --
k=indgen(nx)
y=fltarr(m-1,nx)
 y(*,k)=tab(1:m-1,ind)+dx(1:m-1,ind)*(x0(k)-tab(0,ind))/dx(0, ind)
```

```
return,reform(y)
end


----------8<    end of table1.pro     8>------------------

----------8<    start of table2.pro   8>------------------

FUNCTION table2,tab,x0,y0
;+
;
; PURPOSE: Two-dimensional table look-up.
;
; USAGE:   Z = TABLE2(TAB,X0,Y0) returns a linearly interpolated
;    intersection from table TAB, looking up X0 in the first
;    column and Y in the first row of TAB.  See also TABLE1.PRO
;
; NOTES:   Closely modelled on the MATLAB routine of the same name!
;
; AUTHOR:  M. J. DUTCH   3/MAR/1992
;
;-
if (n_params() ne 3) then message,'three input arguments are required.'

varsize = size(tab)
if (varsize(0) ne 2) then message,'TAB must be a 2D array'

m=varsize(1) & n=varsize(2)

a = table1(tab(*,1:n-1),x0)
tab2 = transpose([[tab(1:m-1,0)],[a]])
z = table1(tab2,y0)
z=transpose(z)
return,z
end

----------8<    end of table2.pro   8>--------------------
```

Subject: Re: Need look-up-table routine
Posted by Thomas A. McGlynn on Wed, 27 Nov 1996 08:00:00 GMT
View Forum Message <> Reply to Message

Adam Ross wrote:
>
> Hello,
>
> I was about to try to write a routine which will, upon input of a

> value or array of values, give a corresponding output value or array
> of values according to a look-up table (also provided as input).
> I'm not sure yet whether I need to use interpolation methods, or
> just nearest-value, but in any case, I thought I should ask around
> here first, to see if such a routine already exists, either in IDL or
> somewhere else. Can anyone help?
>
> Thanks,
>
> Adam Ross
> CRI, Inc.
> cri@world.std.com

If I understand you want a function such that

```
  results = lookup(grid_values, desired_indices)
  where the grid_values = [13., 23., 19., 7.] and
  desired_indices = [0.3, 2.1, 0.4, 1.1, 1.1, 3., 2.2]
```

giving (for nearest neighbor)

```
  results = [13., 19., 13., 23., 23., 7., 19.]
```

For the nearest neighbor case you don't need a function:
you can just do

```
  results = grid_values(desired_indices+.5)
```

assuming you don't want to do anything special for out of
range values.

For linear interpolation just use the interpolate function

```
   results = interpolate(grid_values, desired_indices)
```

My apologies if I've misunderstood what you're asking for.

Yours,
Tom McGlynn
tam@silk.gsfc.nasa.gov
