
Subject: Re: Another "IDL way to do this" question
Posted by [Fabzou](#) on Wed, 09 Nov 2011 09:00:36 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hi Brian,

On 11/08/2011 08:20 PM, Brian Wolven wrote:
> Are your z values the same for each x-y-t slice?
>

No, that's the same problem than with Jeremy Bailin. I am sorry I was not clear enough in my first post...

Thanks a lot,

Fab

```
>
> ;=====
=====
> ;+
> ; ROUTINE: findex
> ;
> ; PURPOSE: Compute "floating point index" into a table using binary
> ;           search. The resulting output may be used with INTERPOLATE.
> ;
> ; USAGE: result = findex(u,v)
> ;
> ; INPUT:
> ;   u    a monitically increasing or decreasing 1-D grid
> ;   v    a scalar, or array of values
> ;
> ; OUTPUT:
> ;   result Floating point index. Integer part of RESULT(i) gives
> ;           the index into to U such that V(i) is between
> ;           U(RESULT(i)) and U(RESULT(i)+1). The fractional part
> ;           is the weighting factor
> ;
> ;           V(i)-U(RESULT(i))
> ;           -----
> ;           U(RESULT(i)+1)-U(RESULT(i))
> ;
> ; DISCUSSION:
> ;   This routine is used to expedite one dimensional
> ;   interpolation on irregular 1-d grids. Using this routine
> ;   with INTERPOLATE is much faster then IDL's INTERPOL
```

```

> ;      procedure because it uses a binary instead of linear
> ;      search algorithm. The speedup is even more dramatic when
> ;      the same independent variable (V) and grid (U) are used
> ;      for several dependent variable interpolations.
> ;
> ;
> ; EXAMPLE:
> ;
> ;; In this example I found the FINDEX + INTERPOLATE combination
> ;; to be about 60 times faster then INTERPOL.
> ;
> ; u=randomu(iseed,200000)& u=u(sort(u))
> ; v=randomu(iseed,10)& v=v(sort(v))
> ; y=randomu(iseed,200000)& y=y(sort(y))
> ;
> ; t=systime(1)& y1=interpolate(y,findex(u,v))& print,systime(1)-t
> ; t=systime(1)& y2=interpol(y,u,v)& print,systime(1)-t
> ; print,f='(3(a,10f7.4/))','findex: ',y1,'interpol: ',y2,'diff: ',y1-y2
> ;
> ; AUTHOR: Paul Ricchiazzi          21 Feb 97
> ;         Institute for Computational Earth System Science
> ;         University of California, Santa Barbara
> ;         paul@icess.ucsb.edu
> ;
> ; REVISIONS:
> ;
> ;-
> ;
> ;=====
> ;=====
> function findex,u,v
> ;=====
> ;=====
> ;=====
> ;=====
> nu = n_elements(u)
> nv = n_elements(v)
> us = u-shift(u,+1)
> us = us(1:*)
> umx = max(us,min=umn)
> if (umx gt 0) and (umn lt 0) then message,'u must be monotonic'
> if (umx gt 0) then inc=1 else inc=0
> ;=====
> ;=====
> ; maxcomp = maximum number of binary search iterations
> ;=====
> ;=====
> maxcomp = fix(alog(float(nu))/alog(2.))+.5)

```

```

> jlim    = lonarr(2,nv)
> jlim(0,*) = 0      ; array of lower limits
> jlim(1,*) = nu-1    ; array of upper limits
> iter    = 0
> repeat begin
>   jj = (jlim[0,*]+jlim[1,*])/2
>   ii = where(v ge u[jj],n)& if (n gt 0) then jlim(1-inc,ii) = jj[ii]
>   ii = where(v lt u[jj],n)& if (n gt 0) then jlim(inc,ii)  = jj[ii]
>   jdif = max(jlim[1,*]-jlim[0,*])
>   if iter gt maxcomp then begin
>     print,maxcomp,iter, jdif
>     message,'binary search failed'
>   endif
>   iter = iter+1
> endrep until jdif eq 1
> w = v-v
> w[*] = (v-u[jlim[0,*]]/(u[jlim[0,*]+1]-u[jlim[0,*]])) + jlim[0,*]
> return,w
> end

```
