Subject: Re: Searching for fast linear interpolation routine
Posted by David Crain on Mon, 07 Apr 1997 07:00:00 GMT
View Forum Message <> Reply to Message

Christian Marquardt wrote:
>
> Hello,
>
> Liam Gumley wrote:
>>
>> Roger J. Dejus wrote:
>>> Did someone write a fast linear interpolation routine for irregular one
>>> dimensional arrays (monotonically ascending or descending abscissas)
>>> similar in functionality to the INTERPOL.PRO routine from RSI?
>>
>> If you can find a way to use the built-in routine INTERPOLATE, you will
>> see a dramatic speed increase - it is very fast on 1D, 2D or 3D arrays.
>>
>> Cheers,
>> Liam.
>
> Recently, Paul Richiazzi posted a solution to this problem to the
> newsgroup; I'll attach his posting...
>
> Hope this helps,
>
> Chris Marquardt.
>
> ---
> Path: fu-berlin.de!news.apfel.de!news.maxwell.syr.edu!news.bc.net!
info.ucla.edu!nnrp.info.ucla.edu!ucsbuxb.ucsb.edu!ucsbuxb.uc sb.edu!paul
> From: paul@orion.crseo.ucsb.edu (Paul Ricchiazzi)
> Newsgroups: comp.lang.idl-pvwave
> Subject: A faster way to INTERPOL
> Date: 21 Feb 1997 21:30:44 GMT
> Organization: University of California, Santa Barbara
> Lines: 130
> Distribution: global
> Message-ID: <PAUL.97Feb21133044@orion.crseo.ucsb.edu>
> NNTP-Posting-Host: orion.crseo.ucsb.edu
>
> I find that using INTERPOL to do 1-d interpolations on large irregular
> grids can be extremely time consuming. The problem with INTERPOL is
> that it uses a linear search to find where a given field point fits
> into the irregular grid. Below you'll find my solution to the
> problem. The procedure FINDEX uses a binary search to obtain a
> "floating point index" which can be used with INTERPOLATE. I have
> found that the FINDEX + INTERPOLATE method can be up to 70 times

```
>  faster than using INTERPOL.  I am donating this procedure to the IDL
>  community in hopes of saving untold millions of machine cycles that
>  would otherwise have been wasted in futile linear searches.  But
>  seriously, give it a try and let me know if it breaks.
>
>  Regards,
>
>  Paul Ricchiazzi
>
>  ---------8<---------8<---------8<---------8<---------8<---------8 <---------8<
>
>  function findex,u,v
>  ;+
>  ; ROUTINE:  findex
>  ;
>  ; PURPOSE:  Compute "floating point index" into a table using binary
>  ;          search.  The resulting output may be used with INTERPOLATE.
>  ;
>  ; USEAGE:   result = findex(u,v)
>  ;
>  ; INPUT:
>  ;   u       a monitically increasing or decreasing 1-D grid
>  ;   v       a scalor, 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:
>  ;
>  ;-
>  ;
>  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=fix(alog(float(nu))/alog(2.)+.5)
>
>  ; maxcomp = maximum number of binary search iteratios
>
>  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
>
> --
>
>
> _____ _____
> Paul Ricchiazzi
> Institute for Computational Earth System Science (ICESS)
> University of California, Santa Barbara
>
> email: paul@icess.ucsb.edu
> _____ _____
>
> --
> ---------- ----------
> Christian Marquardt
>
> Meteorologisches Institut der | tel.: (+49) 30-838-71170
> Freien Universitaet Berlin | fax.: (+49) 30-838-71167
> Carl-Heinrich-Becker-Weg 6-10 | email: marq@strat01.met.fu-berlin.de
> D-12165 Berlin |
> ---------- ----------

Yes, more than a factor of 2 quicker using the same test6.pro described above!

David Crain