Subject: Re: Maximum value array resampling
Posted by Richard French on Sun, 14 Aug 2005 01:05:54 GMT
View Forum Message <> Reply to Message

On 8/8/05 2:20 PM, in article pan.2005.08.08.18.20.38.917907@as.arizona.edu,
"JD Smith" <jdsmith@as.arizona.edu> wrote:

>
> By the way, since for processing many images of the same size you can
> pre-compute the indices, you might consider a small change to the
> modified loop method:
>
>
> ;; Pre-compute the indices:
> d=size(x,/DIMENSIONS) & nx=d[0] & ny=d[1]
> nx2=nx/2 & ny2=ny/2
>
> inds1=rebin(lindgen(nx2)*2L,nx2,ny2,/SAMPLE)+ $
> rebin(transpose(lindgen(ny2)*2L*nx),nx2,ny2,/SAMPLE)
> inds2=inds1+1L
> inds3=inds1+nx
> inds4=inds1+nx+1L
>
> ;; Form the sub-sampled image (for each image)
> xmax=x[inds1]>x[inds2]>x[inds3]>x[inds4]
>
>
> This brings the total processing time per 5000x4000 image to under
> 0.5s on my not-so-fast Linux box (and doesn't have any loops ;).
>
> JD
>

JD also wrote:

I find just the opposite: your loop+REBIN method is much slower using
5000x4000 long integer arrays:

IDL> .run /home/jdsmith/idl/test/max_local.pro
% Compiled module: $MAIN$.
no loop [5000,4000]: 3.8933
French index loop [5000,4000]: 33.1060
modified index loop [5000,4000]: 1.0784

I.e. yours is about 8-10x slower on this size image! All of this of
course depends on memory (1GB here). I suspect your multiple

REBIN'ing of those large images is to blame. That said, I tried with
a much smaller image, but the results were similar...

(end of quote)

John - Just to clarify things, your 8/8/05 routine above is essentially
identical to the revised routine I posted on 8/5/05 (see below), and I was
referring to this new routine when I said it was faster than what you had
posted. I had also posted a lame rebinning approach to the problem
previously, and I think that is the one you used when you found such poor
performance.
Dick

My previous posting is below:

```
nx=4000L
ny=5000L

; put in random values (this is the slow step!)

X=rebin(fix(1000*(randomu(seed,nx*ny))),nx,ny)
nx2=nx/2L
ny2=ny/2L

; get indices of upper left element of each 2x2 cell

 l=rebin(nx*2#lindgen(ny2),nx2,ny2)+rebin(lindgen(nx2)#2,nx2, ny2)


; compare with  indices of ul,ur, ll, lr of each 2x2 cell

print,'Start....'
T10=systime(1)
xfinal=x[l]>x[l+1]>x[l+nx]>x[l+nx+1]
print,'Time=',systime(1)-t10
```