Subject: Re: Cumulative max() in *arbitrary* dimension?
Posted by JDS on Fri, 09 Mar 2012 15:50:24 GMT
View Forum Message <> Reply to Message

On Thursday, March 8, 2012 4:15:19 PM UTC-5, Lajos Foldy wrote:
> On Mar 8, 7:33 pm, JDS <jdtsmith.nos...@yahoo.com> wrote:
>
>>
>> IDL> a=byte(randomu(sd,300,400,3000)*256)
>> IDL> t=systime(1) & b=max(a,DIMENSION=3) & print,systime(1)-t
>> IDL> t=systime(1) & b2=a[*,*,0] & for i=1,3000-1 do b2>=a[*,*,i] & print,systime(1)-t
>> IDL> print,array_equal(b,b2)
>>
>> I can only presume the built-in MAX has some design limitations for final dimension looping.
I presume this all works the same way for MIN, BTW.
>
> b=max(a,DIMENSION=3) is equivalent to
>
> b=a[*,*,0]
> for j1=0,299 do begin
> for j2=0,399 do begin
> b[j1,j2]=a[j1,j2,0]
> for j3=1,2999 do b[j1,j2]>=a[j1,j2,j3]
> endfor
> endfor
>
> Your solution is equivalent to
>
> b2=a[*,*,0]
> for j3=1,2999 do begin
> for j2=0,399 do begin
> for j1=0,299 do b2[j1,j2]>=a[j1,j2,j3]
> endfor
> endfor
>
> The big difference comes from the memory access pattern.

Seems sensible. I guess I'm surprised that MAX doesn't order the nested loops sensibly, given that it must of course have a large different loop sets for each of the possible total dimensions/target dimension combinations.

JD