Subject: Re: Cleaver 2d reverse indicies?
Posted by Chris[6] on Mon, 03 Nov 2008 21:15:08 GMT
View Forum Message <> Reply to Message

```
>  bins_mean = fltarr(size(nd, /dim))
>  nx = (size(nd, /dim))[0]
>  ny = (size(nd, /dim))[1]
>  FOR i = 0UL, nx-1 DO BEGIN
>    FOR j = 0UL, ny-1 DO BEGIN
>      ind_ri = [i+nx*j]
>      IF ri[ind_ri] EQ ri[ind_ri+1] THEN CONTINUE ; nothing to do in
>  this iteration
>      ri_sel = ri[ri[ind_ri]:ri[ind_ri+1]-1]
>      bins_mean[i, j] = mean(pxxm(ri_sel), /nan)
>    ENDFOR
>  ENDFOR
```

Well, it seems like you can definitely eliminated the nested loop:

```
FOR i = 0UL, nx * ny - 1, DO BEGIN
   if ri[i] eq ri[i+1] then continue ; nothing to do here
   ri_sel = ri[ri[i] : ri[i+1] - 1]
   bins_mean[( i mod nx), (i / nx)] = mean(pxxm(ri_sel),/nan)
endfor
```

I've always wondered if there is a way to eliminated the first loop
when using reverse indices, but haven't thought of / seen a way to
yet.

chris

Subject: Re: Cleaver 2d reverse indicies?
Posted by Brian Larsen on Mon, 03 Nov 2008 21:26:17 GMT
View Forum Message <> Reply to Message

Chris,

This works like a charm and is noticeably faster (~1.5 times). Gotta
love this usenet group.

Thanks much.

Say you were to move to 3d (or 4d) does this extend in the same manner
with just one total loop or can you just kill one? Brain is a bit
slow on this today (it is Monday after all).

Brian

Brian Larsen
Boston University
Center for Space Physics
http://people.bu.edu/balarsen/Home/IDL

Subject: Re: Cleaver 2d reverse indicies?
Posted by Chris[6] on Tue, 04 Nov 2008 00:40:42 GMT
View Forum Message <> Reply to Message

On Nov 3, 11:26 am, Brian Larsen <balar...@gmail.com> wrote:
> Chris,
>
> This works like a charm and is noticeably faster (~1.5 times). Gotta
> love this usenet group.
>
> Thanks much.
>
> Say you were to move to 3d (or 4d) does this extend in the same manner
> with just one total loop or can you just kill one? Brain is a bit
> slow on this today (it is Monday after all).
>
> Brian
>
> ---
> Brian Larsen
> Boston University
> Center for Space Physicshttp://people.bu.edu/balarsen/Home/IDL

Yes, I think one loop through all of the indices is always possible.

In fact, come to think of it, you don't even need to do the
transformation. IDL just converts array[x,y,z] back into array[i]
anyways - this could save some extra time.

chris

Subject: Re: Cleaver 2d reverse indicies?
Posted by Chris[6] on Tue, 04 Nov 2008 09:56:03 GMT
View Forum Message <> Reply to Message

>
> In fact, come to think of it, you don't even need to do the
> transformation. IDL just converts array[x,y,z] back into array[i]

> anyways - this could save some extra time.

IN FACT, maybe we don't need loops at all...

```
t0 = systime(/seconds)
mean3 = fltarr(size(nd,/dim))

ind = ulindgen(nx * ny - 1) + 1 ;- don't include 0- do it manually
bad = where(ri[ind] eq ri[ind+1], ct) + 1 ;-add 1 because ind starts
at 1
newRI = ri[0:nx * ny] - ri[0]

runningSum = total(pxxm[ri[ri[0]:ri[nx * ny] - 1]], /cumulative)

mean3[ind] = (runningSum[newRi[ind+1] - 1] - runningSum[newRi[ind] -
1]) / (newRi[ind+1] - newRi[ind])
if ct ne 0 then mean3[bad] = 0 ;- fix empty bins

;-manually fill in first element
if newRI[1] ne 0 then $
  mean3[0] = runningSum[newRI[1] - 1] / newRI[1]

print, 'time: ',systime(/seconds) - t0
```

All of the adding and subtracting of 1s is super ugly, but it runs about 30x faster for me. Also, the /CUMULATIVE keyword for total seems to be unstable - the errors between this method and the earlier method grow with the index number. That seems bizarre, but the errors were minor (.01%) for the input I used.

chris

Subject: Re: Cleaver 2d reverse indicies?
Posted by Jeremy Bailin on Tue, 04 Nov 2008 15:48:44 GMT
View Forum Message <> Reply to Message

On Nov 4, 4:56 am, Chris <beaum...@ifa.hawaii.edu> wrote:
>> In fact, come to think of it, you don't even need to do the
>> transformation. IDL just converts array[x,y,z] back into array[i]
>> anyways - this could save some extra time.
>
> IN FACT, maybe we don't need loops at all...
>
> t0 = systime(/seconds)
> mean3 = fltarr(size(nd,/dim))

>
> ind = ulindgen(nx * ny - 1) + 1 ;- don't include 0- do it manually
> bad = where(ri[ind] eq ri[ind+1], ct) + 1 ;-add 1 because ind starts
> at 1
> newRI = ri[0:nx * ny] - ri[0]
>
> runningSum = total(pxxm[ri[ri[0]:ri[nx * ny] - 1]], /cumulative)
>
> mean3[ind] = (runningSum[newRi[ind+1] - 1] - runningSum[newRi[ind] -
> 1]) / (newRi[ind+1] - newRi[ind])
> if ct ne 0 then mean3[bad] = 0 ;- fix empty bins
>
> ;-manually fill in first element
> if newRI[1] ne 0 then $
> mean3[0] = runningSum[newRI[1] - 1] / newRI[1]
>
> print, 'time: ',systime(/seconds) - t0
>
> All of the adding and subtracting of 1s is super ugly, but it runs
> about 30x faster for me. Also, the /CUMULATIVE keyword for total seems
> to be unstable - the errors between this method and the earlier method
> grow with the index number. That seems bizarre, but the errors were
> minor (.01%) for the input I used.
>
> chris

Try using /INTEGER with it.

-Jeremy.
