Subject: Re: Another "How to efficiently do this in IDL" question
Posted by Jeremy Bailin on Sat, 22 Oct 2011 04:02:28 GMT
View Forum Message <> Reply to Message

On 10/21/11 6:15 PM, Robin Wilson wrote:
> Hi all,
>
> I've got a 2D integer array. For every cell in the array I want to do
> the following:
>
> 1. Get the points that lie on a horizontal profile 7 cells long, with
> the current cell as the central cell.
>
> 2. Find out if there are two (or more) points on both sides of the
> central point that have a lower value than the central point. If so,
> then mark this point in a separate array.
>
> I can think of loads of ways to do this using very inefficient for loops
> and lots of horrible code, but are there any particularly nice ways to
> do this in IDL.
>
> It looks to me like I'd have to loop over every cell in the array - it's
> not something I could do all at once - but I'm willing to be corrected.
>
> The other main question is that if I've got an array of points like the
> following:
>
> 1 3 3 5 6 2 1
> *
>
> What is an efficient way to check that there are at least two points on
> each side of the central point (marked with a star) that have a lower
> value than it. My original thought was to loop through the cells, but I
> suspect some fancy histogram command could do something to help with
> this...
>
> Any advice would be most appreciated.
>
> Cheers,
>
> Robin
> ------------------
> Robin Wilson
> A PhD student studying complexity in remote sensing
> www.rtwilson.com/academic

How about this, if your data is in the array "data". It uses one for
loop, but it only runs from 1 to 3:

```
; sizes
horizwidth = 7
halfwidth = horizwidth/2
datasize = size(data,/dimen)

; arrays to hold how many elements to the left and right are less
left_ngreater = intarr(datasize)
right_ngreater = intarr(datasize)

; loop through each of the three side elements and add up the
; ones that are smaller than the central value
for i=1,halfwidth do begin
   left_ngreater += data gt shift(data,i,0)
   right_ngreater += data gt shift(data,-i,0)
endfor

; which ones have at least two in both left and right arrays?
; only use elements that are far enough away from the edge
desired = where(left_ngreater[halfwidth:datasize[0]-halfwidth-1,*] ge 2 $
    and right_ngreater[halfwidth:datasize[0]-halfwidth-1,*] ge 2, ndesired)

; get out the 2D indices of those
if ndesired gt 0 then begin
   desired_ai = array_indices([datasize[0]-(2*halfwidth),datasize[1]], $
     desired, /dimensions)
   desired_x = reform(desired_ai[0,*]+halfwidth)
   desired_y = reform(desired_ai[1,*])
endif
```

For example, with the following array:

```
IDL> print, data
       4       9       7       5       9       4       6       0
1       8
       0       1       2       3       4       3       2       1
0       1
       3       4       8       5       1       9       3       7
4       6
```

I get:

```
IDL> print, desired_x
           4           6           4           3           5
IDL> print, desired_y
           0           0           1           2           2
```

-Jeremy.

Subject: Re: Another "How to efficiently do this in IDL" question
Posted by Jeremy Bailin on Sat, 22 Oct 2011 04:04:45 GMT
View Forum Message <> Reply to Message

On 10/21/11 6:15 PM, Robin Wilson wrote:
> Hi all,
>
> I've got a 2D integer array. For every cell in the array I want to do
> the following:
>
> 1. Get the points that lie on a horizontal profile 7 cells long, with
> the current cell as the central cell.
>
> 2. Find out if there are two (or more) points on both sides of the
> central point that have a lower value than the central point. If so,
> then mark this point in a separate array.
>
> I can think of loads of ways to do this using very inefficient for loops
> and lots of horrible code, but are there any particularly nice ways to
> do this in IDL.
>
> It looks to me like I'd have to loop over every cell in the array - it's
> not something I could do all at once - but I'm willing to be corrected.
>
> The other main question is that if I've got an array of points like the
> following:
>
> 1 3 3 5 6 2 1
> *
>
> What is an efficient way to check that there are at least two points on
> each side of the central point (marked with a star) that have a lower
> value than it. My original thought was to loop through the cells, but I
> suspect some fancy histogram command could do something to help with
> this...
>
> Any advice would be most appreciated.
>
> Cheers,
>
> Robin
> ------------------
> Robin Wilson
> A PhD student studying complexity in remote sensing

> www.rtwilson.com/academic

How about this, if your data is in the array "data". It uses one for
loop, but it only runs from 1 to 3:

```
; sizes
horizwidth = 7
halfwidth = horizwidth/2
datasize = size(data,/dimen)

; arrays to hold how many elements to the left and right are less
left_ngreater = intarr(datasize)
right_ngreater = intarr(datasize)

; loop through each of the three side elements and add up the
; ones that are smaller than the central value
for i=1,halfwidth do begin
   left_ngreater += data gt shift(data,i,0)
   right_ngreater += data gt shift(data,-i,0)
endfor

; which ones have at least two in both left and right arrays?
; only use elements that are far enough away from the edge
desired = where(left_ngreater[halfwidth:datasize[0]-halfwidth-1,*] ge 2 $
    and right_ngreater[halfwidth:datasize[0]-halfwidth-1,*] ge 2, ndesired)

; get out the 2D indices of those
if ndesired gt 0 then begin
   desired_ai = array_indices([datasize[0]-(2*halfwidth),datasize[1]], $
     desired, /dimensions)
   desired_x = reform(desired_ai[0,*]+halfwidth)
   desired_y = reform(desired_ai[1,*])
endif
```

For example, with the following array:

```
IDL> print, data
       4       9       7       5       9       4       6       0
1       8
       0       1       2       3       4       3       2       1
0       1
       3       4       8       5       1       9       3       7
4       6
```

I get:

```
IDL> print, desired_x
```

```
          4          6          4          3          5
IDL> print, desired_y
          0          0          1          2          2
```

-Jeremy.