Subject: Re: The IDL way: Find last non-zero value
Posted by Chris[6] on Mon, 24 Aug 2009 00:32:18 GMT
View Forum Message <> Reply to Message

On Aug 24, 2:07 am, Eric Hudson <ehud...@mit.edu> wrote:
> Hi,
>
> I have a 2D array that looks something like:
> x x 0 x x 0 0 x 0 0 0 0 0
> x 0 x x x 0 x x x x 0 0 0
> x 0 0 0 0 0 0 0 0 0 0 0 0
> x x x x x 0 0 0 0 0 0 0 0
>
> where x is some non-zero (positive definite) value. You'll notice
> that each row ends with a string of zeros.
> What I'd like to know is the 'IDL way' of returning a vector of the
> location (column) of the last non-zero elements in each row. So in
> this case, [7,9,0,4]
>
> It's straight forward to program with loops, but I figure there must
> be a clever way. I thought that maybe reversing it and doing a
> cumulative total might be a start, but then I can't convince myself
> that that is really going to be faster than doing a loop.
>
> For a sense of scale, the real array is something like 200 x 160000
>
> Thanks,
> Eric

hmmmm.....

```
sz = size(array)
ncol = sz[1]
nrow = sz[2]
nonzero = where(array ne 0)
ind = array_indices(array, nonzero)
sorted = sort(ind[0, *])
result = fltarr(nrow) - 1
result[ind[1, sorted]] = ind[0, sorted]
```

kind of hacky, but heres the idea:

find all of the nonzero elements, and then use array_indices to give
their row/column numbers. Then, find sorting of the array_indices
array that puts the column indices in ascending order. Next, make a
result array with the correct size, and copy, _in this sorted order_,
the columns from the array_indices array into the representative rows
of the result vector. This way, low column indices will get

overwritten by higher column values during the copy.

My test on a small array:

```
IDL> print, array
      1       2       0       0       0
      0       0       0       0       1
      1       1       3       1       0
      0       0       0       0       0
IDL> print, result
     1.00000      4.00000      3.00000     -1.00000
```

Of course, I'm not convinced that this is easier to read or faster
than using a loop....

Chris

Subject: Re: The IDL way: Find last non-zero value
Posted by penteado on Mon, 24 Aug 2009 02:10:54 GMT
View Forum Message <> Reply to Message

On Aug 23, 9:32 pm, Chris <beaum...@ifa.hawaii.edu> wrote:
> sz = size(array)
> ncol = sz[1]
> nrow = sz[2]
> nonzero = where(array ne 0)
> ind = array_indices(array, nonzero)
> sorted = sort(ind[0, *])
> result = fltarr(nrow) - 1
> result[ind[1, sorted]] = ind[0, sorted]

That gave me this idea:

```
function lastnz2d,array
;Returns a vector with the index of the last nonzero element
;of each line of array (-1 if that line only has zeroes)
sz=size(array)
;Find all nonzero elements in array
nonzero=where(array ne 0,count)
;Get out if all elements of array are zero
if (count eq 0) then return,replicate(-1,sz[2])
;Find the row/column number of each nonzero element
ind=array_indices(array,nonzero)
;Initialize the result vector
res=lonarr(sz[2])-1L
;The repeated numbers in the second column of ind are
;the elements in the same line, so find the last occurrence
```

```
;of each sequence of repeated numbers
uni=uniq(ind[1,*])
;The result of uniq is the row number in ind of the last nonzero
element
;of each row in array
res[ind[1,uni]]=ind[0,uni]
return,res
end
```

It is the same up to the use array_indices to get row and column numbers into ind. For each row of array with nonzero elements, the corresponding rows of ind will have the same number in the second column (since it is the row number in array), with the increasing column numbers in the first column. So I use uniq on the second column of ind to retrieve every row in ind that corresponds to a last nonzero element in a row of array.

Subject: Re: The IDL way: Find last non-zero value
Posted by cgguido on Mon, 24 Aug 2009 03:39:46 GMT
View Forum Message <> Reply to Message

Another option - given that your array contains only positive numbers,
I thought total(array, 1, /cumulative) could be handy:

```
array=[ [1,2, 0, 1, 0, 0],$
[0, 0, 3,4, 1, 0],$
[1, 5,6, 0, 0, 0],$
[0,0,0,0,0,0], $
[1, 0,0,0,0,0] ]


tc=total(array, 1, /cumulative)
blah=max(tc, dim=1, w)
result=(array_indices(tc,w))[0,*]

t=total(array, 1)
result[where(t eq 0)]=-1
```

Gianguido

Subject: Re: The IDL way: Find last non-zero value
Posted by Eric Hudson on Mon, 24 Aug 2009 20:42:28 GMT
View Forum Message <> Reply to Message

Thanks to all of you for the suggestions. I timed them all on several of the larger matrices and found that the last suggestion (using cumulative totals) was about twice as fast as the previous ones, and about 5 times faster than the brute force loop. It did depend on how many zeros ended each line on average. When the number was small the loop could win, but for my typical matrices the cumulative total is the clear winner.

Thanks again,
Eric

Subject: Re: The IDL way: Find last non-zero value
Posted by Jeremy Bailin on Tue, 25 Aug 2009 00:12:23 GMT
View Forum Message <> Reply to Message

On Aug 24, 4:42 pm, Eric Hudson <ehud...@mit.edu> wrote:
> Thanks to all of you for the suggestions. I timed them all on several
> of the larger matrices and found that the last suggestion (using
> cumulative totals) was about twice as fast as the previous ones, and
> about 5 times faster than the brute force loop. It did depend on how
> many zeros ended each line on average. When the number was small the
> loop could win, but for my typical matrices the cumulative total is
> the clear winner.
>
> Thanks again,
> Eric

I'm shocked that no one's suggested histogram! Here's my take, which I'd wager does at least as well as the cumulative totals:

```
sz=size(array,/dimen)
ncol=sz[0]
nrow=sz[1]
evenrows = 2*lindgen(nrow)
h = histogram( (array ne 0) + rebin(reform(evenrows,
1,nrow),ncol,nrow), min=0, max=2*nrow-1, $
  reverse_indices=ri)
result = ri[ri[evenrows+2]-1] mod ncol
w = where(h[evenrows+1] eq 0, nw)
if nw gt 0 then result[evenrows[w]]=-1
```

-Jeremy.

Subject: Re: The IDL way: Find last non-zero value
Posted by penteado on Tue, 25 Aug 2009 01:35:31 GMT

View Forum Message <> Reply to Message

On Aug 24, 9:12 pm, Jeremy Bailin <astroco...@gmail.com> wrote:
> I'm shocked that no one's suggested histogram! Here's my take, which
> I'd wager does at least as well as the cumulative totals:
>
> sz=size(array,/dimen)
> ncol=sz[0]
> nrow=sz[1]
> evenrows = 2*lindgen(nrow)
> h = histogram((array ne 0) + rebin(reform(evenrows,
> 1,nrow),ncol,nrow), min=0, max=2*nrow-1, $
> reverse_indices=ri)
> result = ri[ri[evenrows+2]-1] mod ncol
> w = where(h[evenrows+1] eq 0, nw)
> if nw gt 0 then result[evenrows[w]]=-1
>
> -Jeremy.

Very nice! I will keep the idea in mind, for the case I need to do
something similar.

Subject: Re: The IDL way: Find last non-zero value
Posted by Jeremy Bailin on Tue, 25 Aug 2009 02:43:37 GMT
View Forum Message <> Reply to Message

On Aug 24, 8:12 pm, Jeremy Bailin <astroco...@gmail.com> wrote:
> On Aug 24, 4:42 pm, Eric Hudson <ehud...@mit.edu> wrote:
>
>> Thanks to all of you for the suggestions. I timed them all on several
>> of the larger matrices and found that the last suggestion (using
>> cumulative totals) was about twice as fast as the previous ones, and
>> about 5 times faster than the brute force loop. It did depend on how
>> many zeros ended each line on average. When the number was small the
>> loop could win, but for my typical matrices the cumulative total is
>> the clear winner.
>
>> Thanks again,
>> Eric
>
> I'm shocked that no one's suggested histogram! Here's my take, which
> I'd wager does at least as well as the cumulative totals:
>
> sz=size(array,/dimen)
> ncol=sz[0]
> nrow=sz[1]
> evenrows = 2*lindgen(nrow)

> h = histogram((array ne 0) + rebin(reform(evenrows,
> 1,nrow),ncol,nrow), min=0, max=2*nrow-1, $
> reverse_indices=ri)
> result = ri[ri[evenrows+2]-1] mod ncol
> w = where(h[evenrows+1] eq 0, nw)
> if nw gt 0 then result[evenrows[w]]=-1
>
> -Jeremy.

As usual, stupid typo in the last line, which should be:

if nw gt 0 then result[w]=-1

It looks to me like it doesn't beat the cumulative total version after all, at least on my test case. Oh well! Eric, if you have the timings on them, I'd be very curious to see how they stack up in detail!

-Jeremy.

Subject: Re: The IDL way: Find last non-zero value
Posted by Jeremy Bailin on Tue, 25 Aug 2009 02:53:10 GMT
View Forum Message <> Reply to Message

On Aug 23, 11:39 pm, Gianguido <gianguido.cia...@gmail.com> wrote:
> Another option - given that your array contains only positive numbers,
> I thought total(array, 1, /cumulative) could be handy:
>
> array=[[1,2, 0, 1, 0, 0],$
> [0, 0, 3,4, 1, 0],$
> [1, 5,6, 0, 0, 0],$
> [0,0,0,0,0,0], $
> [1, 0,0,0,0,0]]
>
> tc=total(array, 1, /cumulative)
> blah=max(tc, dim=1, w)
> result=(array_indices(tc,w))[0,*]
>
> t=total(array, 1)
> result[where(t eq 0)]=-1
>
> Gianguido

You can substitute array with abs(array) or (array ne 0) if there are negative values, so this method is completely general!

-Jeremy.
