Subject: Re: Help with getting rid of a FOR loop
Posted by Jean H. on Tue, 20 May 2008 22:55:46 GMT
View Forum Message <> Reply to Message

> dist=sqrt((xx-xcenter)^2+(yy-ycenter)^2) ;array of radii
>
> mask=fltarr(imsize,imsize)-1
>
> FOR i=0,num-1 DO BEGIN
> wh=where(dist GE r[i] and dist LE r[i+1])
> mask[wh]=i
> ENDFOR
>
> END
>
> I would like to find some way to get rid of the FOR loop at the end.
> All I'm doing in that loop is going through the annuli one by one,
> finding the pixels in that annuli, and setting the corresponding
> pixels in mask to the correct mask value.
>
> Thanks for any help anyone can provide!
>
> Nathan Goldbaum

Hi Nathan,

if your computer memory permits it, you can
1) reform your dist array so it is now a n_elements(dist) *
n_elements(r) array. basically, you will copy the distances
n_elements(r) times.
2) reform your r array so it is now a n_elements(dist) * n_elements(r)
array.
3) shift the array from (2) by 1
4) do where(new_dist GT new_r and new_dist LT new_r_plus_1)
5) divide the returned index by n_elements(r). You will know, for each
r, which elements satisfies your condition!

Sorry if it is not too clear... that's a "quick answer before to leave"...
Jean

Subject: Re: Help with getting rid of a FOR loop
Posted by pgrigis on Tue, 20 May 2008 23:20:01 GMT
View Forum Message <> Reply to Message

You may try the following strategy:

- compute the distance from Sun center for each pixel, store in r

- histogram the array r from 0 to sunradius in "num" bins
- caveat: this will give 1dim array indices. If you need back 2 dim
array
(probably not) use array_indices

Not sure if it going to be faster, but is worth a try.

Cheers,

Paolo

nathan12343 wrote:
> Hi, all
>
> Recently, I've been going through some old code in an effort to make
> it run a bit faster. My primary tool in this adventure has been the
> elimination of unnecessary FOR loops - all this code was written
> before I read David Fanning's article on 'The IDL Way'.
>
> This particular piece of code is used to approximate the center to
> limb variation in full-disk solar images. It identifies pixels that
> are on the solar disk and assigns them a value according to whether
> they're in between two radii given in a vector, r. This defines a set
> of annuli which cover the solar disk, the number of which is given by
> num. This information is stored in the mask array and returned.
> Here's the actual code:
>
> PRO generate_annuli,radius,xcenter,ycenter,num,mask
>
> ;radius - solar radius
> ;xcenter - x coord of solar disk center
> ;ycenter - y coord of solar disk center
> ;num - number of annuli for CLV approximation
> ;mask - the output mask
>
> ; No integer overflows
> compile_opt IDL2
>
> imsize=2048L ;images are 2048X2048
> mu=reverse(findgen(num+1))/(num) ;Generates values for mu
> r=radius*sqrt(1-mu^2) ;Inner radii for all the
> annuli
>
> xx=rebin(findgen(imsize),imsize,imsize) ;array of x indices
> yy=transpose(xx) ;array of y indices
>
> dist=sqrt((xx-xcenter)^2+(yy-ycenter)^2) ;array of radii
>

> mask=fltarr(imsize,imsize)-1
>
> FOR i=0,num-1 DO BEGIN
> wh=where(dist GE r[i] and dist LE r[i+1])
> mask[wh]=i
> ENDFOR
>
> END
>
> I would like to find some way to get rid of the FOR loop at the end.
> All I'm doing in that loop is going through the annuli one by one,
> finding the pixels in that annuli, and setting the corresponding
> pixels in mask to the correct mask value.
>
> Thanks for any help anyone can provide!
>
> Nathan Goldbaum

Subject: Re: Help with getting rid of a FOR loop
Posted by nathan12343 on Tue, 20 May 2008 23:24:37 GMT
View Forum Message <> Reply to Message

On May 20, 4:55 pm, Jean H <jghas...@DELTHIS.ucalgary.ANDTHIS.ca>
wrote:
> Hi Nathan,
>
> if your computer memory permits it, you can
> 1) reform your dist array so it is now a n_elements(dist) *
> n_elements(r) array. basically, you will copy the distances
> n_elements(r) times.
> 2) reform your r array so it is now a n_elements(dist) * n_elements(r)
> array.
> 3) shift the array from (2) by 1
> 4) do where(new_dist GT new_r and new_dist LT new_r_plus_1)
> 5) divide the returned index by n_elements(r). You will know, for each
> r, which elements satisfies your condition!
>
> Sorry if it is not too clear... that's a "quick answer before to leave"...
> Jean

Just what I was looking for. I'll write some code later - hope
there's enough memory.

Much appreciated!

Subject: Re: Help with getting rid of a FOR loop
Posted by pgrigis on Tue, 20 May 2008 23:33:36 GMT
View Forum Message <> Reply to Message

Jean H wrote:
>> dist=sqrt((xx-xcenter)^2+(yy-ycenter)^2) ;array of radii
>>
>> mask=fltarr(imsize,imsize)-1
>>
>> FOR i=0,num-1 DO BEGIN
>> wh=where(dist GE r[i] and dist LE r[i+1])
>> mask[wh]=i
>> ENDFOR
>>
>> END
>>
>> I would like to find some way to get rid of the FOR loop at the end.
>> All I'm doing in that loop is going through the annuli one by one,
>> finding the pixels in that annuli, and setting the corresponding
>> pixels in mask to the correct mask value.
>>
>> Thanks for any help anyone can provide!
>>
>> Nathan Goldbaum
>
> Hi Nathan,
>
> if your computer memory permits it, you can
> 1) reform your dist array so it is now a n_elements(dist) *
> n_elements(r) array. basically, you will copy the distances
> n_elements(r) times.
> 2) reform your r array so it is now a n_elements(dist) * n_elements(r)
> array.
> 3) shift the array from (2) by 1
> 4) do where(new_dist GT new_r and new_dist LT new_r_plus_1)
> 5) divide the returned index by n_elements(r). You will know, for each
> r, which elements satisfies your condition!

I guess that the original problem is not so much that for loops are
slow,
but that "where" is slow. So I fear that the above strategy won't gain
much speed, as now where must work on a much larger array...

Ciao,
Paolo

>
> Sorry if it is not too clear... that's a "quick answer before to leave"...
> Jean

Subject: Re: Help with getting rid of a FOR loop
Posted by nathan12343 on Tue, 20 May 2008 23:48:25 GMT
View Forum Message <> Reply to Message

On May 20, 5:33 pm, pgri...@gmail.com wrote:
> Jean H wrote:
>>> dist=sqrt((xx-xcenter)^2+(yy-ycenter)^2) ;array of radii
>
>>> mask=fltarr(imsize,imsize)-1
>
>>> FOR i=0,num-1 DO BEGIN
>>> wh=where(dist GE r[i] and dist LE r[i+1])
>>> mask[wh]=i
>>> ENDFOR
>
>>> END
>
>>> I would like to find some way to get rid of the FOR loop at the end.
>>> All I'm doing in that loop is going through the annuli one by one,
>>> finding the pixels in that annuli, and setting the corresponding
>>> pixels in mask to the correct mask value.
>
>>> Thanks for any help anyone can provide!
>
>>> Nathan Goldbaum
>
>> Hi Nathan,
>
>> if your computer memory permits it, you can
>> 1) reform your dist array so it is now a n_elements(dist) *
>> n_elements(r) array. basically, you will copy the distances
>> n_elements(r) times.
>> 2) reform your r array so it is now a n_elements(dist) * n_elements(r)
>> array.
>> 3) shift the array from (2) by 1
>> 4) do where(new_dist GT new_r and new_dist LT new_r_plus_1)
>> 5) divide the returned index by n_elements(r). You will know, for each
>> r, which elements satisfies your condition!
>
> I guess that the original problem is not so much that for loops are
> slow,
> but that "where" is slow. So I fear that the above strategy won't gain
> much speed, as now where must work on a much larger array...
>
> Ciao,
> Paolo
>
>
>

>> Sorry if it is not too clear... that's a "quick answer before to leave"...
>> Jean
>
>

Will histogram work with unevenly spaced bins?

Subject: Re: Help with getting rid of a FOR loop
Posted by nathan12343 on Wed, 21 May 2008 02:36:26 GMT
View Forum Message <> Reply to Message

On May 20, 5:48 pm, nathan12343 <nathan12...@gmail.com> wrote:
> On May 20, 5:33 pm, pgri...@gmail.com wrote:
>
>
>
>> Jean H wrote:
>>>> dist=sqrt((xx-xcenter)^2+(yy-ycenter)^2) ;array of radii
>
>>>> mask=fltarr(imsize,imsize)-1
>
>>>> FOR i=0,num-1 DO BEGIN
>>>> wh=where(dist GE r[i] and dist LE r[i+1])
>>>> mask[wh]=i
>>>> ENDFOR
>
>>>> END
>
>>>> I would like to find some way to get rid of the FOR loop at the end.
>>>> All I'm doing in that loop is going through the annuli one by one,
>>>> finding the pixels in that annuli, and setting the corresponding
>>>> pixels in mask to the correct mask value.
>
>>>> Thanks for any help anyone can provide!
>
>>>> Nathan Goldbaum
>
>>> Hi Nathan,
>
>>> if your computer memory permits it, you can
>>> 1) reform your dist array so it is now a n_elements(dist) *
>>> n_elements(r) array. basically, you will copy the distances
>>> n_elements(r) times.
>>> 2) reform your r array so it is now a n_elements(dist) * n_elements(r)
>>> array.
>>> 3) shift the array from (2) by 1
>>> 4) do where(new_dist GT new_r and new_dist LT new_r_plus_1)

>>> 5) divide the returned index by n_elements(r). You will know, for each
>>> r, which elements satisfies your condition!
>
>> I guess that the original problem is not so much that for loops are
>> slow,
>> but that "where" is slow. So I fear that the above strategy won't gain
>> much speed, as now where must work on a much larger array...
>
>> Ciao,
>> Paolo
>
>>> Sorry if it is not too clear... that's a "quick answer before to leave"...
>>> Jean
>
> Will histogram work with unevenly spaced bins?

Histogram does work for irregular binsizes if you use VALUE_LOCATE, I
think I'll be able to do this using histogram.

Subject: Re: Help with getting rid of a FOR loop
Posted by nathan12343 on Wed, 21 May 2008 03:02:48 GMT
View Forum Message <> Reply to Message

On May 20, 8:36 pm, nathan12343 <nathan12...@gmail.com> wrote:
> On May 20, 5:48 pm, nathan12343 <nathan12...@gmail.com> wrote:
>
>
>
>> On May 20, 5:33 pm, pgri...@gmail.com wrote:
>
>>> Jean H wrote:
>>>> > dist=sqrt((xx-xcenter)^2+(yy-ycenter)^2) ;array of radii
>
>>>> > mask=fltarr(imsize,imsize)-1
>
>>>> > FOR i=0,num-1 DO BEGIN
>>>> >   wh=where(dist GE r[i] and dist LE r[i+1])
>>>> >   mask[wh]=i
>>>> > ENDFOR
>
>>>> > END
>
>>>> > I would like to find some way to get rid of the FOR loop at the end.
>>>> > All I'm doing in that loop is going through the annuli one by one,
>>>> > finding the pixels in that annuli, and setting the corresponding
>>>> > pixels in mask to the correct mask value.
>

>>>> > Thanks for any help anyone can provide!
>
>>>> > Nathan Goldbaum
>
>>>> Hi Nathan,
>
>>>> if your computer memory permits it, you can
>>>> 1) reform your dist array so it is now a n_elements(dist) *
>>>> n_elements(r) array. basically, you will copy the distances
>>>> n_elements(r) times.
>>>> 2) reform your r array so it is now a n_elements(dist) * n_elements(r)
>>>> array.
>>>> 3) shift the array from (2) by 1
>>>> 4) do where(new_dist GT new_r and new_dist LT new_r_plus_1)
>>>> 5) divide the returned index by n_elements(r). You will know, for each
>>>> r, which elements satisfies your condition!
>
>>> I guess that the original problem is not so much that for loops are
>>> slow,
>>> but that "where" is slow. So I fear that the above strategy won't gain
>>> much speed, as now where must work on a much larger array...
>
>>> Ciao,
>>> Paolo
>
>>>> Sorry if it is not too clear... that's a "quick answer before to leave"...
>>>> Jean
>
>> Will histogram work with unevenly spaced bins?
>
> Histogram does work for irregular binsizes if you use VALUE_LOCATE, I
> think I'll be able to do this using histogram.

Thanks for the histogram suggestion!

This code is about 15 times faster than it was before, I'm glad I
learned about histogram :)

-Nathan
