Subject: help - speeding up a loop
Posted by natha on Mon, 11 Jan 2016 17:29:37 GMT
View Forum Message <> Reply to Message

Hi guys,

I am trying to implement a circular smooth on an irregular x, y grid.
The following loop takes too much time. How do you think I could make it faster?

```
  for i=0L, n_rang-1 do for j=0L, n_azim-1 do begin

    distkm=sqrt((xx-xx[i,j])^2. + (yy-yy[i,j])^2.)

    ww=where(distkm lt 5.,nn_w)
    if nn_w gt 0 then data_res[i,j]=total(data[ww]) / nn_w

  endfor
```

Thank you for your help,
nata

Subject: Re: help - speeding up a loop
Posted by greg.addr on Wed, 13 Jan 2016 11:42:40 GMT
View Forum Message <> Reply to Message

Maybe if you set up a regular grid with suitable spacing for the smoothed data - which will likely include far fewer points - and loop through that one point by point? If you need the data back on the original grid, you could sample back from it without a loop.

cheers,
Greg

Subject: Re: help - speeding up a loop
Posted by Burch on Wed, 13 Jan 2016 14:26:23 GMT
View Forum Message <> Reply to Message

On Monday, January 11, 2016 at 11:29:42 AM UTC-6, nata wrote:
> Hi guys,
>
> I am trying to implement a circular smooth on an irregular x, y grid.
> The following loop takes too much time. How do you think I could make it faster?
>
```
>   for i=0L, n_rang-1 do for j=0L, n_azim-1 do begin
>
>     distkm=sqrt((xx-xx[i,j])^2. + (yy-yy[i,j])^2.)
```

```
>
>       ww=where(distkm lt 5.,nn_w)
>       if nn_w gt 0 then data_res[i,j]=total(data[ww]) / nn_w
>
>     endfor
>
>  Thank you for your help,
>  nata
```

There are some quick changes that can be made for modest speed improvements. For example, compare this with the original:

```
for i=0L, n_rang-1 do for j=0L, n_azim-1 do begin

  deltaX = xx - xx[i,j]
  deltaY = yy - yy[i,j]
  distkm_squared=(deltaX*deltaX + deltaY*deltaY)
  ww=where(distkm_squared lt 25.,nn_w)
  if nn_w gt 0 then data_res[i,j]=total(data[ww]) / nn_w

endfor
```

-Jeff

Subject: Re: help - speeding up a loop
Posted by Burch on Wed, 13 Jan 2016 15:38:57 GMT

View Forum Message <> Reply to Message

```
On Wednesday, January 13, 2016 at 8:26:26 AM UTC-6, Jeff B wrote:
>  On Monday, January 11, 2016 at 11:29:42 AM UTC-6, nata wrote:
>>  Hi guys,
>>
>>  I am trying to implement a circular smooth on an irregular x, y grid.
>>  The following loop takes too much time. How do you think I could make it faster?
>>
>>     for i=0L, n_rang-1 do for j=0L, n_azim-1 do begin
>>
>>       distkm=sqrt((xx-xx[i,j])^2. + (yy-yy[i,j])^2.)
>>
>>       ww=where(distkm lt 5.,nn_w)
>>       if nn_w gt 0 then data_res[i,j]=total(data[ww]) / nn_w
>>
>>     endfor
>>
>>  Thank you for your help,
>>  nata
>
```

```
>  There are some quick changes that can be made for modest speed improvements. For example, compare this with the original:
>
>  for i=0L, n_rang-1 do for j=0L, n_azim-1 do begin
>
>    deltaX = xx - xx[i,j]
>    deltaY = yy - yy[i,j]
>    distkm_squared=(deltaX*deltaX + deltaY*deltaY)
>    ww=where(distkm_squared lt 25.,nn_w)
>    if nn_w gt 0 then data_res[i,j]=total(data[ww]) / nn_w
>
>  endfor
>
>  -Jeff
```

After firing up IDL and running a few tests, here are my code and results:

```
nRang = 200
nAzim = 200

xx = (randomu(7.0, [nRang, nAzim]) - 0.5)*1000.0
yy = (randomu(13.0, [nRang, nAzim]) - 0.5)*1000.0


clock = tic('- Original code')
for i=0l, nRang-1 do for j=0l, nAzim-1 do begin
  distkm = sqrt((xx-xx[i,j])^2 + (yy-yy[i,j])^2)
  ww = where(distkm lt 5.0, nn_w)
endfor
toc, clock


clock = tic('- Without sqrt()')
for i=0l, nRang-1 do for j=0l, nAzim-1 do begin
  distkm_squared = (xx-xx[i,j])^2 + (yy-yy[i,j])^2
  ww = where(distkm_squared lt 25.0, nn_w)
endfor
toc, clock

clock = tic('- Without sqrt() and rewriting array^2 to be array*array')
for i=0l, nRang-1 do for j=0l, nAzim-1 do begin
  deltaX = xx - xx[i,j]
  deltaY = yy - yy[i,j]
  distkm_squared = (deltaX*deltaX + deltaY*deltaY)
  ww = where(distkm_squared lt 25.0, nn_w)
endfor
toc, clock
```

% Time elapsed - Original code: 25.422446 seconds.
% Time elapsed - Without sqrt(): 18.863389 seconds.
% Time elapsed - Without sqrt() and rewriting array^2 to be array*array: 7.8514180 seconds.

-Jeff

Subject: Re: help - speeding up a loop
Posted by natha on Wed, 13 Jan 2016 16:57:28 GMT
View Forum Message <> Reply to Message

Hi Jeff,

Thank you for all your comments and tests! I am impressed to see the difference in computation time between array^2 and array*array.
I will implement this last version for sure... I am also parallelizing the code.

Thank you for all your comments,
nata
