
Subject: Re: for loop is killing me

Posted by [tarequeaziz](#) on Tue, 06 Nov 2007 15:24:45 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Nov 6, 8:13 am, Conor <cmanc...@gmail.com> wrote:

> On Nov 6, 1:31 am, tarequea...@gmail.com wrote:

>

>> Hello all,

>

>> help me! I am running a code 'infested' with for loops. And as u can

>> guess, its painfully slow. The part of my code with loop looks like

>> the following:

>

>> jump1:

>

>> ;Setting up the Green functions array which will have the same number

>> of elements as of amp_vec

>

>> G=fltarr(n_elements(amp_vec),n_elements(amp_vec))

>> G_phi=fltarr(n_elements(amp_vec))

>> G_rp=fltarr(n_elements(amp_vec))

>> G_r=fltarr(n_elements(amp_vec))

>> G_in=fltarr(n_elements(amp_vec))

>> G_out=fltarr(n_elements(amp_vec))

>> G_p=fltarr(n_elements(amp_vec))

>> G_h=fltarr(n_elements(amp_vec))

>> gp=fltarr(n_elements(amp_vec))

>> gh=fltarr(n_elements(amp_vec))

>> G_in_msum = fltarr(n_elements(amp_vec))

>> G_out_msum = fltarr(n_elements(amp_vec))

>> G_in_phisum = fltarr(n_elements(amp_vec))

>> G_out_phisum = fltarr(n_elements(amp_vec))

>

>> if ignore_realdata eq 1. then begin

>> amp_vec= abs(randomn(0.5,n_radial_points,/double))

>> a=1.

>> ;print,' amp_vec is',amp_vec

>> endif

>

>> do_u_like_to_start_with_rp_loop = 0 ; Set 1 if YES,0 if NO

>

>> if do_u_like_to_start_with_rp_loop eq 1. then BEGIN

>> PRINT,' WE ARE USING rp PREFERED LOOP'

>> goto,jump2

>

>> ENDIF

>

>> PRINT,' WE ARE USING r PREFERED LOOP'

```

>>         for i_r= 0,n_radial_points-1 do begin
>
>>             loop_time = systime(1)
>
>>             if ignore_realdata eq 1. then begin
>
>>                 r_scl=( i_r*2.)/(n_radial_points -1)    ;-----> We took off
>> trap radius from here for scaling purpose,see note
>
>>                                     ;-----> Also note the use of factor 2. This is because, we
>> want to get values outside
>>                                     ;-----> the cylinder. But
this is just for demonstration purpose!!!!
>
>>             endif else begin
>
>>                 r_scl=( i_r)/(n_radial_points -1)        ; took off the factor 2
>
>>             endelse
>>             ;r_scl = 2.
>
>>             ;print,'r_scl value is ',r_scl
>
>>             for i_rp= 0,n_radial_points-1 do begin
>
>>                 if ignore_realdata eq 1. then begin
>
>>                     rp_mat = randomn(1,n_radial_points,/double)
>>                     ;print,'rp_mat is',rp_mat
>
>>                     rp_scl = rp_mat(i_rp)
>
>>                     endif else begin
>
>>                         ;rp_scl=( i_rp/((n_radial_points-1))) * float(amp_vec(i_rp))
>>                         rp_scl= float(amp_vec(i_rp))
>>                         ;print,' amp_vec is',amp_vec
>
>>                     endelse
>
>>                         ; rp_scl=( i_rp/((n_radial_points-1))) * float(amp_vec(i_rp))
>
>>                         ;rp_mat = randomn(1,n_radial_points,/double)
>>                         ; print,'rp_mat is',rp_mat
>
>>                         ;rp_scl = rp_mat(i_rp)
>
>>                         ;rp_scl = 0.5

```

```

>
>> ;;print,'rp_scd for i_rp= ',i_rp,' is, ',rp_scd
>
>>         if (r_scd gt rp_scd) then begin
>>             r_plus = r_scd
>>             r_minus = rp_scd
>>         endif else begin
>>             r_minus = r_scd
>>             r_plus = rp_scd
>>         endelse
>
>>         ;print,'r_plus value is ',r_plus
>>         ;print,'r_minus value is ',r_minus
>
>>         for i_m=1,20 do begin      ;n_radial_points do begin
>
>>         ;;print,'starting i_m value is ',i_m
>
>>         ;if (n_radial_points le 5) then i_phi_max = 20 else i_phi_max
=
>> n_radial_points
>>         ;print,'i phi max is = ',i_phi_max
>>         i_phi_max = 20.
>
>>         for i_phi=0,i_phi_max - 1 do begin
>>             ; print,'starting i_phi value is',i_phi
>
>>             count= i_phi + 1.
>>             ;print,'count is ',count
>>             phi=(count*2*PI)/(i_phi_max)
>
>>             ; print,'phi value is= ',phi
>>             ; phi= 0.785398163 ; <----- In radian
>
>>             a= 1. ; -----> Scaled trap radius
>
>>             gp(i_phi) = 2.0*(1./i_m)*(r_minus/r_plus)^i_m *
>> cos(i_m*(phi))
>>             gh(i_phi) = 2.0*(1./i_m)*(r_minus *
r_plus/a^2)^i_m *
>> cos(i_m*(phi))
>
>>             ;print,'cos(i_m*phi) is',cos(i_m*phi)
>
>>             ;if (i_r eq 0.) && (i_rp eq 0.) && (i_m eq 1.) then
begin
>
>>             ;print,' the 1st gp is ',gp(i_phi)

```

```

>>                                     ;print,' the 1st gh is ',gh(i_phi)
>>                                     ;endif
>
>>                                     ;if (i_r eq n_radial_points - 1.) && (i_rp eq
>> n_radial_points -1.) && (i_m eq n_radial_points -1.) then begin
>
>>                                     ;print,' the Last gp is ',gp(i_phi)
>>                                     ;print,' the Last gh is ',gh(i_phi)
>
>>                                     ;endif
>
>>                                     endfor ; end of phi loop
>
>>                                     G_in_phisum [i_m -1] = total(gp) ; Add up the phi
elements
>> for a specific m
>>                                     G_out_phisum [i_m -1] = total(gh)
>
>>                                     endfor ; end of m-loop
>
>>                                     G_in_msum(i_rp) = total(G_in_phisum) ; Add up all m values for
>> a specific rp value
>>                                     G_out_msum(i_rp) = total(G_out_phisum)
>
>>                                     G_in(i_rp) = -alog((r_plus)^2.) + G_in_msum[i_rp]
>>                                     G_out(i_rp) = -alog((a^2./rp_sclD)^2.) + G_out_msum[i_rp]
>
>>                                     ;print,'G_in is',G_in
>>                                     ;print,'G_out is',G_out
>
>>                                     G(i_r,i_rp)= G_in(i_rp) - G_out(i_rp) - alog((a/rp_sclD)^2.)
>
>>                                     index=where(finite(G,/NaN) ,count)
>>                                     ;print,' index is ',index
>>                                     if (count ne 0) then G[index] = 0.
>> ;                                     non_zero = where(G,count)
>> ;                                     print,'non zero value is',non_zero
>> ;
>> ;                                     if count ne 0. then G = G[non_zero]
>
>>                                     ;print,' last log part',alog((a/rp_sclD)^2.)
>>                                     ;print,' G_in - G_out is ',G_in - G_out
>>                                     ;print,'G inside rp loop is ',G(i_r,i_rp),' for i_r =',i_r,'
>> and i_rp =',i_rp
>
>>                                     endfor ;end of rp loop
>
>>                                     ;G_p= G_in

```

```

>>             ;G_h= G_out
>>             ;print,'G_p is',G_p,'and G_h is',G_h
>>             ;G(i_r,i_rp)= G_p(i_rp) - G_h(i_rp) - alog((a/rp_scd)^2.)
>
>>             ;print,'rp_scd is ',rp_scd
>
>>             ;    print,' G inside the r loop is',G
>
>>             ;    if i_r eq (n_radial_points-1) then print,'Last r_scd value is=',
>> r_scd
>
>>             ;if i_r eq ( (n_radial_points-1)/10.) then print,'First 1/10th
>> r_scd value is=', r_scd
>>             ;if i_r eq ( (n_radial_points-1)/2.) then print,'First 1/2th
>> r_scd value is=', r_scd
>>             ;if i_r eq ( (n_radial_points-1)*3./4.) then print,'3/4 th r_scd
>> value is=', r_scd
>>             ;if i_r eq (n_radial_points-1) then print,'Last r_scd value is=',
>> r_scd
>
>>             if i_r eq (n_radial_points - 1.) then begin
>
>>             print,'The time it took to finish ',i_r,'th loop is', systime(1) -
>> loop_time,'seconds'
>
>>             endif
>
>>             endfor ; end of r loop
>
>>             ;print,'here is the PROFILER report for r preferred loop'
>
>>             ;print,'G is ',G      ;[n_radial_points - 1]
>
>> if ignore_realdata eq 1 then begin
>
>>             r_vec=(findgen(n_radial_points)/(n_radial_points))
>>             potential=fltarr(n_radial_points)
>>             prod = fltarr(n_radial_points)
>
>> endif else begin
>
>>             r_vec=(findgen(n_theta_points)/(n_theta_points))
>>             potential=fltarr(n_theta_points)
>>             prod = fltarr(n_theta_points)
>> endelse
>
>> if ignore_realdata eq 1 then p= n_radial_points else p= n_theta_points
>

```

```

>>         for k = 0,100 do begin ;p -1 do begin
>>             ;rp_mat = randomn(1,n_radial_points,/double)
>>             ;print,' amp_vec is',amp_vec
>>             ;print,'amp_vec is',amp_vec[k]
>>             ;print,'G[* ,k] is',G[* ,k]
>>             prod =amp_vec[k] * G(* ,k)
>>             prod_invfft = abs( fft(prod,/inverse))
>>             potential[k] = int_tabulated(r_vec,prod_invfft)
>>             ;print,'potential is',potential
>
>> =====
>> So as u can see there are 4 for loops running. the m loop with
>> running index i_m should be 200. But setting it to 200 slows down the
>> program.
>
>> My supervisor said that, this should be a very fast way of calculating
>> green function than the traditional way(which my group-mate is doing).
>
>> what I am doing wrong?
>> Any help will be HIGHLY appreciated!
>
>> Best,
>
>> Tareque
>
> Well, the first thing you are doing wrong is posting a gigantic chunk
> of code for people to go over! I'd love to help, but I really don't
> have time to re-write the whole thing for you. Can you break it down
> into a couple smaller sections? If you can summarize what each chunk
> of code is actually trying to do, isolated from the rest of the code
> as a whole, that will be extremely helpful.
>
> For a first suggestion for now (I have class to teach in a couple
> minutes) I'd point out that your array declarations can be sped up,
> even though those aren't in a for loop. Look:
>
> G=fltarr(n_elements(amp_vec),n_elements(amp_vec))
> G_phi=fltarr(n_elements(amp_vec))
> G_rp=fltarr(n_elements(amp_vec))
> G_r=fltarr(n_elements(amp_vec))
>
> should be:
>
> namp_vec = n_elements(amp_vec)
> g = fltarr(namp_vec,namp_vec)
> g_phi = fltarr(namp_vec)
> g_rp = fltarr(namp_vec)
> g_r = fltarr(namp_vec)

```

>
> or even:
>
> namp_vec = n_elements(amp_vec)
> g = fltarr(namp_vec,namp_vec)
> g_phi = fltarr(namp_vec)
> g_rp = g_phi & g_r = g_phi
>
> In the long run this won't make a big difference for the speed of your
> code, but it does help highlight a very important lesson when you are
> trying to optimize your code for speed - avoid all unnecessary
> computation. Something like that in a for loop that iterates many
> times can make a difference.
>
> Anyway, please, try to extract a few sections of your code and explain
> what they are doing independent of the rest of your code. I really
> doubt anyone will have the time to look through the whole thing for
> you.

Thanks for the reply. Really appreciated it. I am going to cut down
the code into smaller pieces and post it here.

Tareque
