Subject: Fitting plasma waveforms with 10^6 variable combos!
Posted by ted1508 on Wed, 07 Dec 2016 07:08:31 GMT
View Forum Message <> Reply to Message

Hello readers!

This is my first post here and a Hail Mary in a desperate attempt to speed up my IDL code...

I am presenting electric field antenna measurements from dust impacts onto the MAVEN spacecraft around Mars at the AGU conference next week. My code compares these waveforms (600 points) to a theoretical model, of which I have 5 variables, 4 with 10 possible points and 1 with 100 (the higher the resolution the variables the better!).

Problem is, 10^6 loops is a lot to go through. I tried doing it all in massive arrays, and its not much faster (IDL's memory allotment issue versus overhead with for loops)...

At this point I am thinking it would be faster to learn C and recode it from scratch then let this run for a week.

...

Part of the problem is that I am using this function (twice!) for each of the 600 time points:

```
function fun_ant,time,to,Q,deltat,C_ant,dtprime,tao_ant
  tprime=findgen(100)*dtprime
  one=  exp((-(time-tprime-to)^2)/(2*(deltat)^2))*exp(-(tprime)/tao_ ant)
  constant=(-Q/(C_ant*(sqrt(2*!pi)*(deltat))))
  equation=constant*Total(one)*dtprime
  return, equation
end
```

The variables which I will be looping through are time, deltat, tao_ant/bod, and Q_ant/bod (for plasma charge recollection to MAVEN's antenna and body respectively)

...

Right now the main procedure runs roughly as follows:

```
 for a = start_a,stop_a-1 do begin    ;;; loop through each waveform (there are 5000)

    input_max=max(mf_hits[a,*],mf_max_loc)    ;;; used to center waveforms later

    for ta = 1., 9. do begin                ;;;; now all the combinations...
      tao_ant = 0.0001*double(ta)

      for tb = 1., 9. do begin
        tao_bod = 0.0001*double(tb)
        print,tb
        for dt = 1., 9. do begin
          deltat = 0.00001*double(dt)

          for qa = 0, 9. do begin
            q_ant = 0.000000000001*double(qa)

            for qb = 1, 9. do begin
              q_bod = 0.000000000001*double(qb)

              for n=0,n_elements(new_time)-1 do begin    ;;;; and the 600 time points.....
                newtime = new_time[n]   ;;; the function for this is defined above
                pot_ant[n] = fun_ant(newtime,to,q_ant,deltat,C_ant,dtprime,tao_ant)
                pot_bod[n] = fun_bod(newtime,to,q_bod,deltat,C_bod,dtprime,tao_bod)
              endfor

              pot = pot_ant - pot_bod    ;;; difference in potential between antenna and spacecraft

              Lag_mf = 25 ;;;; how much the 2 waveforms (mf_hits and pot) can shift

               similarity=double(max(c_correlate(mf_hits[a,mf_max_loc-103:m
f_max_loc+500],pot[0:603],lag_mf)))

              if similarity GT best_sim then begin
                ;;;; save the best variable values
              endif

            endfor    ; qb
          endfor    ; qa
        endfor    ; dt
      endfor    ; tb
    endfor    ; ta
  endfor   ; a
```

...

So theres my problem: I can choose between massive arrays or massive number of for loops, either way resulting in several days of computation.

For those thinking I could step through the values iteratively, there are local 'potential wells' where the similarity value will be higher than the neighbors but still will not be the best fit.

Does anyone have an idea if this code could be sped up by an order of magnitude?

Appreciate all your thoughts!
-Tenacious Ted