Subject: Re: Avoiding a FOR loop in calculation of SPH potential energy
Posted by cody on Tue, 23 Jun 2009 19:39:34 GMT
View Forum Message <> Reply to Message

On Jun 23, 7:21 am, Chris <beaum...@ifa.hawaii.edu> wrote:
> On Jun 22, 9:29 pm, cody <codyras...@gmail.com> wrote:
>
>
>
>> i've been reading through this discussion group and one thing i see
>> often is that you can vectorize a FOR loop to avoid it. so my code
>> would be something like:
>
>> u = 1 + bytarr(pn)
>> dx = u#s.x
>> dy = u#s.y
>> dz = u#s.z
>> for i = 0L, pn-1 do dx[0, i] = dx[*, i] - s.x
>> for i = 0L, pn-1 do dy[0, i] = dy[*, i] - s.y
>> for i = 0L, pn-1 do dz[0, i] = dz[*, i] - s.z
>> d = sqrt(dx^2+dy^2+dz^2)
>> print,'calculated all ds?'
>
>> but i'm not able to allocate that much memory for 100k particles and i
>> wouldn't know how to do the proper potential energy calculation that
>> way either since not all particles are the same mass.
>
> To distill the problem a little bit:
> say that m, x, y, z represent the mass and positions for each
> particle. Something like the following might work if the vectors were
> small
>
> sz = (number of particles)
> marr = rebin(m, sz, sz)
> marr_1 = rebin(1#m, sz, sz)
>
> dx = rebin(x, sz, sz) - rebin(1#x, sz, sz)
> dy = rebin(y, sz, sz) - rebin(1#y, sz, sz)
> dz = rebin(z, sz, sz) - rebin(1#z, sz, sz)
> delt = sqrt(dx^2 + dy^2 + dz^2)
>
> pe = marr * marr_1 / delt
> ;- diagonal elements are now infinity, and shouldn't be counted
> anyways
> ind = indgen(sz)
> pe[ind,ind] = 0
>
> ;- total and correct for double counting

> pe = total(pe) / 2.
>
> In short, this calculates the distance between every particle pair,
> and stores it in a 2D array. It then calculates the PE contribution
> between each of these pairs, zeroes out the diagonal (because a
> particle doesn't have any pe due to interaction with itself), totals
> it, and divides by 2 (since each pair was counted twice)
>
> Like you said, though, this wouldn't work with 100k elements (the
> arrays would be 10^10 elements large). Some people might try breaking
> up the array into small chunks (say of 100 elements each), calculate
> the PE of these chunks, and then patch that all together at the end.
> Its kind of a mess though (you still end up with nested loops, but
> they have 10^3 instead of 10^5 iterations). For these types of
> problems, IDL doesn't seem to have a great solution (save for the
> ability to call external C programs to do the heavy lifting)
>
> A more efficient algorithm, if you can get away with it, is to ignore
> the contribution to the potential energy from pairs of particles very
> far away from one another. In this case, you can use histograms to
> efficiently index nearby objects. This turns a n^2 algorithm into an
> essentially linear one. Seehttp://www.dfanning.com/code_tips/slowloops.html
>
> chris

do you have a recommendation for where to go to learn how to use a C/C+
+ program inside an IDL program? that seems like it would solve my
problem.
