Subject: Re: A case where FOR loops are unavoidable?
Posted by JD Smith on Mon, 27 Nov 2006 22:44:36 GMT
View Forum Message <> Reply to Message

On Mon, 27 Nov 2006 14:19:16 -0800, gknoke wrote:

> Hi all, I'm trying to setup an array for the computation of a rather
> tricky multidimensional function, but I'm seeing no way around using
> FOR loops in its creation. I've been trying to figure out an
> appropriate vector solution, but so far I'm drawing a blank.
>
> The equation goes as follows:
>
> f(x, b1, b2, p) = alog((1/b1)*(1-p)*exp(-x/b1) + (1/b2)*p*exp(-x/b2))
>
> Here x is a vector roughly of length 2^20, b1 and b2 are roughly
> 100-1000 elements, and p is roughly 10-50 elements. I realize that
> this is a larger array than is likely to fit in memory, but what I'm
> really after is the sum of this function in the x dimension, i.e. the
> final output should be something like:
>
> f_out(b1,b2,p) = total(f(x,b1,b2,p), 1)
>
> My approach to this is to loop over b1, b2, and p to take advantage of
> being able to use the total function to keep the array size manageable,
> like so:
>
> for k=1,np
> for i=1,nb1
> for j=1,nb2
> f_out(i,j,k) = total(f(x, b1[i], b2[j], p[k]), 1)
> etc. etc.
>
> This works, but I'm wondering if there are any better approaches?

You could vectorize the whole thing by making an nx x nb1 x nb2 x np array and totaling in the correct dimension, but a) it won't fit in memory so it will page to disk, and b) this looping method is probably already about as fast as it gets (other than redesigning your algorithm). If the contents of the inner loop take much longer to execute than the looping penalty, removing loops doesn't help much if at all, and in many cases as mentioned can get you into trouble with memory. The looping penalty isn't huge, typically around .1 to .5 microseconds, but that can be very noticeable where inner loop contents also take on the order of a microsecond. It's 10-100x longer than the overhead of a tight loop in C.

JD