
Subject: Re: FOR loops and efficiency

Posted by [Jeremy Bailin](#) on Fri, 22 May 2009 12:35:23 GMT

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

On May 21, 12:04 pm, Rachel <lotsofche...@gmail.com> wrote:

> It was impressed upon me sometime or another that the use of FOR loops
> in IDL applications tends to be an inefficient solution for doing many
> tasks, yet sometimes I have difficulty finding a reasonable
> alternative to the FOR loop. I was wondering if anyone could give me
> advice on the following example code.

>

> I am trying to make a function that takes arrays of parameters and
> then generates a mathematical model. In the following example I use
> gaussian curves, but generally I would want to expand an
> implementation to other mathematical functions (gaussians are just
> easy for this example).

> So basically I can accomplish what I want to do using something like
> the following:

>

> x = findgen(2000)*0.1 + 900.0

> y = fltarr(2000)+1.0

>

> lam0 = findgen(10)*50.0 + 900.0

> depth = findgen(10)/10.0

> width = findgen(10)

>

> for i = 0,n_elements(lam0)-1 do y = y *(1.0 - depth[i]*exp(-(x-width
> [i])^2/2.0/width[i]))

>

> I was thinking about how one might accomplish the same things without
> a for loop and I came up with the following... problem being that for
> large arrays of lam0 this is actually more inefficient (I'm assuming
> because of the use of extraordinarily large arrays)

>

> n = n_elements(x)

> nlines = n_elements(lam0)

> y = product(1.0 - rebin(transpose(depth),n,nlines)*exp(-(rebin

> (x,n,nlines)-rebin(transpose(lam0),n,nlines))^2/2.0/rebin(tr anspose
> (width),n,nlines)),2)

>

> any advise?

>

> Thanks!

> Josh

I don't know about the more complicated functions, but one thing here is that you're probably spending a lot of effort calculating array elements that are far enough away from the center of what is

effectively a 10D Gaussian that the their value is certainly 0 to machine precision. If you restrict the part of it you compute to the innermost N sigmas (where you can figure out an appropriate value of N, and probably just use max(width) to be safe for sigma), it'll take a lot longer to run into memory-related slowdowns.

That said, if your better model functions don't fall to zero as quickly as a Gaussian, that may not be relevant... and as Craig says, the for loop may not be the limiting factor in your case.

-Jeremy.
