Subject: Re: how to speed up multiple regressions?
Posted by Craig Markwardt on Mon, 30 Apr 2001 18:35:38 GMT
View Forum Message <> Reply to Message

Charlotte DeMott <demott@atmos.colostate.edu> writes:

> Hi,
>
> I have some code to construct a composite of a
> meteorological phenomena in three dimensions (x, y, lag).
> The compositing index is a time series (ts) of a certain
> variable, and the data being composited (x, y, time) is
> regressed onto this compositing index. Because of the
> length of the time series and the size of the data array,
> and the fact that I do this compositing for multiple fields,
> I'm looking for ways to speed up the process, which is
> currently quite time consuming. The greatest amount of time
> seems to be spent in computing the significance of the
> correlation, rather than in computing the regressions. The
> regression is only done for periods where the signal is the
> "ts" time series is "big" (i.e., big = WHERE(ts GE
> threshold)).

Charlotte, I hate to say it but you have a severe case of loop-itis.
The success and speed of an IDL program handling large amounts of data
depends on vectorizing the key code. The second section of your code
has no vectorization whatsoever! No wonder it seems so slow. A
secondary benefit of vectorizing code is that it can help make the
code cleaner, since the mathematics are emphases over the loop
constructs.

But it's a little worse than that (groan :-). You call the T_CVF()
function, which computes the Student's T test. You call it for *each*
element of the loop, despite the fact that the arguments remain
constant. Arghh. This is an expensive function to calculate, so it
makes sense to factor it outside of the loop where it will only be
executed once.

I've only looked at the second section, the part you thought was too
slow. Here is my take on the situation:

```
datadof = float(big_count)/data_tau  ;; DOF's are a scalar!
tval = t_cvf(0.1, datadof)           ;; Student's T value, computed once

data_t = abs(datar*sqrt(datadof))/sqrt(1-datar*2)
datcomp = dataf(*,*,*,0) + dataf(*,*,*,1)*tsval
data_sig = datar*sqrt(datadof)/sqrt(1-datar*2) GT tval
```

You may be able to vectorize the first part a little better, but I'll
leave that to you.

Craig

--
--- ----------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
--- ----------

Subject: Re: how to speed up multiple regressions?
Posted by Charlotte DeMott on Mon, 30 Apr 2001 19:48:01 GMT
View Forum Message <> Reply to Message

Hi Craig,

Thanks for taking a look. I was hoping someone would point out to me a very obvious blunder I was making. I had high hopes upon reading your message, but I think I'm sticking all of this in a loop to compute the significance because datadof is NOT a constant for all points in my array. In the first loop I included in the original post, data_tau is the decorrelation timescale at each data point which is, unfortunately, not constant. T_CVF, as you indicated, requires the 2nd argument (datadof in my case) to be a scalar. My problem is that datadof isn't the same for all data points.

However, your post make me realize that I can do the regression in a slightly different way that will eliminate this problem, and save me loads of time.

So while your suggestion wasn't the fix I was looking for, it jarred my tired brain enough to think of another work-around. So thanks!

Charlotte

Craig Markwardt wrote:

> I've only looked at the second section, the part you thought was too
> slow. Here is my take on the situation:
>
>   datadof = float(big_count)/data_tau ;; DOF's are a scalar!
>   tval = t_cvf(0.1, datadof)   ;; Student's T value, computed once
>
>   data_t = abs(datar*sqrt(datadof))/sqrt(1-datar*2)
>   datcomp = dataf(*,*,*,0) + dataf(*,*,*,1)*tsval
>   data_sig = datar*sqrt(datadof)/sqrt(1-datar*2) GT tval
>
> You may be able to vectorize the first part a little better, but I'll

> leave that to you.
>
> Craig
>
> --
> ---------- ----------
> Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
> Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
> ---------- ----------

Subject: Re: how to speed up multiple regressions?
Posted by Craig Markwardt on Mon, 30 Apr 2001 21:04:11 GMT
View Forum Message <> Reply to Message

Charlotte DeMott <demott@atmos.colostate.edu> writes:

> Hi Craig,
>
> Thanks for taking a look. I was hoping someone would point out to me a very
> obvious blunder I was making. I had high hopes upon reading your message,
> but I think I'm sticking all of this in a loop to compute the significance
> because datadof is NOT a constant for all points in my array. In the first
> loop I included in the original post, data_tau is the decorrelation timescale
> at each data point which is, unfortunately, not constant. T_CVF, as you
> indicated, requires the 2nd argument (datadof in my case) to be a scalar. My
> problem is that datadof isn't the same for all data points.
>
> However, your post make me realize that I can do the regression in a slightly
> different way that will eliminate this problem, and save me loads of time.
>
> So while your suggestion wasn't the fix I was looking for, it jarred my tired
> brain enough to think of another work-around. So thanks!

Okay, so I didn't see that data_tau was variable, sorry :-)

I'm glad you figured out a way to do it. I wanted to mention a
possible way to get around the T_CVF problem. What you can do is
precompute a table of values and then use spline interpolation. Spine
interpolation using SPL_INIT and SPL_INTERP is vectorizable, so it
should be really fast. You can put a lot of samples into your table
so it can be quite precise (you might interpolate in log-log space if
your dynamic range is large).

Craig

--
---------- ----------

Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
---