Subject: Re: Avoiding Loops in IDL 8.2.2
Posted by Moritz Fischer on Mon, 24 Jun 2013 05:05:03 GMT
View Forum Message <> Reply to Message

Hi Nate,

transpose(reform(transpose(A),2,2,2),[1,0,2])

cheers

Am 24.06.2013 00:24, schrieb Nate Tellis:
> On Friday, 21 June 2013 17:04:31 UTC-7, Nate Tellis wrote:
>> Hi all,
>>
>>
>>
>> I have a series of 711x4096x3 arrays. I am searching for good fits
>> to a model, which is an 11x19 array, using a reduced chi-square
>> fit. As it is now, I step across, pixel by pixel, column by column,
>> pane by pane, and perform the fit to a subimage centred at the loop
>> indices (normalized to the value of the central pixel). The fit is
>> simple element-wise subtraction and squaring of the sub images,
>> followed by one call to 'total' on the sub-image:
>>
>>
>>
>> Chi^2_red = 1/Npixels * Sum over each pixel((image -
>> fit)^2/error^2)
>>
>>
>>
>> (This is of course fast, as the -, ^2, /, and 'total' operations
>> utilize the IDL thread pool)
>>
>>
>>
>> I know I can speed this up by using operations that leverage
>> multithreading. How can I go about avoiding these hated nested for
>> loops? Performing the fits on all ~8,500,000 subimages without
>> multithreading takes way too long - about 90 seconds on average.
>>
>>
>>
>> Thank you for the help,
>>

>> Nate
>
> Here's a simpler question. I think I can solve my problem if I can do
> this efficiently:
>
> Say I have an array like:
>
> A =
>
> 1 2 3 4 5 6 7 8
>
> where A is 4 by 2
>
> How can I use reform and rebin to get an array of dimension 2 by 2 by
> 2 that looks like
>
> 1 2 5 6
>
> 3 4 7 8
>
> Any help is much appreciated.
>
