Subject: Re: Sky Falling, etc. : Array substitution + addition with plus-equal (+=)
Posted by JD Smith on Tue, 13 Mar 2007 20:37:18 GMT
View Forum Message <> Reply to Message

On Tue, 13 Mar 2007 11:35:53 -0700, Ed Hyer wrote:

> Thanks for the great explanations, that one really had my head
> spinning.
>
>> One additional point is worth mentioning: when setting large arrays, the
>> "offset" method of specifying a single index on the LHS is much faster
>> than the '*' method of building the full index list to match up the
>> dimensions of left and right-hand side arrays:
>
> Well, yes. Which is how I came across this problem. The "offset"
> method does not work for "+=" or any of the "iterative" operations,
> because of the behavior you described.
>
> My application is building a grand [X,Y,T] sum, from contributions of
> a routine which returns an [X,Y,n] array, with n varying from, say, 1
> to 10. Without the increment operation, I have to do
>
> totals=fltarr(2,2,6);
> add=GetAdd(InputArgs,ReturnT0=T0); Get ADD and the initial index in T
> (T0)
> sz=size(add)
> if(sz[0] eq 2) then NT=1 else NT=sz[3]; get size of 3rd dimension
> T1=T0 + NT - 1
> previous_totals=totals[*,*,T0:T1]; extract subarray corresponding to
> ADD
> totals[0,0,t0] = previous_totals + add ; copy new totals back into
> TOTALS
>
> Which comes at considerable computational cost. But, that's life.
> Thanks again for the explanations.

Yes, that's an issue, but then again, without knowing in advance how large the computed array will be on the RHS, you can't blame IDL for not knowing to expand totals[0,0,t0] into a large array of the desired size. The only minor suggestion is that if you are doing totals[*,*,T0:T1] many times in an inner loop, you could pre-compute the corresponding index list as a properly dimensioned array, and subscript with that instead. Won't save memory but will save repeated recomputation of the same list of indices. Another option, if 'add' is a large chunk, is to keep a large array of zeroes the same size as 'totals', use the offset trick to stick 'add' into it, and then increment the entire 'totals' array. Probably won't save much unless 'add' is already a large fraction of the size of 'totals'.

JD