Subject: Re: Efficiently multiplying an array by a vector
Posted by Burch on Wed, 28 Sep 2016 12:50:22 GMT
View Forum Message <> Reply to Message

On Tuesday, September 27, 2016 at 9:51:47 PM UTC-5, wlandsman wrote:
> Three things:
>
> 1. Have you actually found the speed to be problematic? In general, looping over a single loop index is not a large speed penalty, and what you want to avoid is looping over 2 or more indices. The IDL maxim for efficient programming is not "avoid loops" but "try to get a much done as possible during each loop iteration". If your matrices have a million elements then you are already following this maxim.
>
> 2. The process can likely be sped up by avoiding the use of the asterisk on the left hand side.

> http://www.idlcoyote.com/code_tips/asterisk.html
>
> In your case, this would require a transpose
>
> c = transpose(c)
> FOR i=0,ncol-1 DO c[0,i] = a[i,*] * b
> c = transpose(c)
>
> though the first transpose can likely be incorporated into the creation of the c matrix.
>
> 3. I believe that Python does have the capability to efficiently multiply every column in a 2D array by a vector using "broadcasting"
>
> http://eli.thegreenplace.net/2015/broadcasting-arrays-in-num py/
>
> I don't believe that this capability is available in IDL but it would be a nice feature for Harris Geospatial to add.
>
> --Wayne
>
>
>
>
> On Tuesday, September 27, 2016 at 9:38:15 PM UTC-4, LH wrote:
>> The following text is a slightly edited version of a post by Mark Plonski several years ago that was never answered. Just saving typing time because my problem is the same. In fact, it seems like it must be a common problem when one tries to vectorize and speed up a program.
>>
>>
>> What is the most efficient way to multiply every col in a 2D array
>> (ncol x nrow) by a vector of length (ncol)?
>>
>> Example:

>>
>> input array   vector   output array
>>
>> a11 a12 a13   b1   a11b1 a12b1 a13b1
>> a21 a22 a23   b2   a21b2 a22b2 a23b2
>>
>>
>> This could be done by looping over the cols:
>>
>>   FOR i=0,ncol-1 DO c[i,*] = a[i,*] * b
>>
>> Is there a more efficient way (w.r.t. computational speed) to do this?
>>
>> I know I could replicate the column vector into a matrix (b # identity row)
>> and then do a ptwise matrix multiply, but my matrices can
>> be very large (1M elements) and I occasionally run out of swap
>> space. I don't know if that would run any faster anyway.

Expanding on #2 given by Wayne, I think you would benefit by altering how you're storing data and looping over multidimensional arrays. In IDL the elements of the first dimension are contiguous in memory (i.e, a[0,0], a[1,0], a[2,0] etc.). So indexing arrays like a[i,*] causes IDL to have to "jump around" in memory, whereas with a[*,i] IDL basically just has to go to one continuous chunk of memory.

For instance, compare the following two examples:

```
A =dblarr(10000, 5000) + 1d
B = dindgen(5000)
C = dblarr(10000, 5000)
nCol = 10000
tic
for i=0, nCol-1 do C[i,*] = A[i,*]*B
toc
% Time elapsed: 1.4337058 seconds.
```

```
A = dblarr(5000, 10000)+1d
B = dindgen(5000)
C = dblarr(5000, 10000)
nRow = 10000
tic
for i=0, nRow-1 do C[0,i] = A[*,i]*B
toc
% Time elapsed: 0.099428177 seconds.
```

Of course, the benefit you gain depends on the number of loops you have to do.
See Also: http://www.harrisgeospatial.com/docs/columns__rows__and_arra
y.html#arrays_3727706888_737332