Subject: Re: multiplication
Posted by John-David T. Smith on Tue, 28 Mar 2000 08:00:00 GMT
View Forum Message <> Reply to Message

James Kuyper wrote:
>
> meron@cars3.uchicago.edu wrote:
>>
>> In article <38E03BDC.868B8396@hotmail.com>, marc <m_schellens@hotmail.com> writes:
>>> Is there a function like TOTAL but for multiplication.
>>> Like the big PI symbol in mathematical notation.
>>> Or this really something for the for loop?
>>>
>>> I.E.
>>>
>>> a=[1,2,3,...]
>>>
>>> result=a[1]*a[2]*a[3]...
>>>
>> if all the elements of a are positive then you can simply do
>>
>> result = exp(total(alog(a)))
> ...
>> If some of the elements are negative, you can still handle it. do
>>
>> dum = where(a lt 0, ndum)
>> sig = (-1)^ndum
>> result = sig*exp(total(alog(abs(a))))
>
> You can't honestly be suggesting that this is a good technique? Ignore
> for a momement what happens if any element of 'a' is 0. That code
> performs two transcendental function evaluations per element of 'a'. IDL
> would have to be very badly engineered (which I suppose is possible),
> for a 'for' loop to execute more slowly than your code.

Only one transcendental is computed for each a, alog(). The exp occurs on the
single value after the total. Results for a 10,000 element random floating
array finely tuned to avoid under or overflow:

```
Loop Method:

Average Time:     0.017213961
   0.00528653

Log Method:

Average Time:    0.0049092293
   0.00528580
```

4 times as fast. Suppose you'd like to do an array with 100,000 double elements... you get:

Loop Method:

```
% Loop limit expression too large for loop variable type.
  <LONG     (       99999)>.
```

Log Method:

```
Average Time:      0.050116260
  7.92382e+10
```

And if you hack it with two nested loops to avoid the loop limit error:

```
c=1. & for j=0L,n/100-1 do for k=0L,99L do c=c*a[j*100L+k]
```

you get:

Hacked Loop Method

```
Average Time:       0.30190306
      0.97063262
```

Log Method:

```
Average Time:      0.068175601
      0.97063262
```

A full 5 times faster.

And now, just for fun, the same data set, but with multiplication computed in a heavily optimized C program. The core of the C code is simply the straightforward: "for(i=0;i<N;i++) res*=a[i]"; The result:

```
Got 0.97063262 (Average Time: 0.001710 s)
```

Ouch! Another speedup of by a factor of 40!

Morals: IDL loops are pitifully slow, and you can't loop over very large arrays without trickery, and for many operations, compiled C is *significantly* faster.

JD

--
J.D. Smith |*| WORK: (607) 255-5842

```
Cornell University Dept. of Astronomy  |*|          (607) 255-6263
304 Space Sciences Bldg.               |*|     FAX: (607) 255-5875
Ithaca, NY 14853                       |*|
```