

Re: Matrix Multiplication

Source: <http://sci.tech-archive.net/Archive/sci.math.num-analysis/2008-01/msg00030.html>

- *From:* Gordon Sande <g.sande@xxxxxxxxxxxxxxxxxxxx>
 - *Date:* Fri, 04 Jan 2008 20:04:49 GMT
-

On 2008-01-04 15:20:18 -0400, Evgenii Rudnyi <usenet@xxxxxxxx> said:

Hello Gordon,

Thank you for your comments.

If one were looking at this like a referee one would be questioning an author who seems to like to use a 30 year old version of Fortran. There are three revisions since 1977. Fortran 90 introduced a lot of matrix operations which are never mentioned.

I have written that I know Fortran 77 only and I have mentioned that the newer versions are available. If you are interested why, the answer is simple – I have switched to C++ long ago.

You can find my comments Fortran vs. C++ here

<http://evgenii.rudnyi.ru/doc/misc/fortran.txt>

I see that you have Van Synder's citation of the Les Hatton work on empirical error rates in real programs. C++ does not look good there. Fortran 90 helps find some of the call mismatches that are possible in Fortran 77 if one uses the newer facilities. The subscript errors would remain and are a more serious issue.

Fortran 90 has matrix multiply in the language. Why do you then talk about implementing matrix multiply? Because you choose to make comparisons against the 1977 standard? Perhaps you should also use the C++ version from 30 years ago. If you can not then perhaps that indicates that things do change.

but we do have to agree on this. Taste differs.

Re: Matrix Multiplication

When an author goes out of the way to "think old" one wonders how much of the rest of the material has the same attributes.

I would appreciate if you will be more specific here.

Both a good implementation of an interpreter and of Fortran 90 are free to use the best available subroutine libraries for their purposes. The GNU successor to *G77* may or may not have reached the maturity of using a highly optimized run time for matrix multiply.

Is Intel Fortran good? If you want it to, I will compile the code with it the next week.

As the saying goes "Old Fortran programmers can write Fortran in any language" so using a newer compiler for Fortran 77 changes nothing other than some operating system dependence.

You might want to redo things with Fortran 90 array notation and use the matrix multiply intrinsics as an exercise. If Numerical Recipes can have (and have had for a long time) a Fortran 90 version then one would expect others to meet even that minimal standard.

By the way, *g95* produces exactly the same results as *g77* with this code. If you could donate the code in newer Fortran, please. It would be my pleasure to compile it and compare timing.

Gfortran has the intent of reproducing *G77* on the source that *G77* accepts, subject to exceptions over dropped extensions. So that is hardly very surprising.

There is also the question of the intended audience for an introductory text and whether that audience should be treated to content which is so dated.

I am not sure I understand you. Do you mean that NumPy, C, C++ and ATLAS are outdated?

There is an old saying giving two recipes for swill.

One has a tablespoon of swill and a gallon of fine wine. The other has a gallon of swill and a tablespoon of fine wine. Both produce swill.

Re: Matrix Multiplication

In a fast skim I noticed some comments on memory issues. I did not notice comments on the issues of size scaling when the matrix fits in the cache memory and when it does not. When it does not then is the issue of subscript order that may or may not be cache friendly. The advice that good subroutine packages deal with such issues is good. One would like some indication of the problem they address even if the full technical nature of the solution is not given. Keeping such a discussion both accurate and introductory is not easy.

Well, the goal was learning by doing. One first observe something and then starts thinking why it is. I guess that it is understandable that a matrix of 1000x1000 does not fit in the processor cache.

If you are trying to give advice then mentioning the effects of subscript order on both cache and page hits as sizes grow would seem to reflect the modern world. Time the filling a large array in random order, as some problems might suggest, against the same content in a sequential order.

Best wishes,

Evgenii
<http://MatrixProgramming.com>