Re: Who uses clapack?

Source: http://sci.tech−archive.net/Archive/sci.math.num−analysis/2004−12/0341.html

---

From: Jentje Goslinga (*goslinga_at_telus.net*)
Date: 12/11/04

Date: Sat, 11 Dec 2004 06:27:37 GMT

Ron Shepard wrote:
> *In article <41BA43C3.3040502@telus.net>,*
> *Jentje Goslinga <goslinga@telus.net> wrote:*
> *>*
> *>*
> *>>>The issue here is scaling the elements of a vector by a scalar. I*
> *>>>don't think it matters if that scalar is stored as a 64−bit or as an*
> *>>>extended 80−bit result, the result, the vector of 64−bit values,*
> *>>>will be the same. In other words, those 16−bits are not used (and*
> *>>>cannot be used) in either case.*
> *>>*
> *>>Not quite so, there are four operations: (1) pulling each*
> *>>element of the array through the processor, squaring and*
> *>>adding the result to the accumulator, (2) taking the square*
> *>>root of the accumulator, (3) dividing the result into unity*
> *>>and (4) loading the scale factor and pulling each element*
> *>>of the vector through the processor once more and scaling*
> *>>it. Use of traditional BLAS interposes a mandatory memory save*
> *>>between (2) and (3)*
> *>*
> *>*
> *> First, the dnrm2() function is not required to store the internal*
> *> result of the accumulation in 64−bit form. It is all internal to*
> *> the dnrm2() function with no external storage required for the*
> *> intermediate square root. Your dnrm2() function may or may not do*
> *> the sqrt() in extended precision, but it is not forced to truncate*
> *> by the design of the BLAS.*
> *>*
> *> Even the reference fortran version of the dnrm2() function does not*
> *> do a simple dot product to compute the vector norm, it accumulates*
> *> into several scalar values based on the magnitude of the individual*
> *> elements. The goal of this approach is to reduce the effect of*
> *> roundoff error of the additions.*
> *>*