

RC 22599 October 22, 2002
Computer Science/Mathematics

IBM Research Report

TWBLAS: A Flexible Interface to some Multithreaded BLAS

Version 1.0


<http://www.cs.umn.edu/~agupta/wsmg>

Anshul Gupta

IBM Research Division
T. J. Watson Research Center
P. O. Box 218
Yorktown Heights, NY 10598
anshul@watson.ibm.com

LIMITED DISTRIBUTION NOTICE

This report has been submitted for publication outside of IBM and will probably be copyrighted if accepted for publication. It has been issued as a Research Report for early dissemination of its contents. In view of the transfer of copyright to the outside publisher, its distribution outside of IBM prior to publication should be limited to peer communications and specific requests. After outside publication, requests should be filled only by reprints or legally obtained copies of the article (e.g., payment of royalties).

 **Research Division**
Almaden · Austin · China · Delhi · Haifa · Tokyo · Watson · Zurich

TWBLAS: A Flexible Interface to some Multithreaded BLAS

Version 1.0*

Anshul Gupta

IBM T. J. Watson Research Center

P. O. Box 218

Yorktown Heights, NY 10598

anshul@watson.ibm.com

IBM Research Report RC 22599

October 22, 2002

1 Introduction

The TWBLAS library is a set of selected level-2 and level-3 double precision BLAS with a flexible interface suitable for developing multithreaded applications. This document assumes that the reader has access to the ESSL and LAPACK manuals and is familiar with the original routines for which the multithreaded variants are provided. The following six computational routines are available:

1. TWGEMM
2. TWTRSM
3. TWSYRK
4. TWOTRF
5. TWGEMV
6. TWDTRSV

Each routine has the standard LAPACK calling sequence, but has two additional integer input parameters at the end. These are NUMTHREADS and PRIORITY. NUMTHREADS instructs the routine to execute

*Since this document describes software that might be upgraded from time to time, *TWBLAS* users and the readers of this report are encouraged to download the latest versions by following the link at <http://www.cs.umn.edu/~agupta/wsmv>. The software and the documentation were last updated on October 22, 2002

in parallel using NUMTHREADS threads. It must be an integer greater than 0. The input PRIORITY is useful when the application is using nested parallelism and multiple threads call the multithreaded BLAS routines independently and asynchronously. For example, if one thread makes a call to TWGEMM with NUMTHREADS = 2 and another makes a call to TWTRSM with NUMTHREADS = 4, and the hardware has only 4 CPUs, then the call with the higher value of PRIORITY will get a priority access to the CPUs. PRIORITY must be an integer between 0 and 1024.

The routines can be called from Fortran or C programs. If calling from a C program, all arguments, including the last two, must be passed by reference.

2 Initialization and Finalization

In addition to the computational routines, the TWBLAS library contains two very important routines: TWBLAS_INIT(NCPUS) and TWBLAS_FINALIZE().

TWBLAS_INIT, with an integer input parameter NCPUS must be called once by the main thread before any calls to the computational routines. NCPUS, if set to an integer greater than 0, indicates to the TWBLAS library the total number of CPUs that are available for use by the program. If NCPUS is set to 0, then TWBLAS_INIT picks up the number of CPUs to use from the hardware and returns this value as output in NCPUS.

The use of TWBLAS_FINALIZE is optional. This routine deallocates the resources used by the TWBLAS library and must be called only once by the main thread after the last call to any computational routine.

3 Compiling, Linking, and Running

The library is distributed as the file *libtwblas.a* and `-ltwblas -lessl` must be used during linking to create the executable. A 64-bit version of TWBLAS is available as *libtwblas64.a*. The ESSL library is required because TWBLAS routines need access to serial BLAS. Note that the WSMP sparse solver library and the ESSLSMP libraries should not be used in conjunction with the TWBLAS library.

For example, the following program segment measures speedup of parallel matrix multiplication on a 4-CPU machine.

```
call twblas_init(4) ! use 4 CPUs
call dtrca(c,ldc,d,ldd,m,n,1.d0) ! make a copy of result matrix C
call flush_cache ! a user routine to flush cache to get fair timings
t1 = rtc()
call dgemm('N','N',m,n,k,alpha,a,lda,b,ldb,beta,c,ldc)
t1 = rtc() - t1
call flush_cache
t2 = rtc()
call twgemm('N','N',m,n,k,alpha,a,lda,b,ldb,beta,c,ldc,4,1)
t2 = rtc() - t2
print *, 'Speedup on 4 CPUs = ', t1/t2
```