Workshop on Batched, Reproducible, and Reduced Precision BLAS

May 18th - 19th, 2016

Reference implementation and testing of batched BLAS routines

The University of Manchester

Innovative Computing Laboratory

University of Tennessee

and NLAFET

Motivations

Reference implementation

- Demonstrate that the specification is implementable
- Help to improve the specification
- Provide a reliable code to validate other implementations

Testing

- Valid calling sequences
- Correctness of the computed results

Remarks

The reference implementation does not aim to be efficient

This presentation focuses on Level 3 batched BLAS routines

Outline

1. Reference implementation
2. Accuracy testing
3. Experimental results
4. Concluding remarks

Outline

1. Reference implementation
2. Accuracy testing
3. Experimental results
4. Concluding remarks

Overview of the reference implementation

1. Calling sequences with respect to the specification

$$
\begin{aligned}
\texttt{dgemm\_batch}(&\mathit{const\ enum} * transA,\ \mathit{const\ enum} * transB,\ \mathit{const\ int} * m,\\
&\mathit{const\ int} * n,\ \mathit{const\ int} * k,\ \mathit{const\ double} * alpha,\\
&\mathit{const\ double} * \mathit{const\ double} * arrayA,\ \mathit{const\ int} * lda,\\
&\mathit{const\ double} * \mathit{const\ double} * arrayB,\ \mathit{const\ int} * ldb,\\
&\mathit{const\ double} * beta,\ \mathit{double} * *arrayC,\ \mathit{const\ int} * ldc,\\
&\mathit{const\ int}\ batch\_count,\ \mathit{const\ enum}\ batch\_opts,\ \mathit{int} * info)
\end{aligned}
$$

- **batch_count**: style for the batched (**BATCH_FIXED, BATCH_VARIABLE**)
- **batch_opts**: number of sub-problems to be processed
- **info**: error handling array

Overview of the reference implementation

1. Calling sequences with respect to the specification
2. Arguments checking and error handling

Critical arguments

- Fixed size case: exit if any argument is incorrect
- Variable size case:
-
  - Exit for whole batch error (e.g. batch_count)
  - Continue for sub-problem error (e.g. matrix size)
- **Xerbla** for error handling
- Flexible error handling strategy

Overview of the reference implementation

1. Calling sequences with respect to the specification
2. Arguments checking and error handling
3. Call **reference** BLAS implementation to solve each sub-problem

```
for (iter = 0; iter < batch_count; iter++)
{
  Call a reference BLAS routine
}
```

According to the specification, the reference implementation should provide results as accurate as reference BLAS routines

Outline

Accuracy testing

Basic idea

- The quantity of interest is the relative error $err = \frac{\|C-\hat{C}\|}{\|C\|}$
- Typically we want $err \leq \tau\varepsilon$ where τ is some tolerance and ε is the machine epsilon e.g. $\varepsilon = 10^{-16}$

Remarks

- The tolerance τ is selected independently for each problem, e.g. $\tau = 10, 30, 100$ in MAGMA/LAPACK etc
- The choice of $\tau = 10, 30 \ldots$ may not be very rigorous
- Investigate a rigorous approach for small BLAS problems?

Accuracy testing

Error analysis approach

- Perform error analysis of BLAS routines [Nicholas J. Higham, 2002]
- Use the **forward/backward error bound** for accuracy testing
- $\|Forward\_Error\| \leq \|Forward\_Error\_Bound\|$ or $\|Residual\| \leq \|Backward\_Error\_Bound\|$

Remarks

- No need of intensive experiments to set a tolerance τ
- A rigorous error bound can seem pessimistic in general but meaningful for small BLAS problems

Accuracy testing

Forward error bound of GEMM

- Formula: $C = \alpha AB + \beta C_{init}$
- Forward error $\|C - \hat{C}\|_\infty$
- Forward error bound: $(N|\alpha|\|A\|_\infty\|B\|_\infty + |\beta|\|C_{init}\|_\infty)\varepsilon$

Remark

The computed solution $\hat{C}$ is acceptable if

$$\frac{\|C-\hat{C}\|_\infty}{(N|\alpha|\|A\|_\infty\|B\|_\infty+|\beta|\|C_{init}\|_\infty)\varepsilon} \leq 1$$

Accuracy testing

Backward error analysis of TRSM [Nicholas J. Higham, 2002]

Theorem: *Let the triangular systems $Tx = b$ where $T \in \mathbb{R}^{n\times n}$ is nonsingular, be solved by substitution with any ordering. Then the computed solution $\hat{x}$ satisfies $(T + \Delta T)\hat{x} = b$, $|\Delta T| \leq \gamma_n |T|$, where $\gamma_n = \frac{n\varepsilon}{1-n\varepsilon} \approx n\varepsilon$.*

Backward error bound of TRSM

- Formula: $AX = \alpha B$
- Backward error bound: $N\|A\|_\infty \|X\|_\infty \varepsilon$
- Note: TRSM's backward error bound does depend on B
- The solution $\hat{X}$ is acceptable if $\frac{\|A\hat{X} - \alpha B\|_\infty}{N\|A\|_\infty \|\hat{X}\|_\infty \varepsilon} \leq 1$

Accuracy testing: BLAS3 error bound summary

Routine	Formula	Error bound based success criteria
GEMM SYMM HEMM	$C = \alpha AB + \beta C_{init}$	$\frac{\|C - \hat{C}\|_\infty}{(N\|\alpha\|\|A\|_\infty \|B\|_\infty + \|\beta\|\|C_{init}\|_\infty)\varepsilon} \leq 1$
SYRK HERK	$C = \alpha AA^{T/H} + \beta C_{init}$	$\frac{\|C - \hat{C}\|_\infty}{(N\|\alpha\|\|A\|_\infty \|A\|_1 + \|\beta\|\|C_{init}\|_\infty)\varepsilon} \leq 1$
SYR2K HER2K	$C = \alpha AB^{T/H} + \alpha BA^{T/H} + \beta C_{init}$	$\frac{\|C - \hat{C}\|_\infty}{(N\|\alpha\|(\|A\|_\infty \|B\|_1 + \|B\|_\infty \|A\|_1) + \|\beta\|\|C_{init}\|_\infty)\varepsilon} \leq 1$
TRSM	$AX = \alpha B$	$\frac{\|AX - \alpha B\|_\infty}{N\|A\|_\infty \|X\|_\infty \varepsilon} \leq 1$

Accuracy testing: some remarks on the forward error bound

$$\frac{\|C - \hat{C}\|_\infty}{(N|\alpha|\|A\|_\infty\|B\|_\infty + |\beta|\|C_{init}\|_\infty)\varepsilon} \leq 1\,?$$

In our accuracy checking, we assume that the correct solution C of GEMM exists, what is not true in real-world applications.

$$\frac{\|C - \hat{C}\|_\infty}{(N|\alpha|\|A\|_\infty\|B\|_\infty + |\beta|\|C_{init}\|_\infty)\varepsilon} \leq 2\,?$$

- Our exact solutions: reference BLAS
- Problem: reference BLAS results have the same error bound
- Pessimistic approach: authorise twice the error bound

Outline

Some implementation details

Batched BLAS implementations tested

- **Batched_cuBLAS**: GEMM and TRSM
- **Batched_MAGMA**: GEMM, HERK, SYRK and TRSM
- **Batched_MKL**: only GEMM

Alternative to Batched on multi-core processor?

- **OMP_LOOP_MKL**: OpenMP loop over sequential MKL
- **PARALLEL_MKL**: single loop over parallel MKL

Additional details

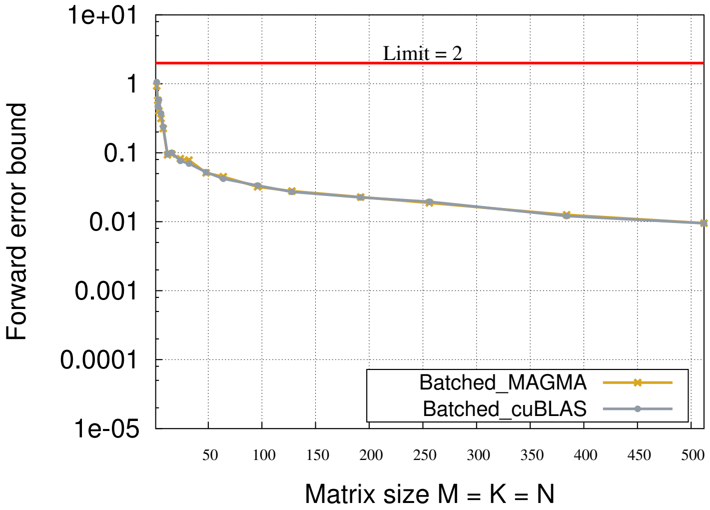
About the source code

- Programming language: C
- Precision: single, double, single complex and double complex
- Documentation: well documented (doxygen)
- First release coming soon

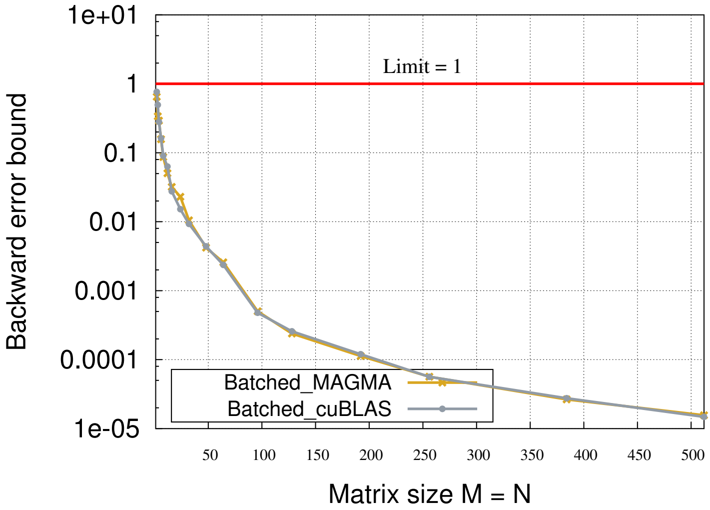
Hardware

- CPU: 2x10 Intel Xeon E5-2650 v3, 25M Cache, 2.30 GHz
- GPU: Nvidia K40c GPU with 2, 880 CUDA cores

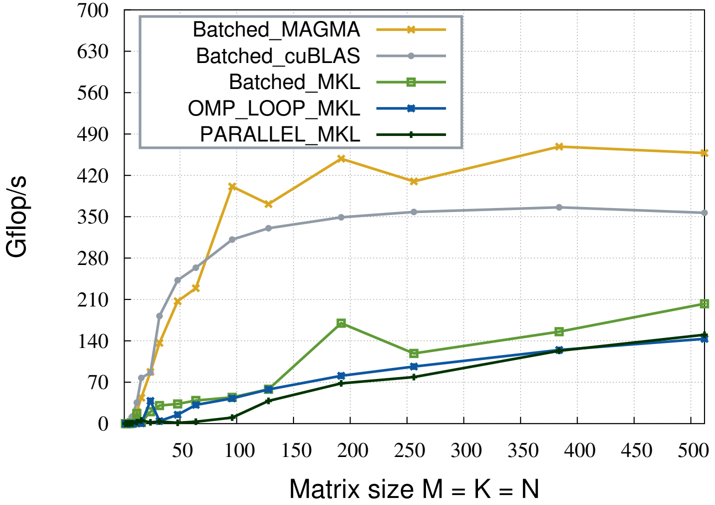
Accuracy of **1K** batched **DGEMM**



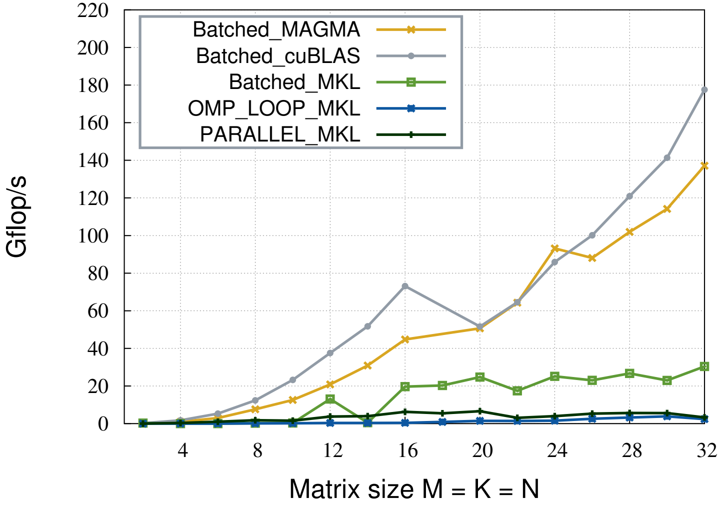
Accuracy of **1K** batched **DTRSM**



Performance of 1K batched DGEMM



Performance of **1K** batched **DGEMM**



Outline

1. Reference implementation
2. Accuracy testing
3. Experimental results
4. Concluding remarks

Concluding remarks and perspectives

Conclusion remarks

- Investigation of new accuracy checking
- Competitive performance cuBLAS/MAGMA
- MKL batched BLAS seems promising

Future work

- First release with respect to the final specification
- Extension to level 1 & 2 BLAS
- Provide fortran interface

Concluding remarks and perspectives

Conclusion remarks

- Investigation of new accuracy checking
- Competitive performance cuBLAS/MAGMA
- MKL batched BLAS seems promising

Future work

- First release with respect to the final specification
- Extension to level 1 & 2 BLAS
- Provide fortran interface