Workshop on Batched, Reproducible, and Reduced Precision BLAS.
Klaus Advanced Computing Building
Georgia Tech, Atlanta
February 24th - 25th, 2017

Workshop website: http://bit.ly/Batch-BLAS-2017

Sven Hammarling

School of Mathematics
The University of Manchester
Alan Turing Building
Manchester, M13 9PL, UK

sven@ma.man.ac.uk

April 13, 2017

1 Introduction

This was the second workshop intended to present and discuss proposals for a set of batched BLAS, a set of reproducible BLAS and a set of reduced, extended and mixed precision BLAS. The aim is to define a standard interface for each of these sets of BLAS. Information on the first workshop can be found at http://bit.ly/Batch-BLAS-2016.

The event also included a reception on the evening of the 23rd February and a dinner on the 24th February, both kindly sponsored by Intel.

The workshop was opened by Jack Dongarra with a welcome and an introduction to the workshop, as well as thanks to Pradeep Dubey, Jason Reidy and Anna Stroup of GATech and Jack's team at UCL for their help and organization, and to Intel for their financial support. Jack also reminded the attendees of an earlier community effort, the BLAS Technical Forum standard[1], published as a technical report on August 21, 2001, that presented a number of extensions to the BLAS, including sparse BLAS and extended and mixed precision BLAS. That was followed by each of the participants introducing themselves.

Jack's introduction was followed by a talk "Workshop, 18 May - 19 May 2016" by Sven Hammarling, which was intended to briefly remind or inform participants about the previous workshop.

Links to the talks can be found in the agenda for the above workshop web page, which also has links to further information on the Batched BLAS and the Reproducible BLAS. In addition there is a link to a draft paper on "A Proposal for a Next-Generation BLAS"[2], which was the subject Jim Demmel's talk at the workshop.

This report mainly highlights, briefly, those parts of the presentations that are immediately relevant to the purpose and aims of the workshop. A number of other interesting and important topics were raised, please see the presentations for the detail.

In the main and for the sake of brevity, only the discussions related to the proposals have been included. The discussion of the presentations does not necessarily reflect the order in which they were given.

2 The Batched BLAS

There were seven presentations directly related to the Batched BLAS (BBLAS). In particular, there were presentations discussing proposed specifications for a set of Batched BLAS routines. Jim Demmel's talk, mentioned above, is included in Section 5.

The Batched BLAS are intended for HPC applications where a large number of the same small matrix operations, such as matrix multiplication, can be applied simultaneously.

[1] http://www.netlib.org/blas/blast-forum/

[2] A Proposal for a Next-Generation BLAS, J. Demmel, G. Henry, X.S. Li, J. Riedy, P.T.P. Tang, February 16, 2017

2.1 BBLAS APIs and Memory Layouts

Sam Relton, University of Manchester

This presentation was principally concerned with the specification, or API, of the Batched BLAS, as well as the need to standardize the memory layout once an API has been chosen. The following APIs were discussed:

1. an API with a flag, *batch_type* having the options BATCH_VARIABLE and BATCH_FIXED, to denote whether the batches all have fixed size, or if the batches are of variable size;
2. separate functions for BATCH_VARIABLE and BATCH_FIXED;
3. an API based on groups, where a group has batches of fixed size.

The following data layout options were considered:

1. pointer to pointer (P2P);
2. strided;
3. interleaved, for fixed batch size.

There was considerable discussion about the proposed API. Some of the issues raised included:

- the aliasing rules;
- concerns about the BATCH_VARIABLE option for *batch_type*, such as when to use that rather than several calls with BATCH_FIXED, the amount of data needed when *batch_count* is large;
- the effect of a large value of *batch_count*;
- the number of cases that need to be implemented.

Other discussion centred on the notion of locality and if some utilities, such as the handle in CUBLAS, could be provided to help with the issues.

There was also discussion on the data layout issues, particularly on the interleaved format and whether, or not efficiency overrides complication for users.

2.2 Autotuning Batched Kernels

Jacob Kurzak, ICL, UTK

This presentation first looked at the Cholesky factorization as a case study using the batched BLAS and an interleaved layout. Algorithmic variants and various tuning parameters were considered. The presentation then described the bench testing environment for automated software testing, BEAST[3] and the associated software autotuning infrastructure, BONSAI.

There was discussion about the tuning parameters considered in Cholesky. Some concerns were expressed about the lack of checking for positive definiteness since no synchronization takes place.

Following a question about pruning the search space in BONSAI, Jacob said that there is currently no discovery mechanism, but hopes that pruning will be considered in the future.

2.3 A Proposed Modification to the Batch BLAS Interface

Ahmad Abdelfattah, ICL, UTK

In this presentation a few suggested modifications to the BBLAS API were made, particularly based on experience with MAGMA[4] examples from the LU factorization were given to illustrate the ideas.

2.4 MAGMA Batched Computations: Approaches and Applications

Stan Tomov, ICL, UTK

In this presentation the BBLAS and batched LAPACK routines currently covered by MAGMA were described, as well as the interfaces of the routines and the design and optimization strategies used. A number of applications for which batch computations are appropriate, were also discussed.

2.5 Compact Batched BLAS

Tim Costa, Intel MKL Team

This presentation described the Intel MKL BBLAS, particularly their Compact BBLAS, where the data layout is in the interleaved format and the API is based on groups. As in the first workshop, a nice table comparing various batched GEMMs was given.

2.6 KokkosKernels: Compact Layouts for Batched Blas and Sparse Matrix-Matrix multiply

Siva Rajamanickam, Sandia National Laboratories.

Kokkos[5] is part of the Trilinos[6] project. Kokkos Kernels provides a number of kernel routines, including some sparse and dense linear algebra kernels and in particular some Compact BBLAS. There is ongoing collaboration with the Intel MKL team and the ICL

[3] http://www.icl.utk.edu/research/profile/beast

[4] The ICL dense linear algebra library for GPU and multicore architectures. icl.cs.utk.edu/magma/

[5] https://github.com/kokkos

[6] https://trilinos.org/

MAGMA team. As well as discussion of the BBLAS, the work on a sparse matrix-matrix multiplication routine, SPGEMM was also presented.

3 Reproducibility

Software reproducibility concerns getting bitwise identical results from multiple runs of a program with the same input. This topic is included in the talk by Jim Demmel in Section 5.

4 BLAS for Different Precisions and Integer BLAS

BLAS for both reduced, extended and mixed precision are included in the talk by Jim Demmel in Section 5, as well as a talk by Piotr Luszczek on Half Precision Benchmarks. There was one presentation on a version of GEMM for integer, or quantized matrices.

4.1 Matrix Multiplication with Quantized matrices

Murat Guney, Intel MKL Team

A number of applications, such as speech and face recognition, were mentioned to motivate the need for integer matrix multiplication. An introduction to quantizing with integers and quantized matrices was also presented, as well as the API for the Intel version of IGEMM.

In answer to a question it was said that the quantized representation is not necessarily exact and that an error analysis has not yet been performed. Interaction with Google was mentioned.

5 Related Presentations

As well as Jim Demmel's talk, there were eight other talks presenting work related to the goals of the workshop.

5.1 A Proposal for a Next-Generation BLAS

Jim Demmel, UC Berkeley

The main goal of this presentation was to propose an interface to accommodate the current and future needs of the BLAS, including the standard BLAS, the mixed and new precision BLAS, the BBLAS and the reproducible BLAS. The API would be a wrapper around the existing BLAS when the semantics match. As mentioned in Section 1 there is a draft paper on the workshop website that gives a fuller description of the proposal.

Although the proposal covers a large number of routines, it is not proposed that everything necessarily be implemented, or optimized.

A naming scheme to cover all the routines was proposed which, in particular, would allow overloading and thus higher level languages to infer which routine is needed.

It was proposed to deprecate the BLAS error handling routine and to include an LAPACK style INFO argument.

An algorithm for reproducibility was described.

Although there was discussion about the extent of the proposal and the naming scheme, there did not appear to be serious dissent to the proposal.

5.2 Half Precision Benchmarks

Piotr Luszczek, ICL, UTK

A number of applications that use 16 bit floating point arithmetic for speed were mentioned and a fast mixed precision linear equation solver, using iterative refinement was described, together with benchmark results.

5.3 High Performance Design of Batched Tensor Computations: Performance Analysis, Modelling, Tuning and Optimization

Azzam Haidar, ICL, UTK

This presentation talked about a number of issues that arise in considering batched tensor computations, particularly in the development of a MAGMA kernel for deep learning and tensor contraction. Topics such as reproducibilty and reliability and data layout were discussed.

5.4 The Landscape of High-Performance Tensor Contractions

Paul Springer, Aachen Institute for Advanced Study in Computational Engineering Science, Aachen University

This presentation discussed three approaches for tensor contractions; batch GEMM, flatten into large matrices, or perform transposes implicitly during data movement. It was noted that the performance of GEMM can vary wildly when in a memory bound situation.

Computational chemistry and deep learning were mentioned in answer to the question of what are the applications.

5.5 Tensor Contractions with Extended BLAS Kernels on CPU and GPU

Cris Cecka, Nvidia

In this presentation tensor contractions were explained, and applications such as machine learning, deep learning and distributed FFTs were mentioned.

The use of the BBLAS for tensor contractions, especially a strided BBLAS, was described.

5.6 Batched Factorization and Inversion Routines for Block-Jacobi Preconditioning on GPUs

Hartwig Anzt, ICL, UTK

This presentation looked at using a block-Jacobi preconditioner based on diagonal scaling in conjunction with an iterative Krylov method for block structured sparse matrices, such as those arising in finite element methods. The preconditioner setup typically requires the inversion of many small diagonal blocks, so batched inversion of many small systems. It was suggested that Gauss-Jordan elimination be used for the inversion. A batched inversion routine was requested.

There was some discussion on the accuracy of Gauss-Jordan. Nick Higham suggested doing an approximate inverse.

5.7 Exploiting Batched Operation in Applications

Hatem Ltaief, Extreme Computing Research Centre, KAUST

This presentation showed a number of applications that lead to the need to solve large covariance matrix problems. The proposed solution methods require the Cholesky factorization of many, often low rank, matrices. The KAUST KBLAS, and the HBLAS for batched operations, were described.

There was discussion about the effect of missing data on positive definiteness, and the preservation of positive definiteness of the original matrix.

5.8 Autotuning Dense Batched QR Factorizations on GPU

Wissam Sid-Lakhdar, Texas A&M University

This presentation considered the factorization of a sparse matrix using the multifrontal QR method on a shared memory computer with one, or more, GPUs. The proposed method requires a batched dense QR solver and batched GEMM. Since it is hard to write efficient portable code for GPUs, it was proposed to write highly parameterised code and use autotuning. The approach to autotuning as described, with especial attention to the optimization problem.

5.9 NLAFET: Overview and Status Report

Bo Kågström,Umeå University

This presentation described the project NLAFET[7] funded under the European Commission's program Horizon 2020, the goals of which have much in common with those of this workshop. An update on the progress since the previous workshop was given, together with some of the results obtained so far.

[7] Numerical Linear Algebra for Future and Emerging Technologies. www.nlafet.eu/

There was some discussion about on-line autotuning concerning speed and suitable optimization methods.

6 Vendor Presentations

In this session seven vendors gave presentations, but the presentation by Tim Costa of Intel is described in Section 2 and the presentation by Cris Cecka of Nvidea is in Section 5.

6.1 ARM Performance Libraries – Current and future interests

Chris Goodyer, ARM

This talk introduced the ARM performance libraries and discussed some of their plans and hopes that were relevant to the workshop. For example, ARM will include the BBLAS once an API has been agreed. An update from the previous workshop was given on High Precision Accumulator and High Precision Anchored Numbers.

6.2 NAG Update

Mike Dewar, NAG Ltd

This talk introduced NAG and the NAG Library indicating new and planned functionality. Mention was made of the strong demand for bitwise reproducibility from big finance companies because their code needs to be certified.

6.3 MATLAB and BLAS

Pat Quillen, MathWorks

This presentation described the relationship between the MATLAB and the BLAS, the importance of reproducibility and how MATLAB handles it. The BBLAS are included, but currently little used. There is also appears to be little demand for reduced and mixed precision BLAS.

6.4 BLAS Usage + (Sparse) Tensor Compilation

Shoaib Kamil, Adobe Research

This presentation discussed the importance to Adobe of the BLAS, particularly in the area of neural networks. Requirements for BBLAS, quantized integer and FP16, and mixed-precision were mentioned. The development of a Tensor Algebra Compiler, to automatically design kernels for specific tensor contractions, was described.

7 Wrap Up

Jack Dongarra, ICL, UTK

Jack Dongarra closed the workshop by briefly discussing what might happen next. It is intended to have a report on the workshop (this report) and a follow on workshop, perhaps in spring 2018. It is hoped to produce a revised proposal for the BBLAS in the coming months. A half day meeting at a meeting such as SC17 might be a possibility. Jim Demmel suggested having a birds of a feather session at a machine (deep) learning conference.

The slides presented at the workshop will be on the workshop website and it is planned to use the website for interaction, perhaps combining the two workshop web pages.

The thanks given at the opening to Pradeep Dubey, Jason Reidy and Anna Stroup of GATech and Jack's team at UCL for their help and organization, and to Intel for their financial support were reiterated. The meeting also expressed their grateful thanks to Jack Dongarra for an excellent workshop.

8 Brief Summary

The purpose of the workshop was to look further into defining a standard interface for the batched BLAS, reproducible BLAS, and reduced, extended and mixed precision BLAS. Whilst no finality was expected, it seems fair to say that good progress was made.

It was interesting to see the growing number of applications that require these standards.

As a reminder, the presentations can be found at the workshop web page, bit.ly/Batch-BLAS-2017, as well as links to further information on the Batched BLAS and ReproBLAS. The web page for the first workshop is at bit.ly/Batch-BLAS-2016.

Comments on the proposals would be welcome. Please send these, and questions about the proposals presented in this report to Jack Dongarra at dongarra@icl.utk.edu.