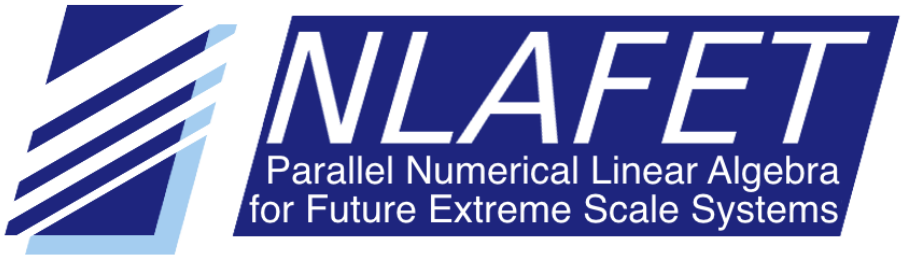


H2020–FETHPC–2014: GA 671633

# D2.3

# Prototype software for Batched BLAS

October 2016

### Document information

| | |
|---|---|
| Scheduled delivery | 2016-10-31 |
| Actual delivery | 2016-10-31 |
| Version | Final |
| Responsible partner | UNIMAN |

### Dissemination level

PU — Public

### Revision history

| Date | Editor | Status | Ver. | Changes |
|---|---|---|---|---|
| 2016-10-21 | Samuel Relton | Draft | 0.1 | Initial version of document produced. |
| 2016-10-31 | Samuel Relton | Final | | Final version based on comments from all partners. |

### Author(s)

Samuel Relton, Pedro Valero-Lara, and Mawussi Zounon, UNIMAN.

### Internal reviewers

Simplice Donfack, INRIA; Florent Lopez, STFC; and Lars Karlsson, UMU.

### Contributors

In addition to the reviewers, the following team members have contributed with comments: Bo Kågström and Björn Adlerborn, UMU.

### Copyright



### Acknowledgements

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under the grant agreement number 671633.

# Table of Contents

**1 Introduction** **3**

**2 Prototype Software for the Batched BLAS** **3**

**3 Related Publications** **4**

# 1 Introduction

The batched BLAS (BBLAS) aims to efficiently compute many small linear algebra problems simultaneously, making efficient use of the user's available hardware. For example, we may wish to compute, in parallel,

$$C_i \leftarrow \alpha_i A_i B_i + \beta_i C_i, \quad \text{for } i = 1 : N,$$

where the matrices $A_i$, $B_i$, and $C_i$ are small but $N$ is large. There are numerous application areas that could benefit from using batched BLAS including computational fluid dynamics, image processing, and deep learning.

One key part of the NLAFET project is to develop a community standard for batched BLAS operations so that implementations by academics and vendors (such as Intel and NVIDIA) are compatible. This allows users and application engineers to use a standard interface for all batched BLAS operations.

The aim of this deliverable is to produce a complete implementation of BBLAS routines, extending all of the standard level-3 BLAS. This forms a foundation for discussion, comments, and constructive criticism which can be improved upon in the next proposed standard. We plan to hold a number of workshops between prominent figures in the linear algebra community, vendors, and application developers in the coming few years to converge on a finalised standard for BBLAS operations.

There are a number of issues to be discussed within the community, which impact both the API design and performance, before a final standard can be obtained. For instance:

- the layout of the matrices in memory,
- the type of batch operations to support (fixed / variable / groups), and,
- the differences between current and emerging architectures.

More detail on these issues can be found in the related publications mentioned at the end of this document.

# 2 Prototype Software for the Batched BLAS

The prototype software forming the bulk of this deliverable can be found at the projects software repository (`https://github.com/NLAFET/BBLAS-ref`).

In our initial implementation we have focused on providing:

- a full extension of the level-3 BLAS routines to support batching,
- an "optimized" implementation using OpenMP for baseline comparison, and,
- a framework for comparing current BBLAS implementations from vendors.

The software is fully documented (with the documentation automatically generated during compilation) and comes with a simple Python script to aid compilation against the CUDA, MAGMA, and MKL libraries.

We hope that this forms a foundation upon which further discussions on the eventual standard can be based. To this end, we are currently organising a workshop to discuss advances in BBLAS to be held in early 2017.

# 3 Related Publications

Whilst exploring the issues related to BBLAS performance, members of the NLAFET consortium have produced a number of related publications.

- *Draft Specification for Batched Basic Linear Algebra Subprograms.* Jack Dongarra, Iain Duff, Mark Gates, Azzam Haidar, Sven Hammarling, Nicholas J. Higham, Jonathon Hogg, Pedro Valero-Lara, Samuel D. Relton, Stanimire Tomov and Mawussi Zounon. NLAFET Deliverable 7.3, April 2016.
- *A Comparison of Potential Interfaces for Batched BLAS Computations.* Samuel D. Relton, Pedro Valero-Lara and Mawussi Zounon. NLAFET Working Note 5.
- *The Design and Performance of Batched BLAS on Modern High-Performance Computing Systems* Samuel D. Relton, Pedro Valero-Lara, Mawussi Zounon, Sven Hammarling, Nicholas J. Higham, and Jack Dongarra. Submitted to IPDPS 2017.