Hierarchical Approach for Deriving a Reproducible LU factorization on GPUs

Roman Iakymchuk, Stef Graillat, David Defour, Enrique Quintana-Ortí

To cite this version:

Roman Iakymchuk, Stef Graillat, David Defour, Enrique Quintana-Ortí. Hierarchical Approach for Deriving a Reproducible LU factorization on GPUs. The Numerical Reproducibility at Exascale (NRE16) workshop held as part of the Supercomputing Conference (SC16), Nov 2016, Salt Lake City, UT, United States. hal-01382645

HAL Id: hal-01382645
https://hal.archives-ouvertes.fr/hal-01382645
Submitted on 17 Oct 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Hierarchical Approach for Deriving a Reproducible LU factorization on GPUs

Roman Iakymchuk¹, Stef Graillat², David Defour³, Enrique S. Quintana-Ortí⁴

¹ KTH Royal Institute of Technology, Stockholm, Sweden, riaakymch@kth.se
² Sorbonne Universités, UPMC Univ Paris 06, Paris, France, stef.graillat@lip6.fr
³ Université de Perpignan, Perpignan, France, david.defour@univ-perp.fr
⁴ Universidad Jaime I, Castellón, Spain, quintana@uji.es

Abstract. We propose a reproducible variant of the unblocked LU factorization for graphics processor units (GPUs). For this purpose, we provide Level-1/2 BLAS kernels that deliver correctly-rounded and reproducible results for the dot (inner) product, vector scaling, and the matrix-vector product. In addition, we draw a strategy to enhance the accuracy of the triangular solve via inexpensive iterative refinement. Following a bottom-up approach, we finally construct a reproducible implementation of the LU factorization for GPUs, which can easily accommodate partial pivoting for stability and be eventually integrated into a (blocked) high performance and stable algorithm for the LU factorization.

Key words: LU factorization, BLAS, reproducibility, accuracy, long accumulator, error-free transformation, GPUs.

The solution of a linear system of equations is often at the core of many scientific applications. Usually, this process relies upon the LU factorization, which is also the most compute-intensive part of it. Although there exist implementations of this factorization that deliver high performance on a variety of processor architectures, their reproducibility and, even more, accuracy cannot be guaranteed. This problem is mainly due to the non-associativity of floating-point operations, combined the concurrent thread-level execution of independent operations on conventional multicore processors or the nondeterminism of warp scheduling on graphics processing units (GPUs).

In this work, we address the problem of reproducibility of the LU factorization on GPUs due to cancellations and rounding errors associated with floating-point arithmetic. Instead of developing a GPU implementation of the LU factorization from scratch, we benefit from the hierarchical and modular structure of linear algebra libraries, and start by developing and augmenting reproducible algorithmic variants for the BLAS (Basic Linear Algebra Subprograms) kernels that serve as building blocks in the LU factorization. In addition, we enhance the accuracy (in case of non-correctly-rounded results) of these underlying BLAS routines.

We consider the left-looking algorithmic variant of the unblocked LU factorization (also known as jik or jki variant), which is especially appealing for fault tolerance.

¹ We define reproducibility as the ability to obtain a bit-wise identical floating-point results from multiple runs of the code on the same input data.
² By accuracy, we mean the relative error between the exact result and the computed result.
out-of-core computing, and the solution of linear systems when the coefficient matrix
does not fit into the GPU memory. This specific variant can be formulated in terms
of the Level-1 and Level-2 BLAS kernels for the dot product (DOT), vector scaling
(SCAL), matrix-vector product (GEMV), and triangular system solve (TRSV). We prevent
cancellations and rounding errors in these kernels by applying the following techniques:

- We leverage a long accumulator and error-free transformations (EFT) designed for
  the exact, i.e. reproducible and correctly-rounded, parallel reduction (EXSUM) [1] in
  order to derive an exact DOT. For this purpose, we extend the multi-level parallel
  reduction algorithm and apply the traditional EFT, called TwoProduct [4], to the
  multiplication of two floating-point numbers.
- By its nature, SCAL is both reproducible and correctly-rounded. However, in the
  considered unblocked LU factorization, SCAL multiplies a vector by an inverse of a
  diagonal element, which causes two rounding-off errors. For that reason, we provide
  an extension to SCAL (INVSCAL) that performs the division directly, which ensures
  the exact results.
- We develop an accurate and reproducible algorithm for GEMV by combining together
  a high performance GPU implementation of this operation with the exact DOT.
- To improve the parallel performance of TRSV, we use a blocked variant that relies
  upon small TRSV involving the diagonal blocks and rectangular GEMV with the off-
  diagonal blocks. This approach leads to a reproducible, but not yet correctly-rounded,
  triangular solve (EXTRSV) [3]. We tackle the accuracy problem by applying a few
  iterations of inexpensive iterative refinement.

In addition, we integrate partial pivoting [2] into the left-looking variant of the un-
blocked LU factorization which, as part of future work, will allow us to employ this
unblocked LU factorization as a building block in the development of high performance
blocked algorithms.

In summary, following the bottom-up approach, we construct a reproducible al-
gorithmic variant of the unblocked LU factorization, presenting strong evidence that
reproducible higher-level linear algebra operations can be constructed following a hier-
archical bottom-up approach.

References

1. Sylvain Collange, David Defour, Stef Graillat, and Roman Iakymchuk. Numerical re-
producibility for the parallel reduction on multi- and many-core architectures. Parallel
3. Roman Iakymchuk, Sylvain Collange, David Defour, and Stef Graillat. Reproducible tri-
angular solvers for high-performance computing. In Proceedings of the 12th International
Conference on Information Technology: New Generations (ITNG 2015), Special track on:
Wavelets and Validated Numerics, April 13-15, 2015, Las Vegas, Nevada, USA, pages 353–
358, February 2015. HAL: hal-01116588v2.
5. James M. Ortega. The ijk forms of factorization methods I. Vector computers. Parallel