terry.cojean@kit.edu, yu-hsiang.tsai@kit.edu

**MS23**

**Using the PETSc/TAO ADMM Methods on GPUs**

The Alternating Direction Method of Multipliers (ADMM) is an algorithm for structured numerical optimization problems that solves a sequence of subproblems that update the variables and the multipliers. The variables can typically be further subdivided and updated in parallel. In this talk, we explore using the ADMM code in the Toolkit for Advanced Optimization for solving structured regression problems on GPUs and provide some initial performance results.

Hansol David Suh
Georgia Tech
hsuh7@gatech.edu

Alp Dener
Argonne National Laboratory
Mathematics and Computer Science Division
adener@anl.gov

Tobin Isaac
Georgia Tech
tisaac@cc.gatech.edu

Todd Munson
Argonne National Laboratory
Mathematics and Computer Science Division
tmunson@mcs.anl.gov

**MS23**

**Vendor-Optimized vs. Portable Performance: Approaches to Get Both**

Scientific software should run at maximum performance on any hardware. Reaching this noble ideal can be simplified by reusing well-tuned low-level routines provided by hardware vendors. However, these vendor libraries only provide good performance on the vendor's hardware. As such, scientific software has to either interface different vendor libraries, or provide portable performance by itself. If no vendor-tuned implementations of a particular functionality exist, a strategy for achieving portable performance is required anyway. On the one hand, this talk surveys available strategies and experience on providing portable performance. On the other hand, a software library approach is presented that acts as an intermediate wrapper to provide a single, unified interface for several vendor-optimized linear algebra libraries. This resolves the burden of interfacing many different vendor libraries; instead, only the intermediate wrapper needs to be interfaced, while preserving the full performance benefit of vendor libraries tuned for the particular hardware available.

Karl Rupp
TU Wien
me@karlrupp.net

**MS23**

**Optimization of SpMV on GPU for Iterative Solvers in PETSc**

Large sparse systems resulted from the discretization of partial differential equations often need to be solved by iterative linear solvers due to scalability and memory constraints. Typically the computation of sparse matrix-vector (SpMV) products is the workhorse of iterative linear solvers. There are several optimized general-purpose SpMV implementations available, such as Intel MKL, CUSPARSE, CUSP, and ViennaCL. However, to use these libraries, explicit conversion between different matrix format may be required, which hampers the performance. Thus it is desirable and beneficial to have a native implementation of SpMV in the solver that can be optimized to exploit the characteristics of a hardware platform (e.g. memory bandwidth, thread-level parallelism) as well as the characteristics of the application problem (e.g. the block structure, sparsity pattern). In this talk, I will present our recent work on optimizing the Sliced Ellpack (SELL) matrix class in PETSc on NVIDIA GPUs. SELL was previously designed for multicore CPUs with long SIMD and tailored to the needs of PETSc iterative solvers and preconditioners. I will show how we extend SELL for better vectorization on GPU and compare the performance of SELL with other options supported by PETSc (ViennaCL and CUSPARSE).

Hong Zhang
Argonne National Laboratory
hongzhang@anl.gov

**MS24**

**Efficient Sparse Triangular Solve**

In the context of the solution of sparse system of equations using direct methods, many efforts have been dedicated to improve the performance of the factorization. However the triangular solve still suffers from a lack of optimisation. It is especially true when looking at preconditioned iterative methods such as the Preconditioned Conjugate Gradient. The application of the Block Jacobi preconditioner is performed at each iteration whereas the factorization of the diagonal block is done once. It often results a time to solve led by the application of the preconditioner. This phasis becomes critical when considering a large number of right-hand sides. In this talk, we present the parallelization of the triangular solve phasis using a task based programming model in a the sparse Chokesly solver called SpLLT [Duff, Hogg, and Lopez. Numerical Algebra, Control and Optimization. Volume 8, 235-237, 2018]. Using OpenMP runtime as well as BLAS3 operations, we show that this approach can outperform state-of-the-art solvers such as MKL Pardiso. We also interface SpLLT with Enlarged Conjugate Gradient [Grigori, and Tissot. Research Report RR-9023, Inria] where the number of directions of research is typically 10.

Sebastien Cayrols, Florent Lopez
University of Tennessee, Knoxville
sebastien.cayrols@gmail.com, flopez@icl.utk.edu

Iain Duff
Science & Technology Facilities Council, UK
and CERFACS, Toulouse, France
iain.duff@stfc.ac.uk

**MS24**

**Parallel Direct Matrix Factorization Methods for Dynamic Optimization**

Dynamic optimization problems are a special class of optimization problems that involve decision or control variables, and environment or state variables which vary with time. Their goal is to minimize/maximize some cost func-