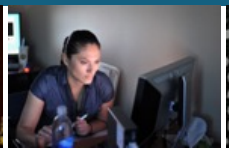
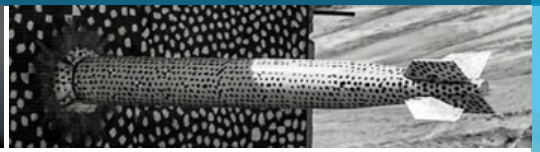
Mixed Precision in Trilinos





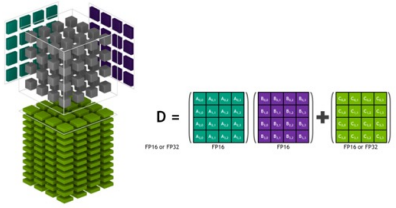
Jennifer Loe, Siva Rajamanickam

SAND2021-15114C

Mixed Precision Motivation

Mixed Precision is here to stay

- **Primary driver**: AI/ML workloads at data centers drive new hardware features
- **Secondary driver**: Reduced power, data movement costs, new floating point formats (bfloat16, TF32 etc)
- This is very important for SNL and DOE to be part of
  - Actively investigating a variety of dataflow architectures as an accelerator option for forthcoming NNSA system procurements
    - Already systems like Lassen connected to a Cerebras accelerator
  - Mistakes here can be costly, life-threatening (Past mistakes: See "Round off error and Patriot Missile Failure")

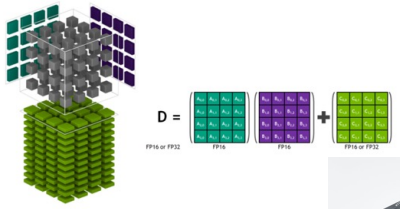


NVIDIA Sparse Tensor Core

TRILINOS

Building Blocks for Mixed-Precision

- Sandia is very active in this emerging field
  - xSDK multi-precision project
    - Algorithm and Trilinos focus
  - ARIAA Co-design center
    - Architecture focus
  - Half precision support in Kokkos Core and Kokkos Kernels (ECP Sake project)
    - Programming models and linear algebra focus



NVIDIA Sparse Tensor Core

"A survey of numerical methods utilizing mixed precision arithmetic." arXiv preprint arXiv:2007.06674 (2020). **From a large DOE multi-precision effort, several SNL authors.**

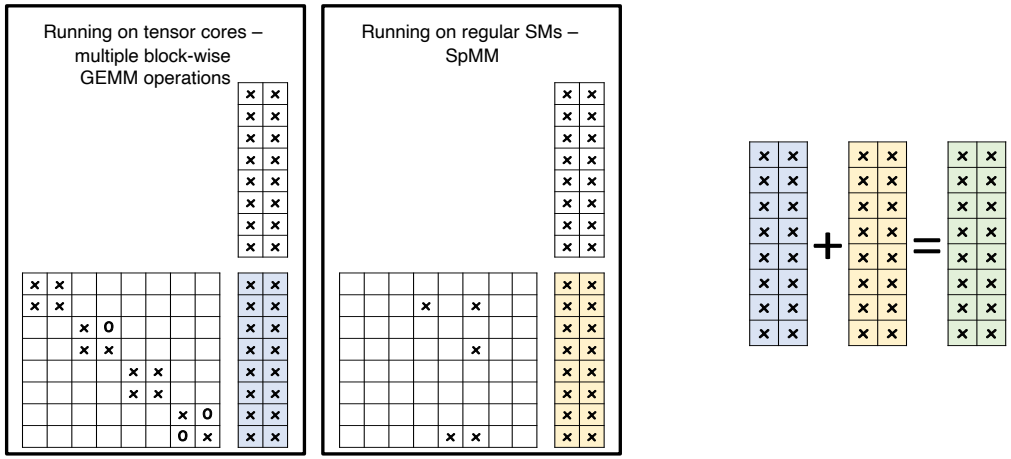
Mixed-Precision Kernels - SpMM

SpMM: Use block matrices from multiphysics use cases or find blocks of entries to do low-precision multiplies but accumulate in higher precision.

Use cases

- **Multiphysics:** Natural block sparse matrices and block dense vectors
  - Block sizes are not what tensor core wants
- **General use case:** Identify blocks in arbitrary sparse matrices
  - Blocks can be sparse too, not supported in V100, special ML like use case supported in A100. (Results are on V100)

SpMV with sparse and Dense portions split for Tensor Cores



Early Results

- **Kokkos Kernels implementation by Carl Pearson,** CUDA version by Gordon Moon
- **CUDA version**: Up to 2x improvement on SpMM on synthetic use cases
- 0.64x – **1.7x faster than cuSPARSE**, 0.74 - **2.3x faster than academic state-of-the-art** on Multiphysics (SPARC-like), poor performance when block sizes do not match hardware which requires padding and more computation

Caveats

- WMMA (low level NVIDIA interface) implementation
- Hard to do Kokkos portability

TRILINOS

Mixed-Precision in Solvers

- **GMRES-IR:** Develop solver algorithms that use low precision advantages, but provide high precision accuracy.
- Trilinos based experience:
  - **Experimental Evaluation of Multiprecision Strategies for GMRES on GPUs**, Jennifer Loe, Christian Glusa, Ichitaro Yamazaki, Erik Boman, Siva Rajamanickam, IPDPSW, 2021
  - **A Study of Mixed Precision Strategies for GMRES on GPUs,** Jennifer Loe, Christian Glusa, Ichitaro Yamazaki, Erik Boman, Siva Rajamanickam, https://arxiv.org/abs/2109.01232

Iterative Refinement with GMRES (GMRES-IR)

Algorithm 1 Iterative Refinement with GMRES Error Correction

1: $r_0 = b - Ax_0$ [double]
2: **for** $i = 1, 2, \ldots$ until convergence: **do**
3: Use GMRES(m) to solve $Au_i = r_i$ for correction u_i [single]
4: $x_{i+1} = x_i + u_i$ [double]
5: $r_{i+1} = b - Ax_{i+1}$ [double]
6: **end for**

(At each restart, update solution vector and recompute residuals in double precision.)

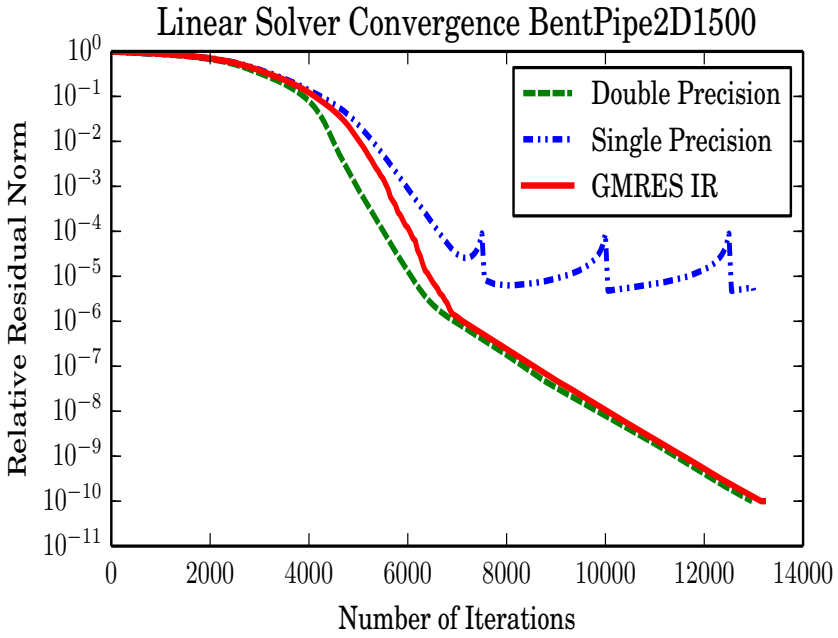
Note: We store TWO copies of matrix A (double and single).

Not a new algorithm. See related works:

- Neil Lindquist, Piotr Luszczek, and Jack Dongarra. *Improving the performance of the GMRES method using mixed-precision techniques.*
- Hartwig Anzt, Vincent Heuveline, and Bjorn Rocker. *Mixed precision iterative refinement methods for linear systems: Convergence analysis based on Krylov subspace methods.*
- Erin Carson and Nicholas J. Higham. *Accelerating the solution of linear systems by iterative refinement in three precisions.*

Mixed-Precision in Solvers

- **Test Problem:**
  - 2D convection-diffusion, 5-pt stencil (Highly nonsymmetric.)
  - n = 2.25 million, nnz = 11,244,000
- Convergence Tolerance: 1e-10
- **GMRES Convergence:**
  - **Single:** Stalls near 1e-5
  - **Double:** 12,967 iterations, 50.26 seconds
  - **IR:** 13,150 iterations, 38.03 seconds



GMRES-IR convergence follows convergence of GMRES Double!

Mixed-Precision in Solvers

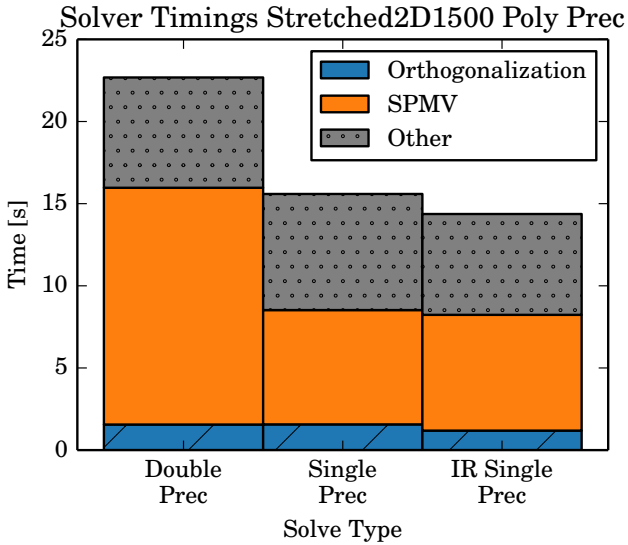
Test Problem (same as previous slide):

- 2D Laplacian, Stretched Grid
- n = 2.25 million
- Polynomial preconditioner based upon GMRES polynomial**

Three solves compared:

1. GMRES double w/ double precision polynomial. (left)
2. GMRES double w/ single precision polynomial. (middle)
3. GMRES-IR w/ single precision polynomial. (right)

- (Solve times do not include preconditioner creation.)



Takeaways:

- Single precision preconditioning improves solve time ~ 30%.
- GMRES-IR improves solve time even more.
- Polynomial preconditioning shifts main expense to SpMV rather than dense orthogonalization kernels.

Lot of work remains

- Several challenges remain on architectures, algorithms, and software
  - Precisely characterize the need of higher precision at different levels of the software stack similar to solvers and kernels
  - Adapt the software to handle the need while maintaining confidence in the higher precision results