GPU Accelerated Memory-bound Linear Algebra Kernels

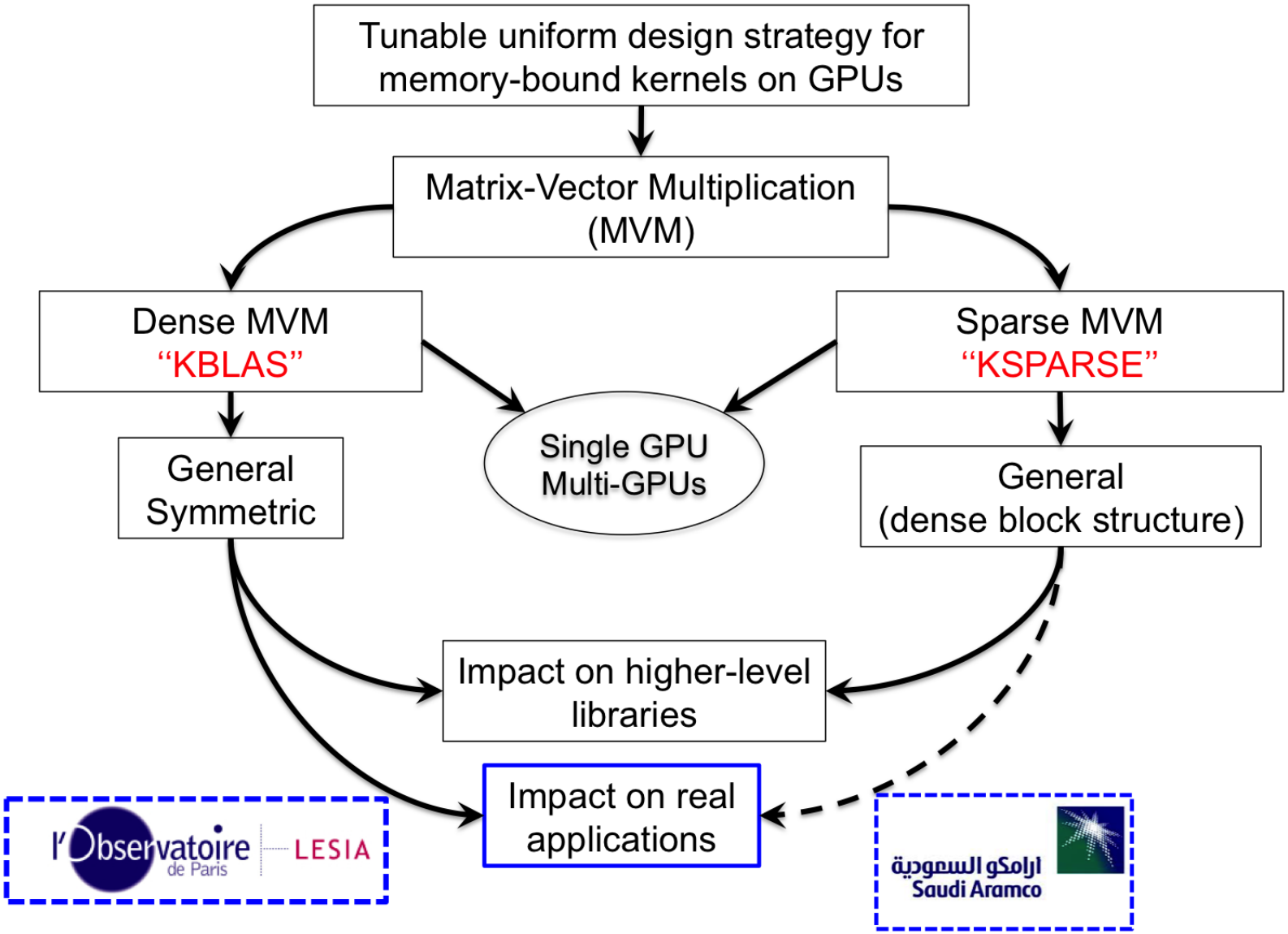
Ahmad Ahmad (Ahmad Abdelfattah)

Supervised by:
David Keyes
Hatem Ltaief

Extreme Computing Research Center (ECRC), KAUST

April 17th, 2015

What this work is all about ...



Outline

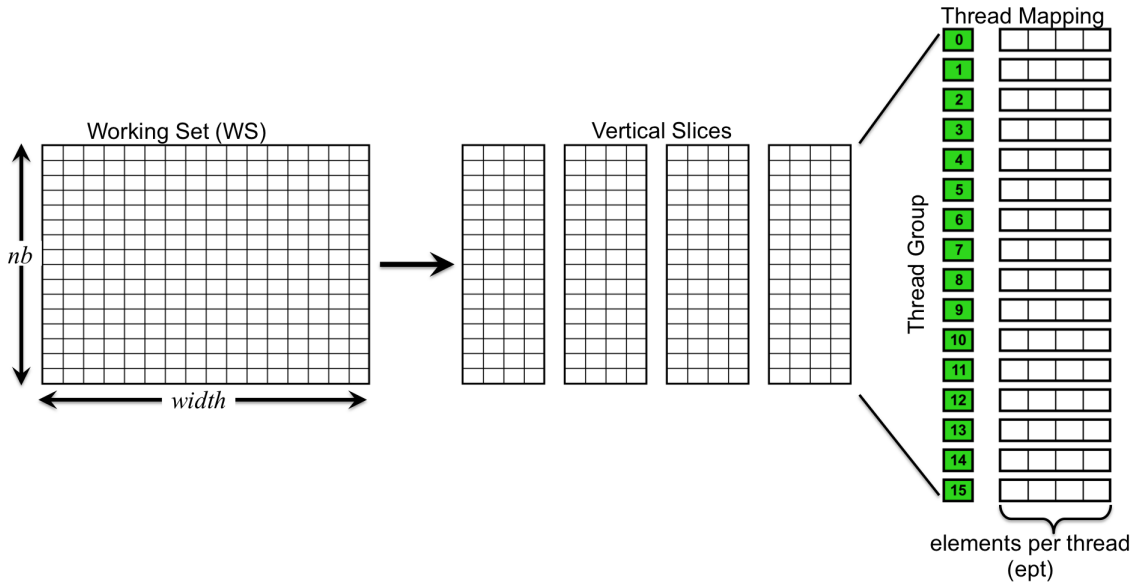
Outline

What is proposed?

- A set of design ideas that are incorporated together in one design strategy
  - Controllable through a set of tuning parameters
- Applied to dense MVM (GEMV and SYMV kernels)
  - Single GPU and multi-GPUs (block-column 1D cyclic format)
- Applied to sparse MVM (BSR format)
  - Single GPU and multi-GPUs (block row 1D cyclic)

Design Ideas

- Hierarchical Register Blocking
  - An input matrix is subdivided into square or rectangular blocks
  - Working set (WS): the minimum amount of work assigned to a TB
    - Can be a block
    - Can span multiple adjacent blocks
    - Can be part of a block
  - Dimensions of the working sets are tunable in most cases



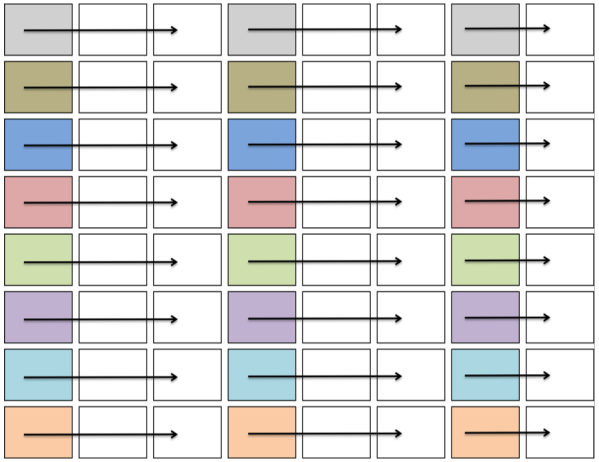
Design Ideas

- Double Buffers: To enable data prefetching
- Always process in registers
  - Except for final reduction in shared memory
- Latency Hiding
  - On thread level: Assign more work per thread (ILP)
  - On TB level: Run multiple warps per TB
- Collaboration among TBs
  - Using atomic operations

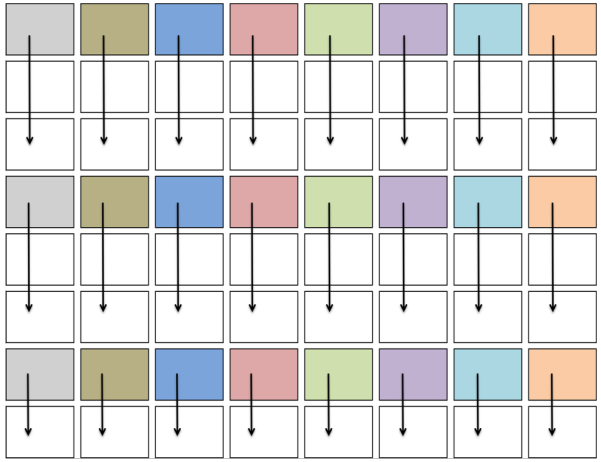
Outline

Dense MVM: Grid Design

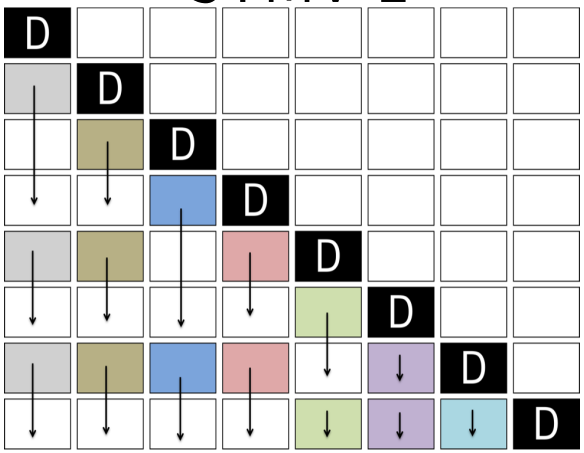
GEMV-N



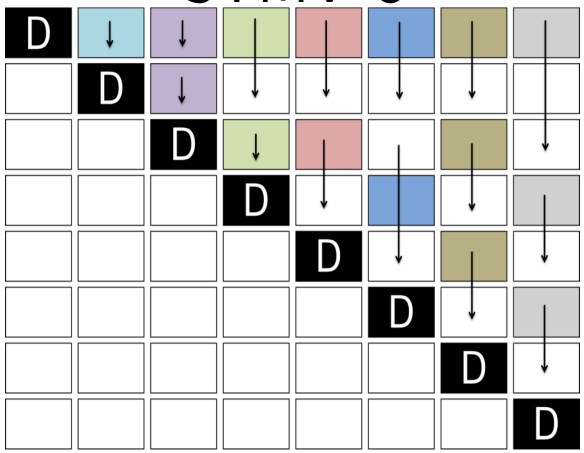
GEMV-T



SYMV-L

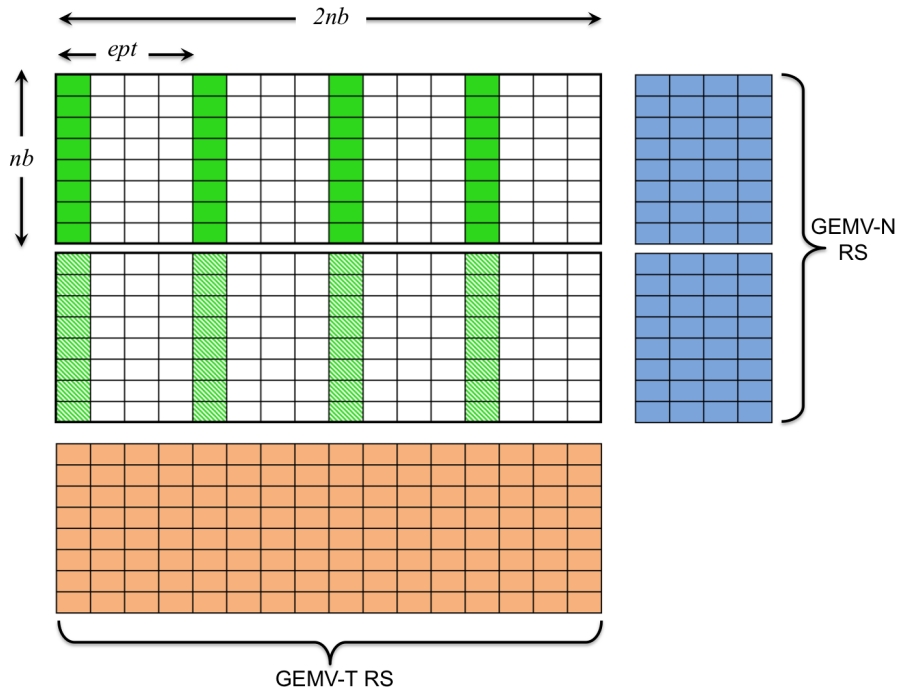


SYMV-U



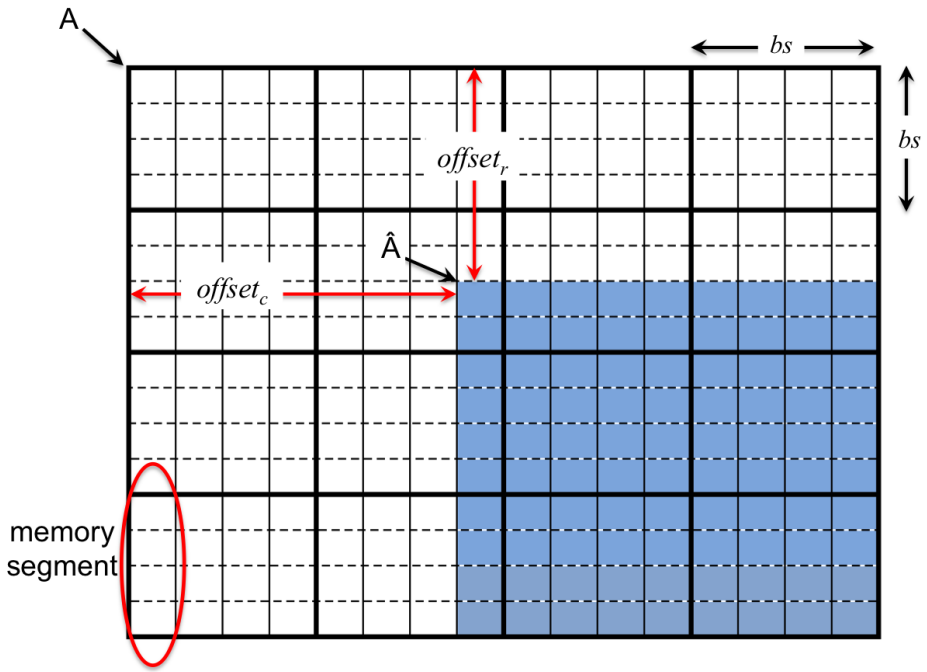
Dense MVM: TB Design

- A block ≡ two working sets
- There is a variant of GEMV where a block ≡ one working set



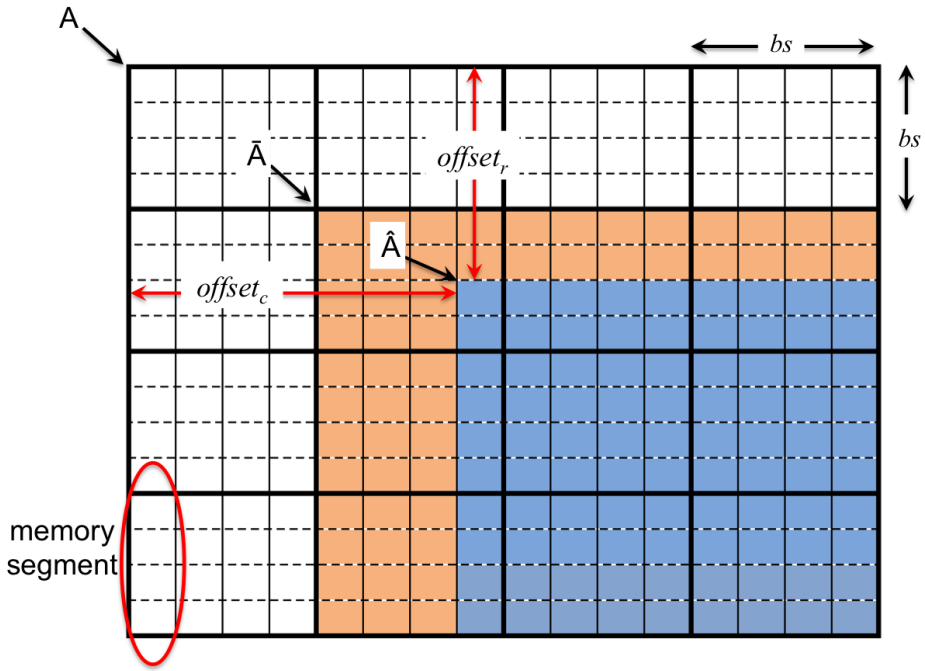
Multiplication by a Submatrix

- Leading dimension of dense matrices are often padded to facilitate coalesced memory access
- However, multiplication by a submatrix $\hat{A}$ does not guarantee coalesced memory access



The Offsetting Technique

- Multiplies by $\bar{A}$ instead, guarantees coalesced memory access for any submatrix
- Needs a new interface to convey the offset information (GEMV-OFFSET and SYMV-OFFSET)



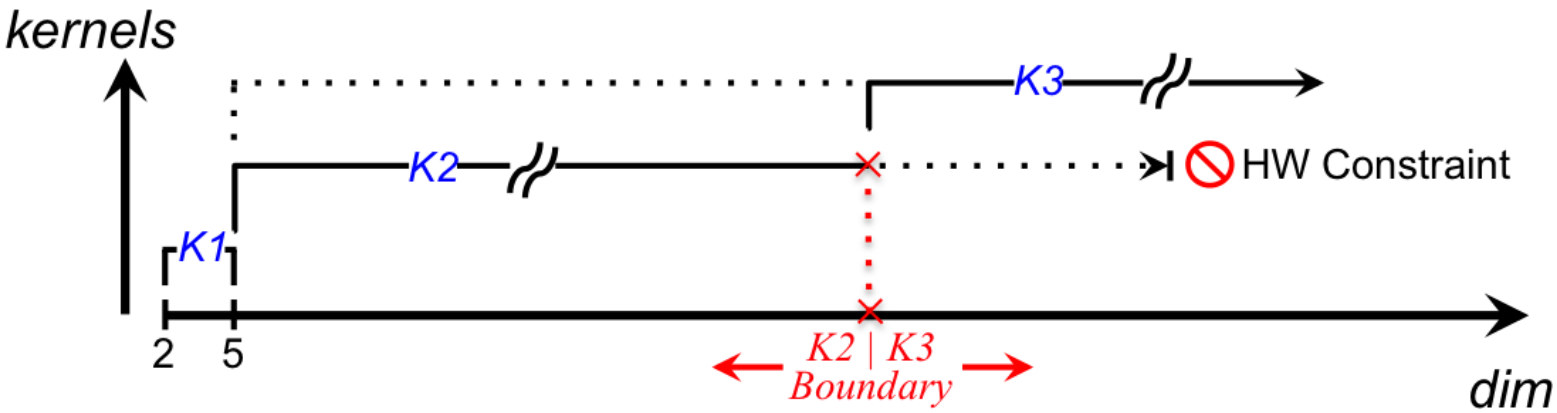
Multi-GPU kernels

- Two kernels: GEMV-MGPU and SYMV MGPU
- Very similar to their respective single GPU kernels
  - Each TB/thread has to compute a global ID with respect to other TBs/threads across all GPUs
  - The multi-GPU kernels use the offsetting technique

Outline

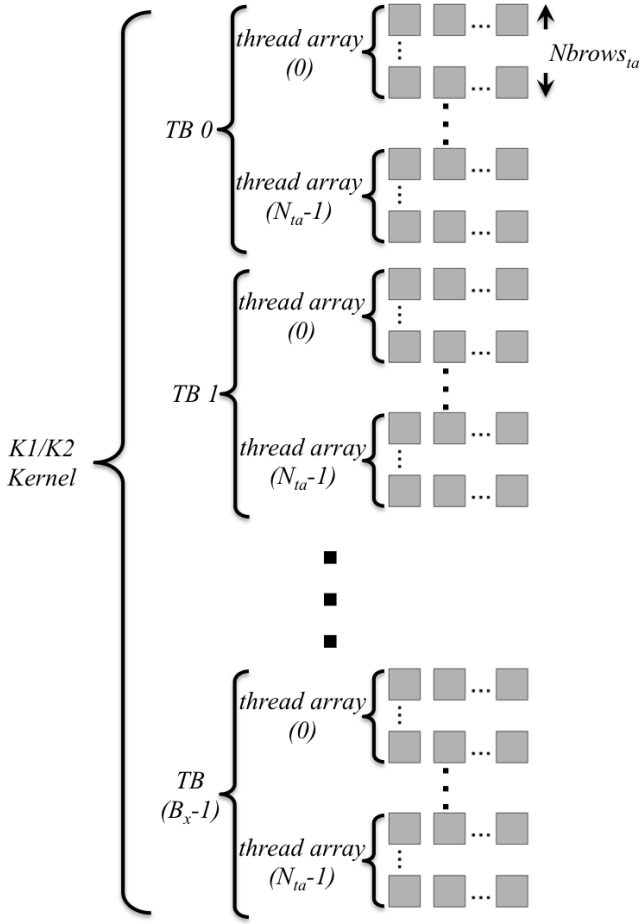
Splitting Block Size Range

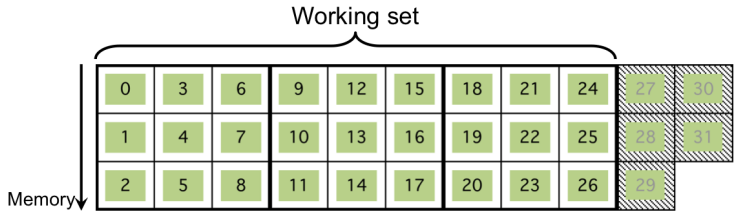
- Generally, block size *bs* can be any value
- It is difficult to have one kernel that covers the entire range
- We propose three kernels to cover the range of *bs*
  - *K1*: small blocks ($2 \leq bs \leq 5$)
  - *K2*: medium blocks ($5 \leq bs \leq 45$)
  - *K3*: large blocks ($bs > 45$)
- Ranges are flexible, except for *K1*



Kernel *K1*: Small Blocks

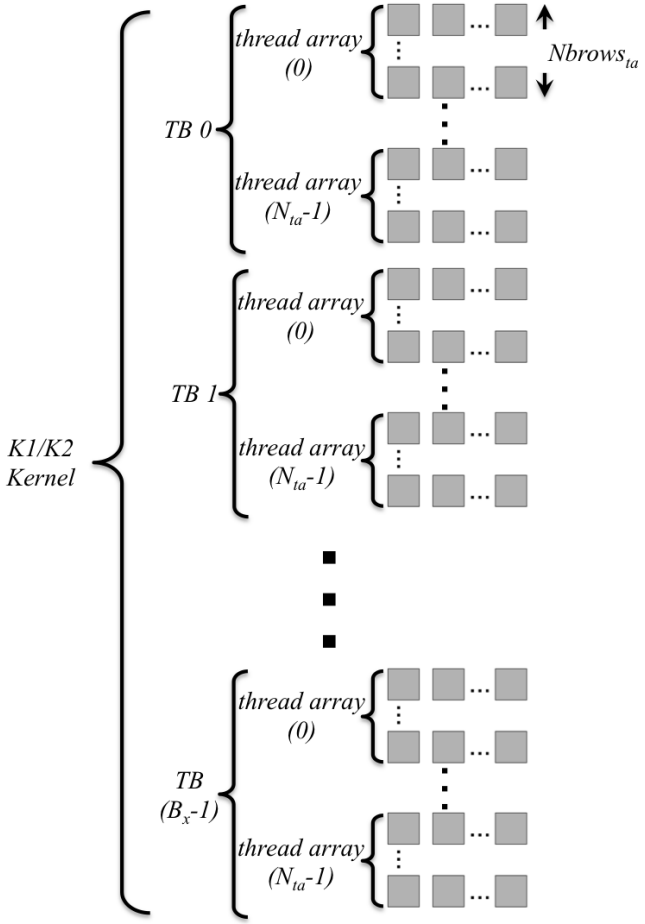
- Thread arrays are strictly warps

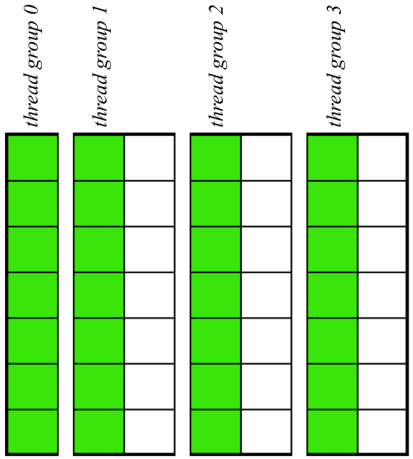




Kernel *K2*: Medium Blocks

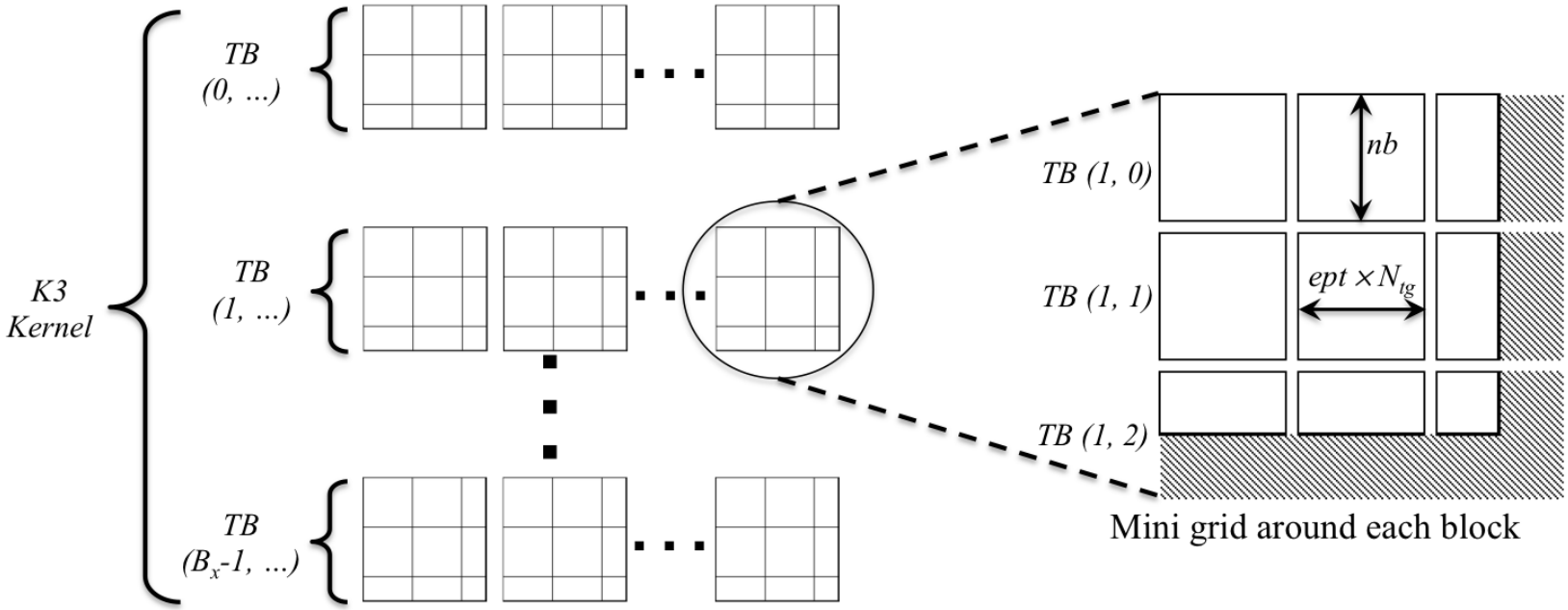
- Thread arrays are NOT strictly warps





Kernel *K3*: Large Blocks

- A working set is part of a block
- Design is typical to a GEMV kernel within a block



Multi-GPU kernels

- KSPARSE uses 1D cyclic distribution of block rows
- Each GPU ends up computing certain segments of y
- The `RowPtr` array has to be reevaluated on each GPU
- The single-GPU BSRMV kernel can be used out of the box

Outline

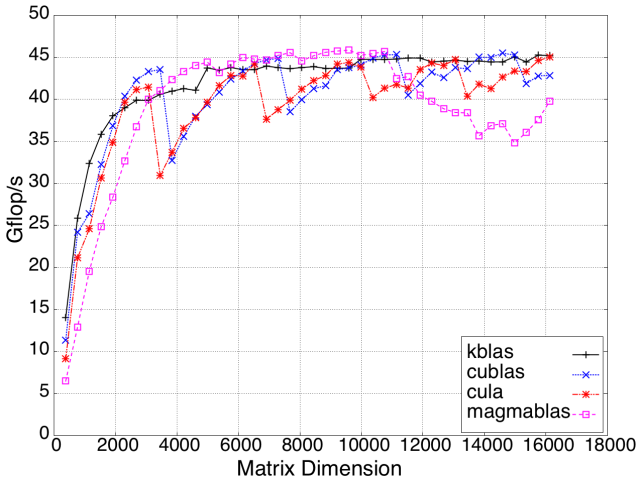
System Setup

- Single GPU experiments
  - 16-core Intel Xeon CPU E5-2650 (2.00GHz)
  - 4 Tesla K20c GPUs (ECC off)
  - Ubuntu 14.04.1 LTS
  - CUDA driver version 340.32
  - CUDA Toolkit 5.5
- Multi-GPU experiments
  - Located at The Swiss National Supercomputing Center
  - 16 core Intel Xeon CPU E5-2670 (2.60GHz)
  - 8 K20c GPUs (ECC off)
  - CentOS release 6.3
  - CUDA driver version 331.62
  - CUDA Toolkit 5.5
- Sustained memory bandwidth of the GPU is measured at 184.18 GB/s (out of 208 GB/s theoretically)

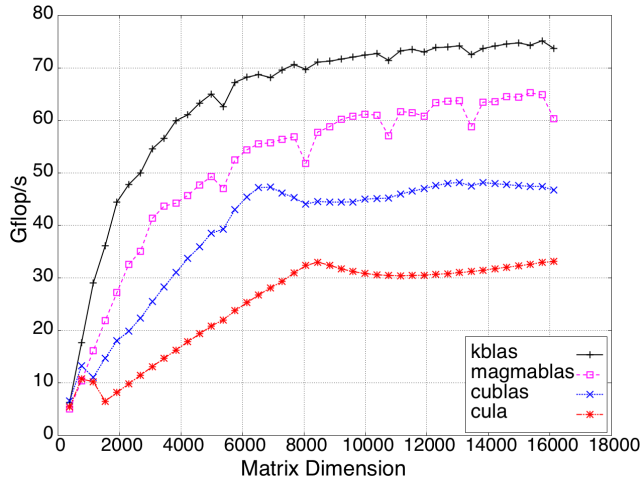
KBLAS Performance (K20c GPU)

Higher is better

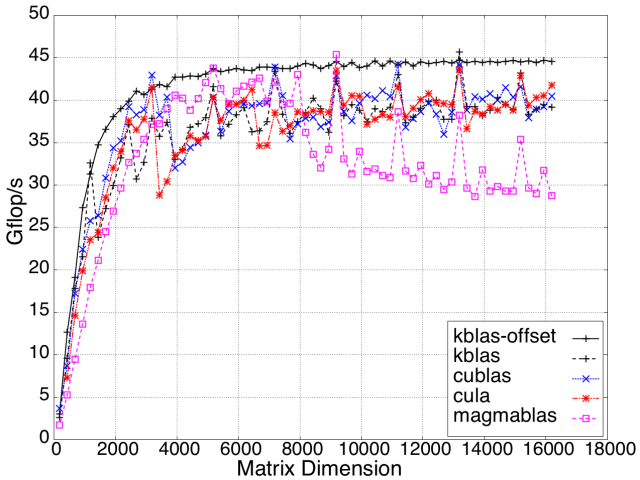
DGEMV



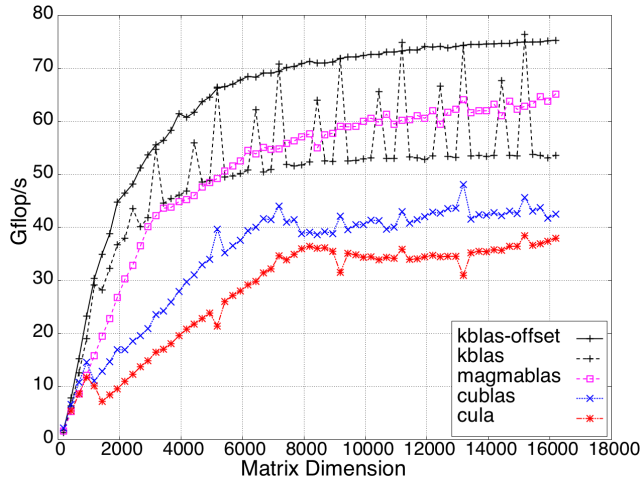
DSYMV (integrated into cuBLAS)



DGEMV-OFFSET



DSYMV-OFFSET

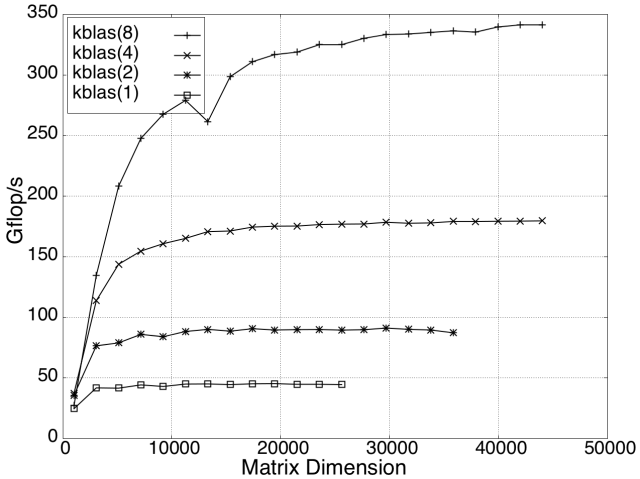


- DGEMV: smoother performance, 98% of memory performance
- DSYMV: 15% asymptotic speedup, 1.83x speedup for relatively small matrices

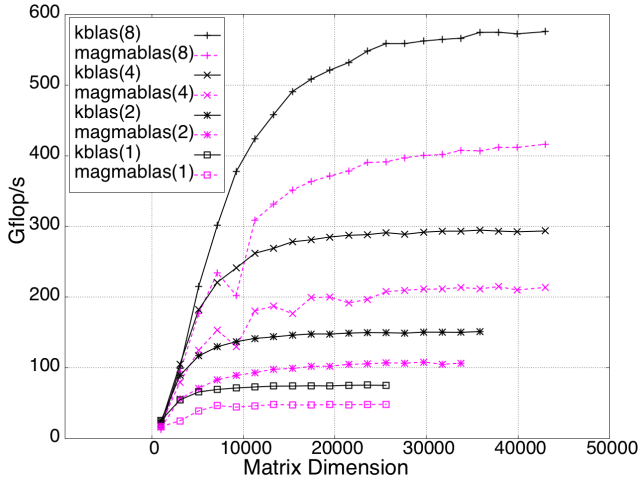
KBLAS Multi-GPU Performance

Higher is better

DGEMV-MGPU



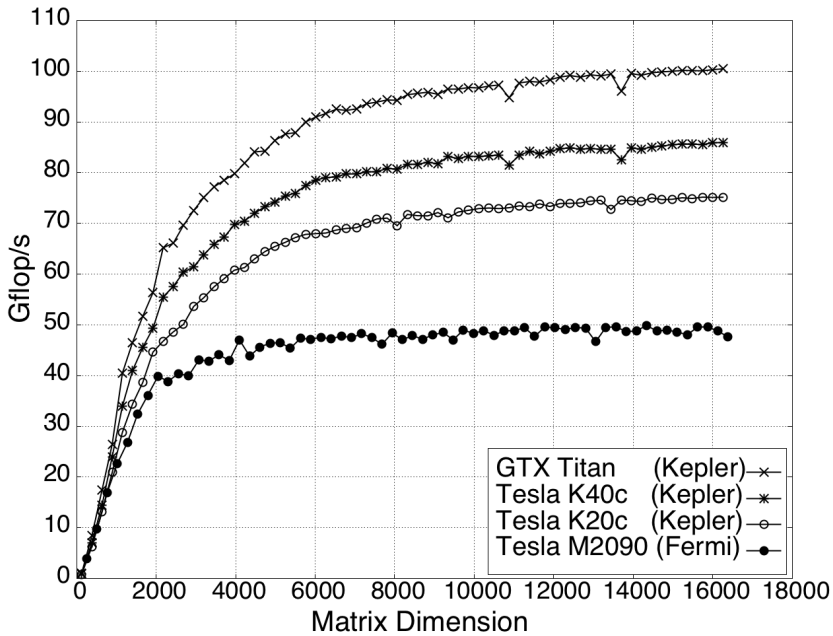
DSYMV-MGPU



- Performance is scaled almost linearly across multi-GPUs
- DSYMV: 38% asymptotic speedup on 8 GPUs

Dense MVM: Maintaining Performance

- As an example, DSYMV on four different GPUs
  - Fermi M2090 (130.32 GB/s) - 76%
  - Kepler K20c (184.18 GB/s) - 82%
  - Kepler K40c (230.92 GB/s) - 75%
  - GTX TITAN (254.77 GB/s) - 79%

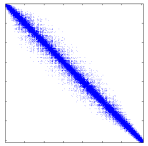


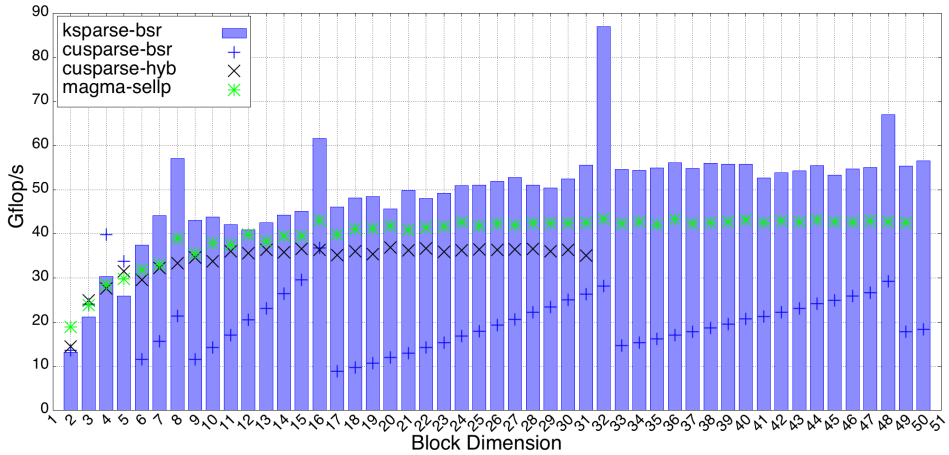
Sparse MVM: Matrix Test Suite

- We did some tests against synthetic matrices
- We also used matrices from the UFlorida collection, promoting every non-zero to a square block of a given size

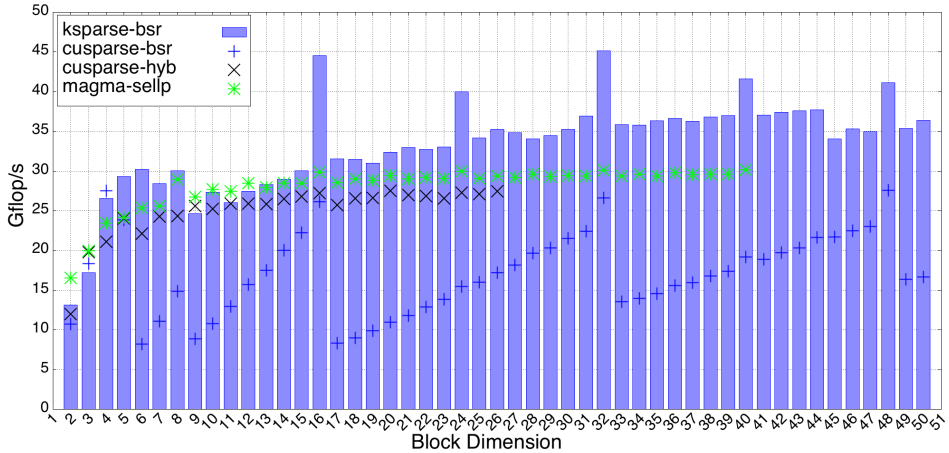
Name	Size	Non-zeros	Description
airfoil_2d	14,214	259,688	Computational fluid dynamics
bauru5727	40,366	145,019	Eigenvalue/model reduction pb
cage10	11,397	150,645	Directed weighted graph
hvdc1	24,842	158,426	Power network pb
rajat22	39,899	195,429	Circuit simulation

BSRMV against Other Formats (airfoil_2d)



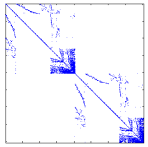


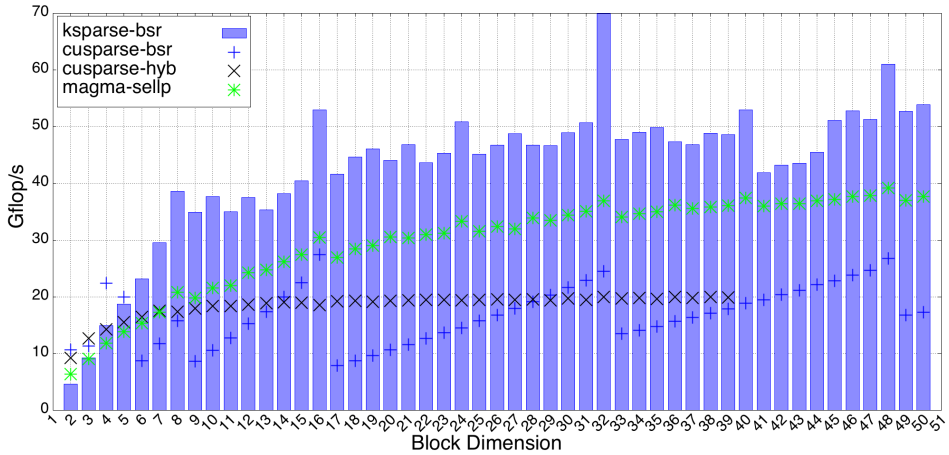
Double Precision



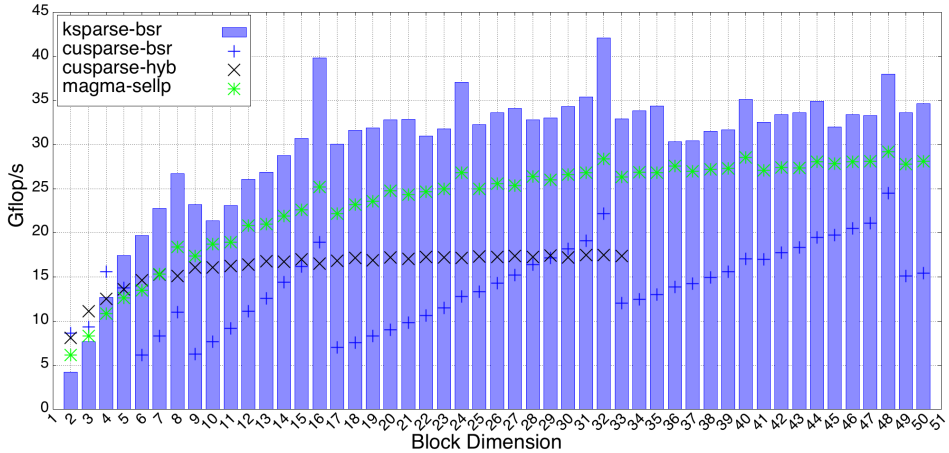
- For SP/DP, speedups are up to 1.71x/1.64x against cuSPARSE-HYB, 2.00x/1.50x against MAGMA-SELLP, and 5.21x/3.78x against cuSPARSE-BSR

BSRMV against Other Formats (bauru5727)



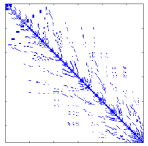


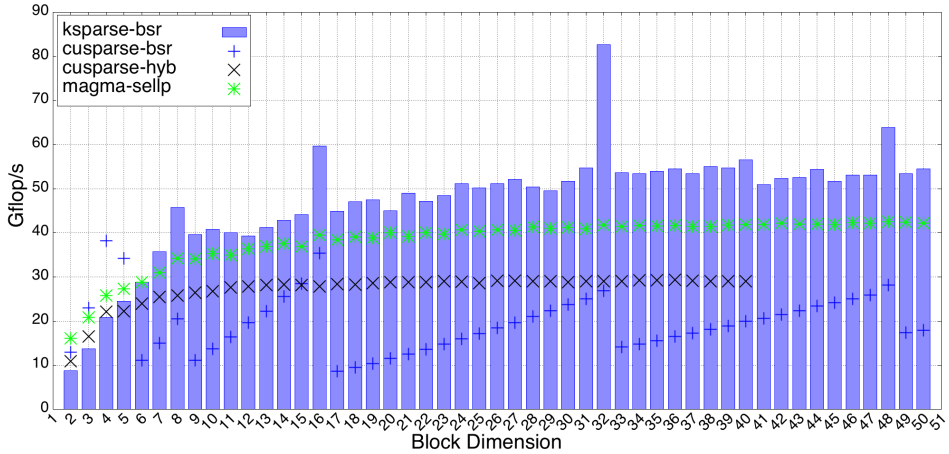
Double Precision



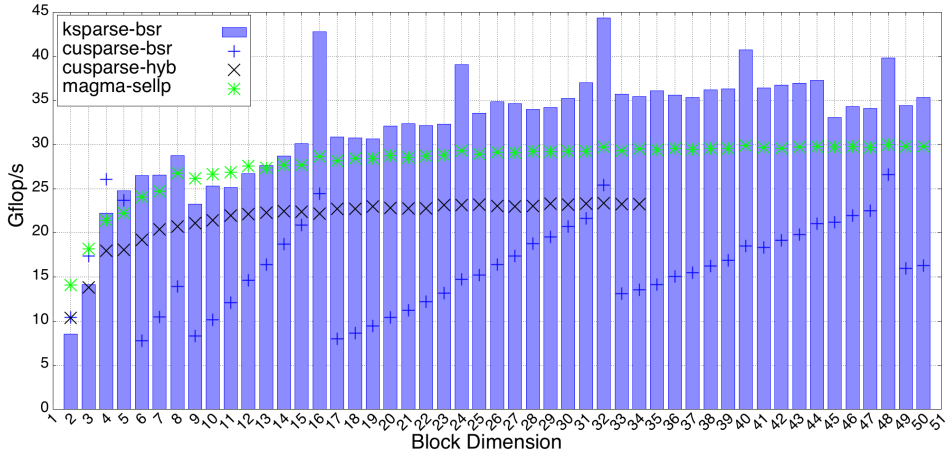
- For SP/DP, speedups are up to 3.50x/2.41x against cuSPARSE-HYB, 1.89x/1.58x against MAGMA-SELLP, and 5.24x/4.28x against cuSPARSE-BSR

BSRMV against Other Formats (cage10)



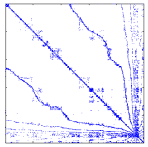


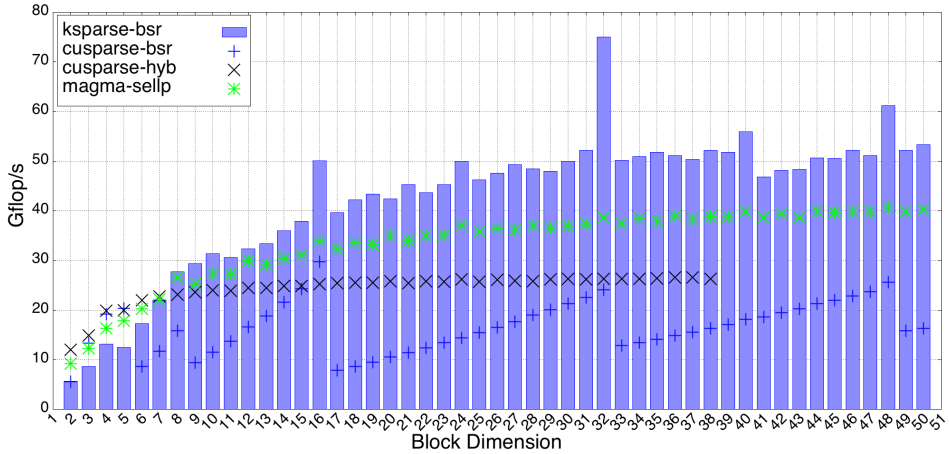
Double Precision



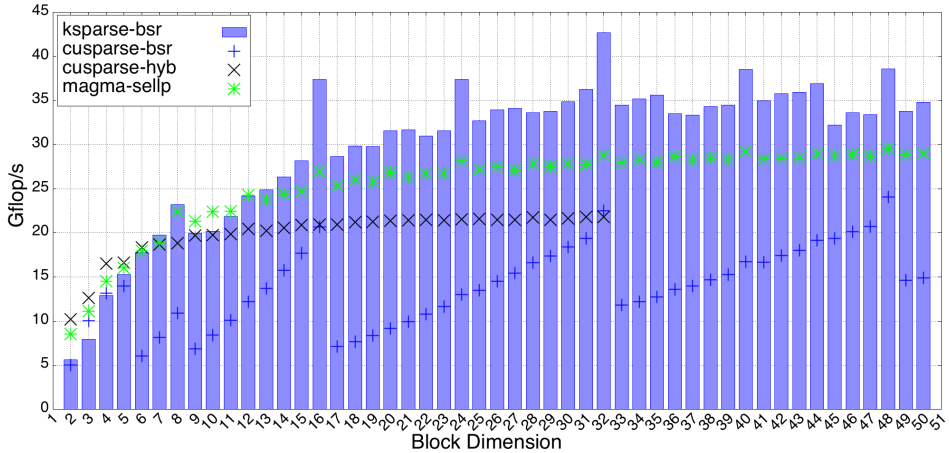
- For SP/DP, speedups are up to 2.84x/1.93x against cuSPARSE-HYB, 1.98x/1.49x against MAGMA-SELLP, and 5.19x/3.85x against cuSPARSE-BSR

BSRMV against Other Formats (hvdc1)



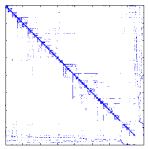


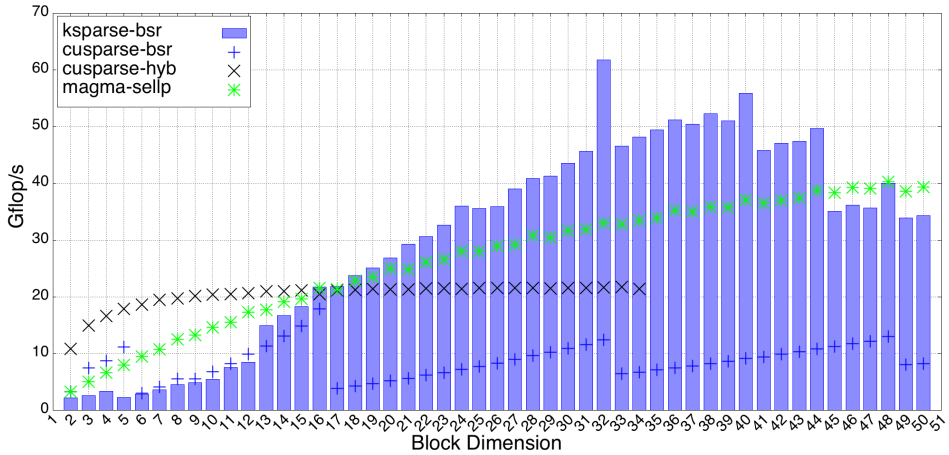
Double Precision



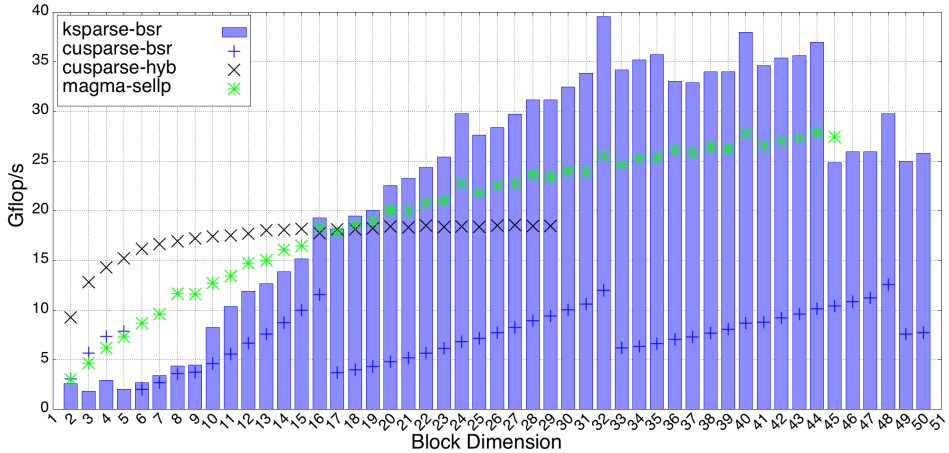
- For SP/DP, speedups are up to 4.39x/2.57x against cuSPARSE-HYB, 1.96x/1.49x against MAGMA-SELLP, and 4.94x/3.58x against cuSPARSE-BSR

BSRMV against Other Formats (rajat22)





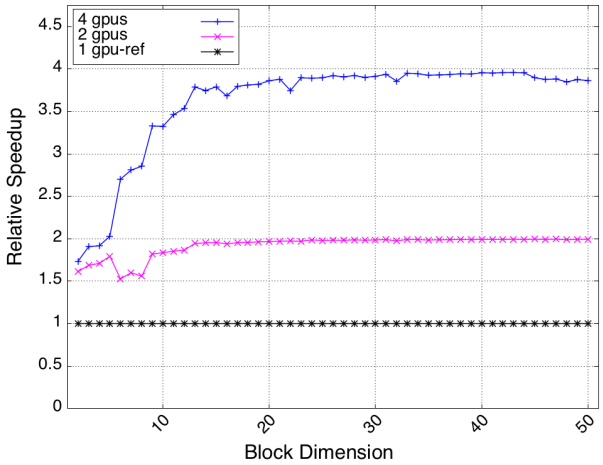
Double Precision



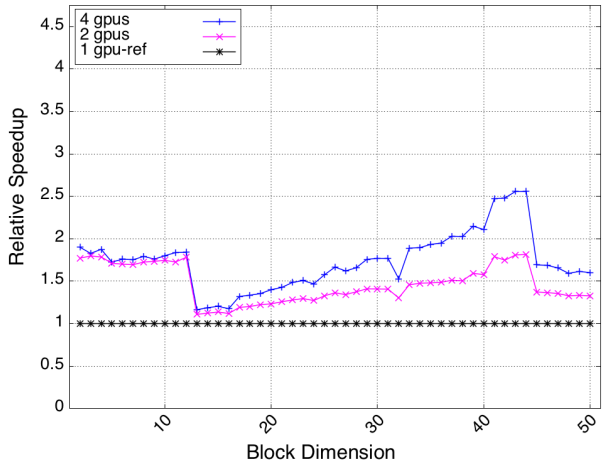
- For SP/DP, speedups are up to 2.85x/1.69x against cuSPARSE-HYB, 1.88x/1.55x against MAGMA-SELLP, and 7.17x/5.54x against cuSPARSE-BSR

Sparse MVM: Multi-GPU Scaling

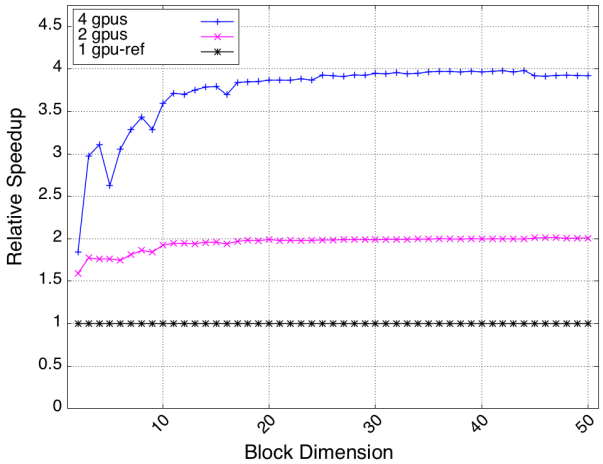
airfoil_2d (SP)



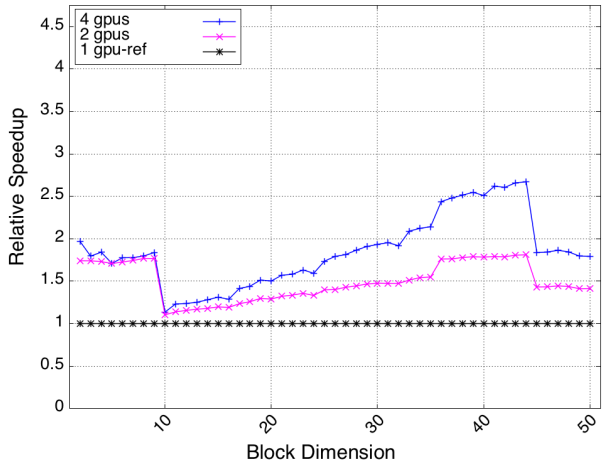
rajat22 (SP)



airfoil_2d (DP)

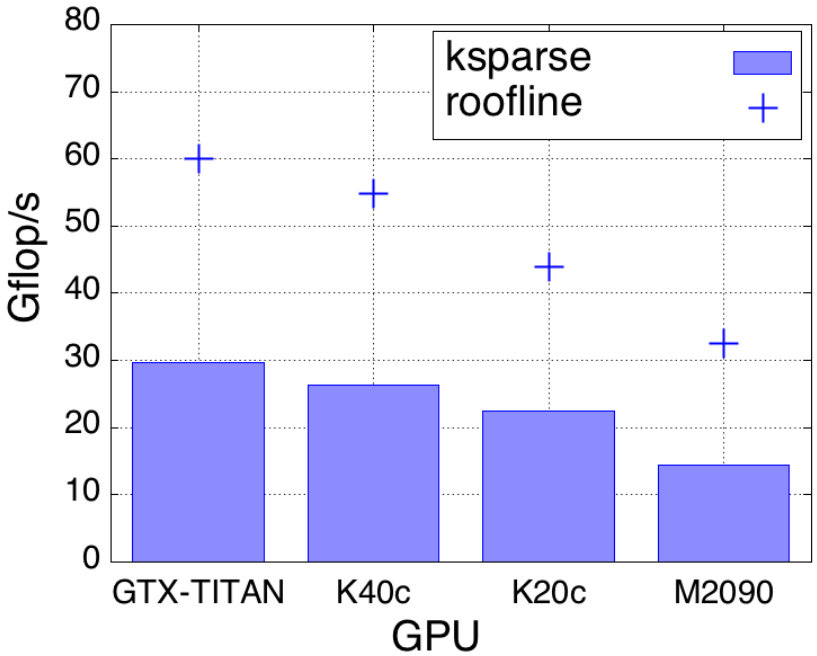


rajat22 (DP)



Sparse MVM: Tuning KSPARSE

- Consider DBSRMV for $bs = 7$
- Same level of performance is maintained across different GPUs
  - Performance is within 46.5%, 45.76%, 48.97%, and 44.29% from its roofline on GTX TITAN, K40c, K20c, and M2090 GPUs

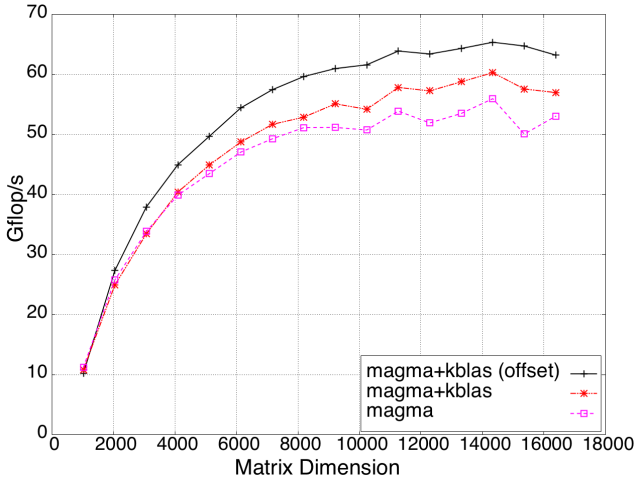


Outline

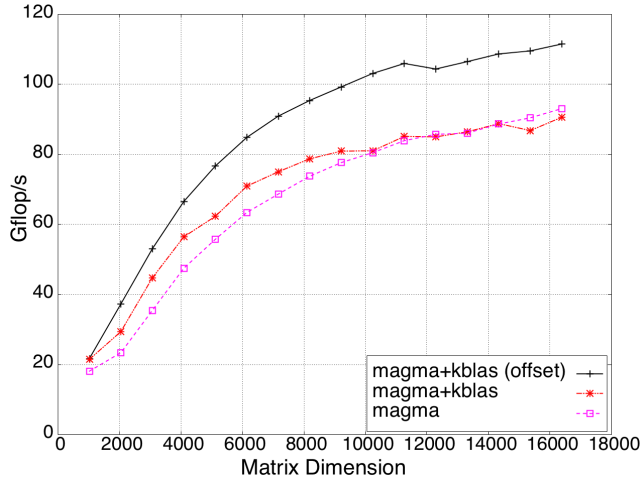
Impact on MAGMA

Higher is better

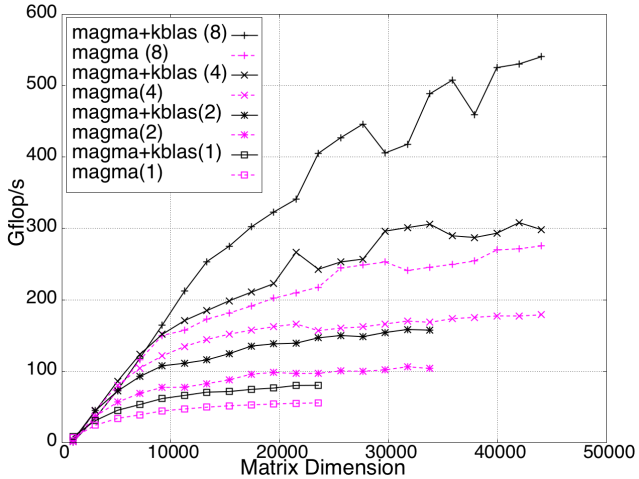
DGEBRD



DSYTRD



DSYTRD-MGPU

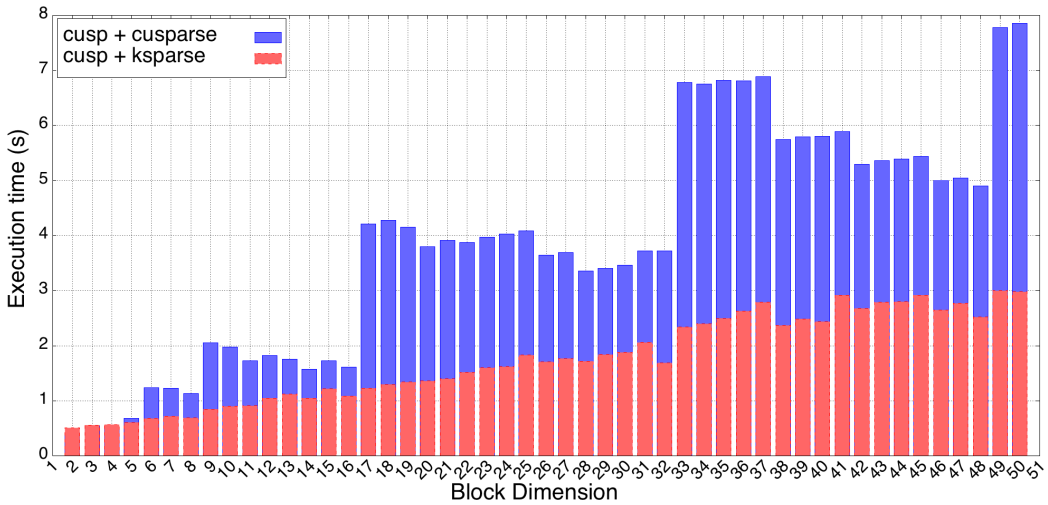


- Performance improvements up to: 29% for DGEBRD, 59% for DSYTRD, and 103% for DSYTRD-MGPU

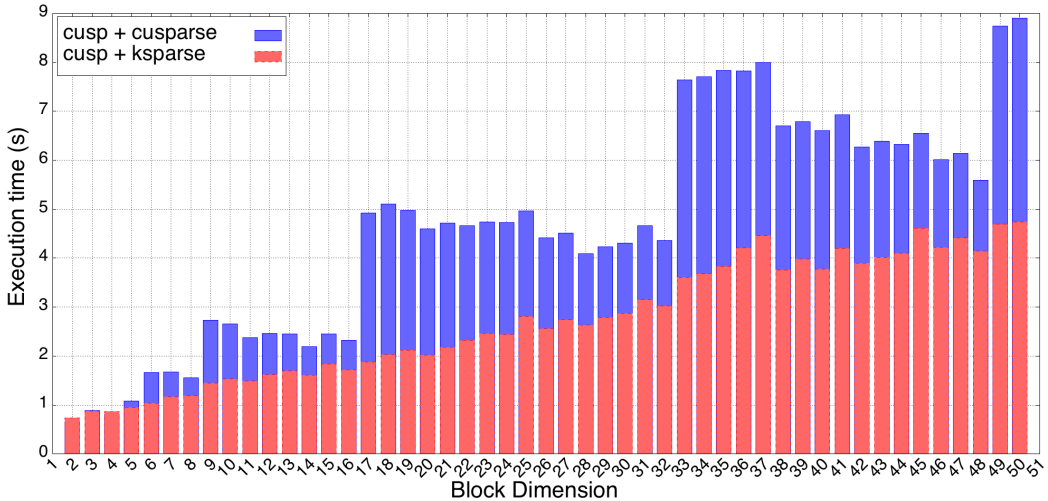
Impact on CUSP: GMRES

Lower is better

single precision

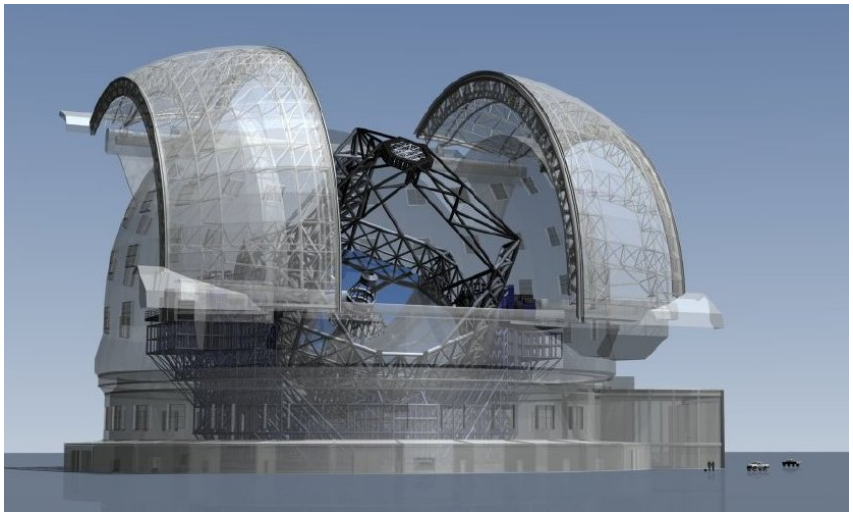


double precision



- For block sizes larger than 4, GMRES+KSPARSE is up to 3.42x (SP) and 2.62x (DP) faster

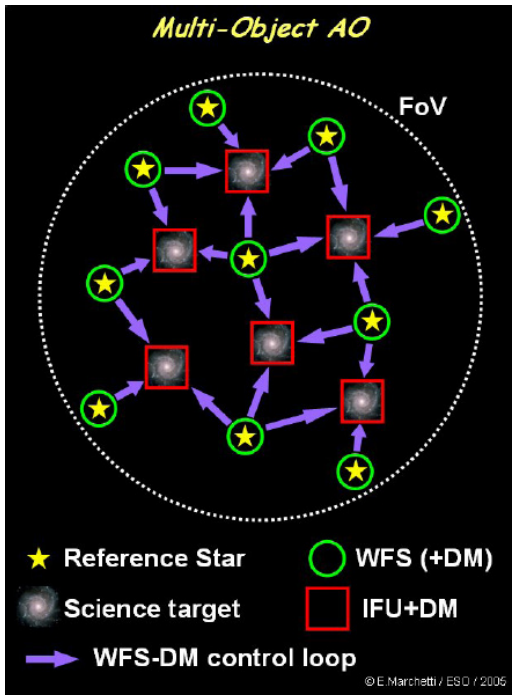
Impact on a Real Application: The E-ELT



Credits: ESO (http://www.eso.org/public/outreach/copyright/)

- In collaboration with Paris Observatory
- The project helps the design of MOSAIC
- MOSAIC is a multi-object spectrograph (MOS), proposed for the European Extremely Large Telescope (E-ELT)
  - The largest optical/near-infrared telescope in the world
    - weighs about 2700 tons, 39m is the main mirror diameter
  - Named the "biggest eye on the sky"
  - First light is expected early 2020s

Impact on E-ELT: Multi-object Adaptive Optics (MOAO)



Credits: ESO (http://www.eso.org/public/outreach/copyright/)

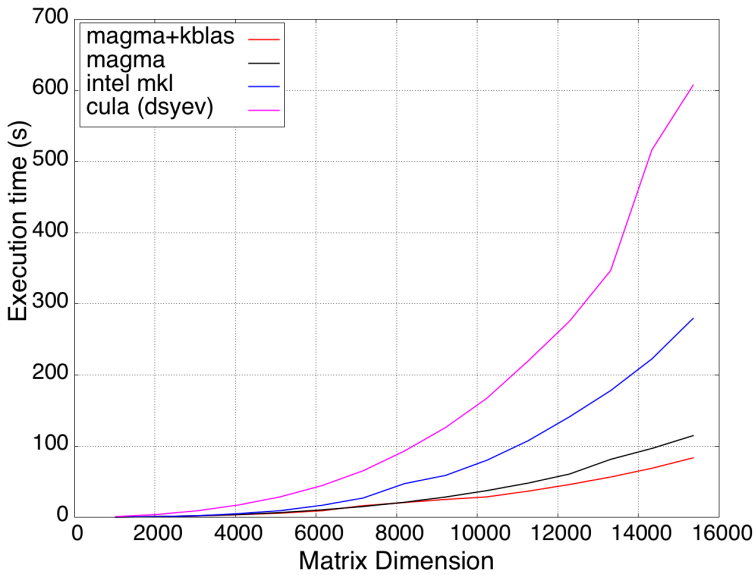
- MOAO is the main concept behind the design of MOSAIC
- MOAO helps observe the evolution of a number of objects in parallel
- Fields of views are too large to be observed by a conventional AO system
- Only images of objects of interests are corrected

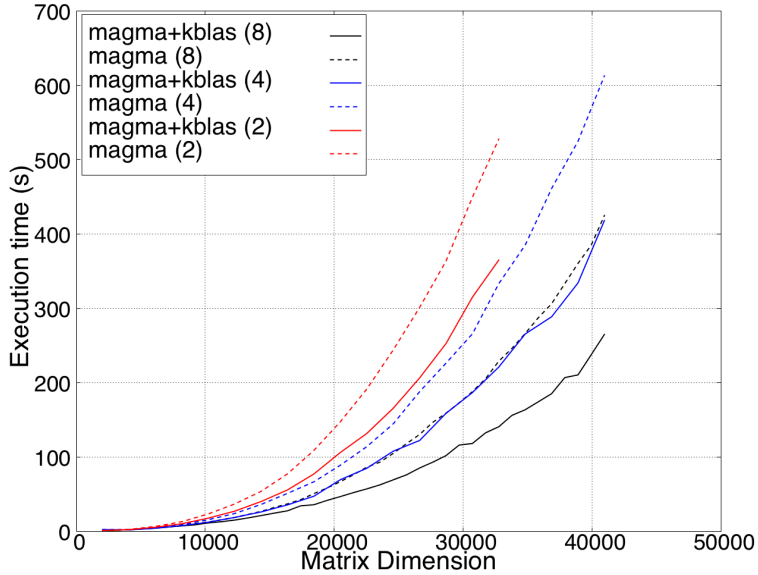
Impact on E-ELT: The Tomographic Reconstructor

- We want to simulate the image quality generated by the telescope
- Dependent on frequent computation of the tomographic reconstructor (R)
- $R = C_{tm} \cdot C_{mm}^{-1}$
- C_{mm} (40k×40k) is pseudo-inverted using SYEVD
- Contributions
  - Accelerate MAGMA-DSYEVD using KBLAS
  - An out-of-core DGEMM kernel

Impact on E-ELT: Results

DSYEVD Performance (lower is better)

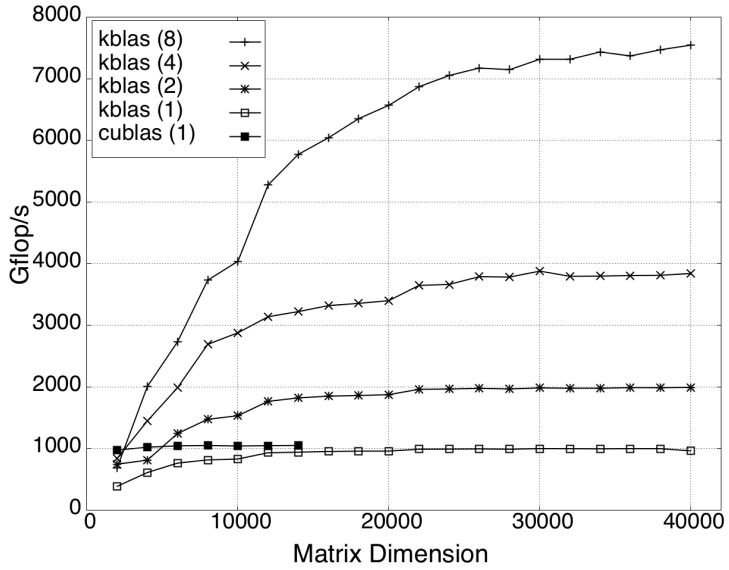


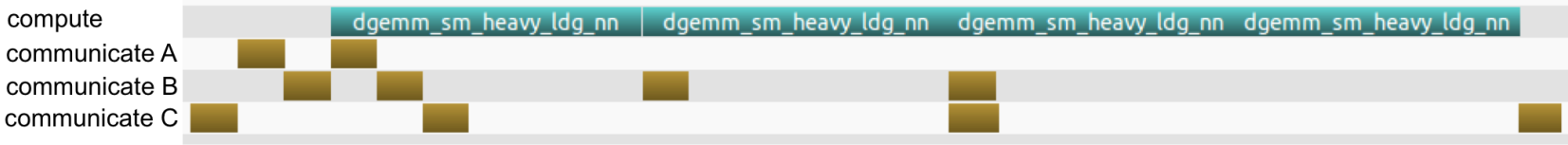


- Single GPU Performance
  - 7.2x, 3.4x, and 1.35x speedups over CULA, MKL, and MAGMA
- Multi-GPU Performance
  - 1.45x, 1.6x, and 1.7x speedups over MAGMA on 2, 4, and 8 GPUs

Impact on E-ELT: Results

DGEMM Performance





- Up to 7.6 Tflop/s on 8 K20c GPUs
- 90% close to linear scaling
- Includes initialization and cleanup times

Impact on E-ELT: Results

Overall Simulation Time: 263.49 s using 8 K20c GPUs

- 17.5x speedup over Intel MKL
- 60% improvement over MAGMA using 8 GPUs

Outline

In a nutshell

This work

- proposes a uniform design strategy for memory-bound kernels on GPUs
- focuses on MVM kernels (dense, sparse)$\times$(single GPU, Multi-GPUs)
- shows impact on higher-level libraries
- shows impact on a real application

Future Directions

- Autotuning
- Other memory-bound kernels
- Distributed memory systems
  - 2D cyclic layout (following ScaLAPACK)
- Batch operations and H-matrices (KSPARSE)

Thank You!

KBLAS: http://ecrc.kaust.edu.sa/Pages/Res-kblas.aspx
KSPARSE: http://ecrc.kaust.edu.sa/Pages/ksparse.aspx