February 23rd - 25th, 2017

Workshop on Batched, Reproducible, and Reduced Precision BLAS

Georgia Tech
Atlanta, GA

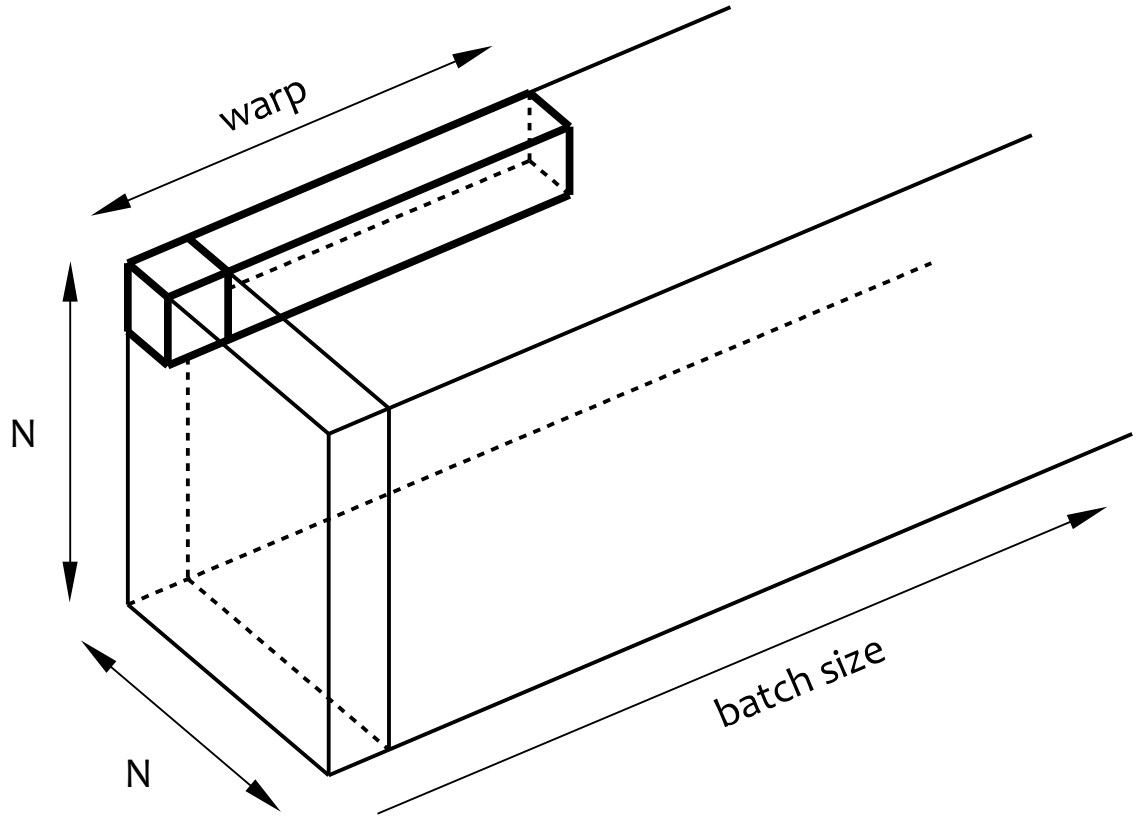
Autotuning Batched Kernels:

- **Batched Cholesky with Interleaved Layout**
- **The BONSAI Project**

J. Kurzak (PI)
P. Luszczek (PI)
M. Gates
Y. Tsai
M. Bachstein
Y. Pei

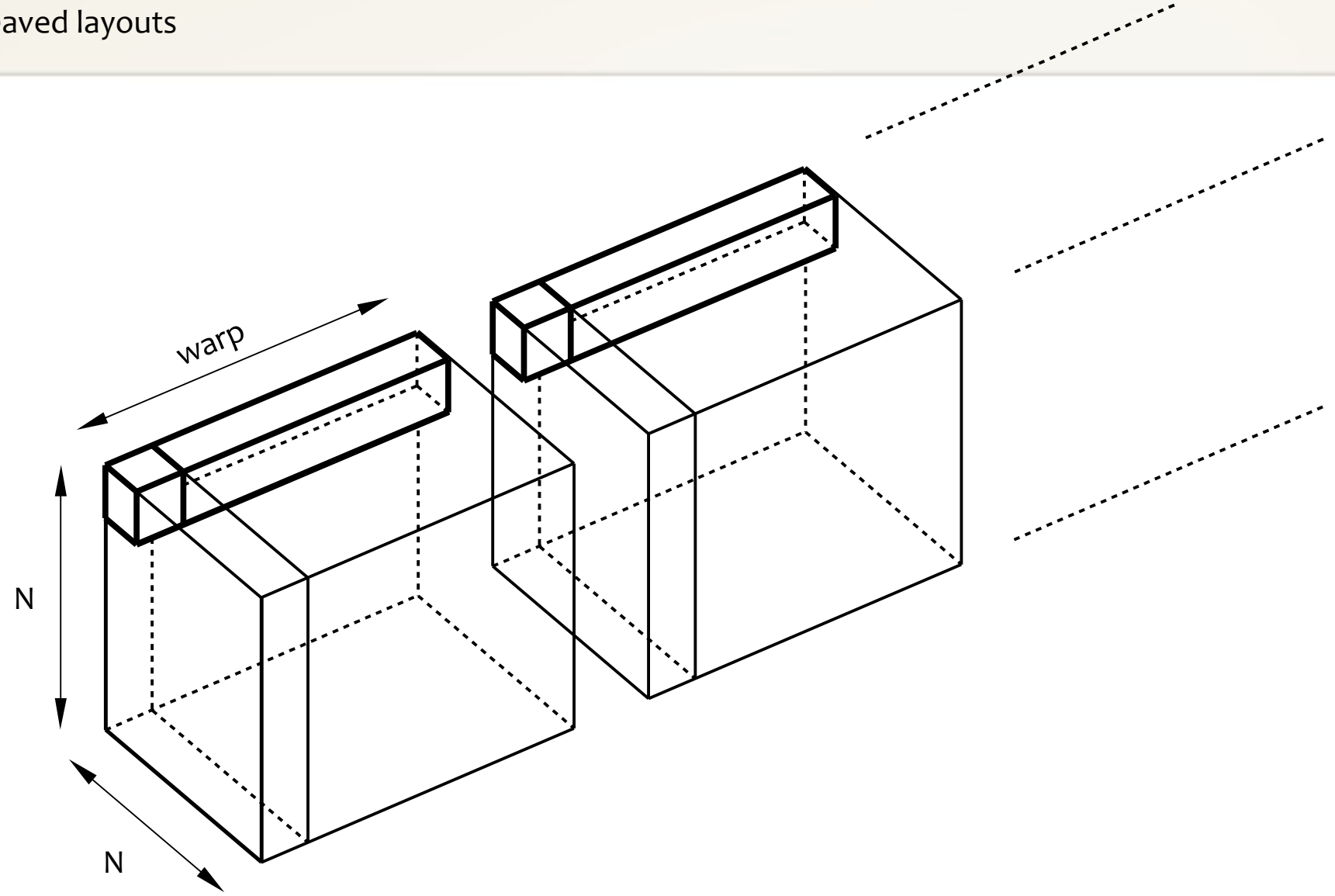
Batched Cholesky

interleaved layouts



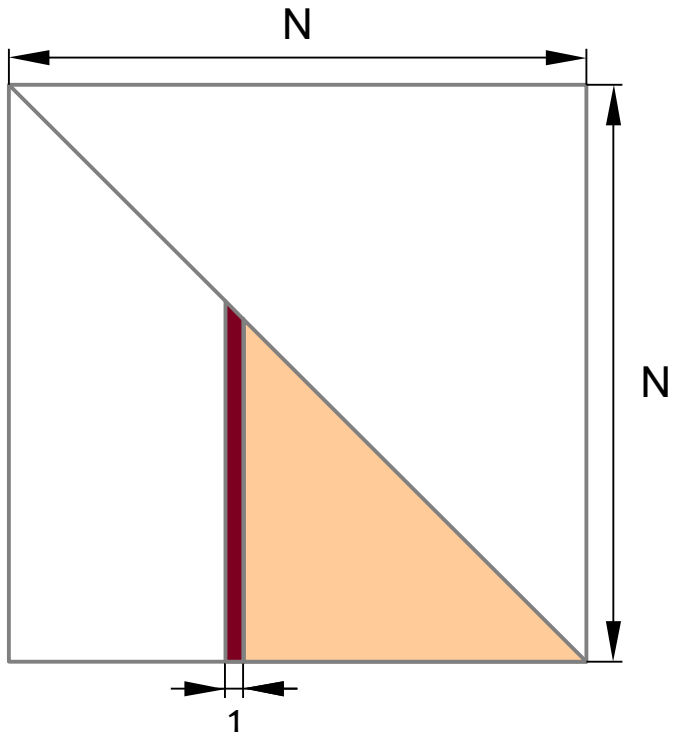
Batched Cholesky

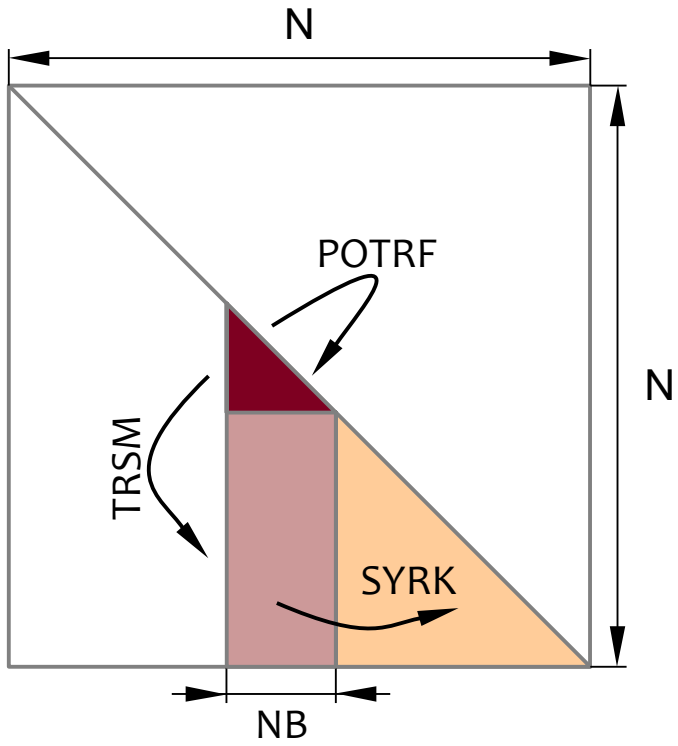
interleaved layouts



Batched Cholesky

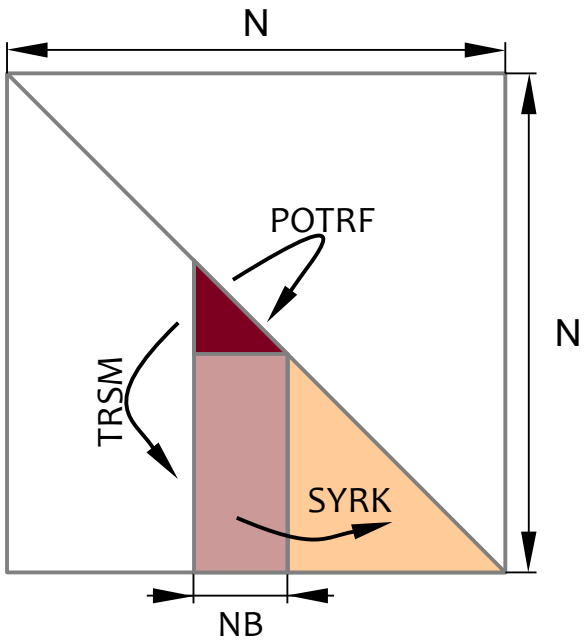
algorithmic blocking

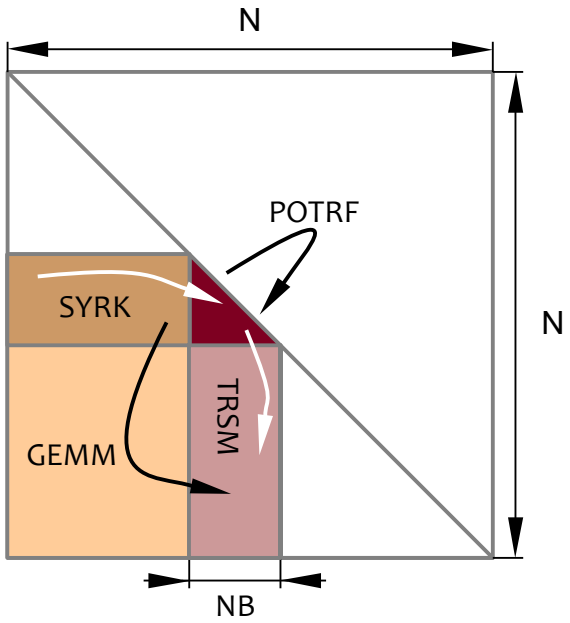


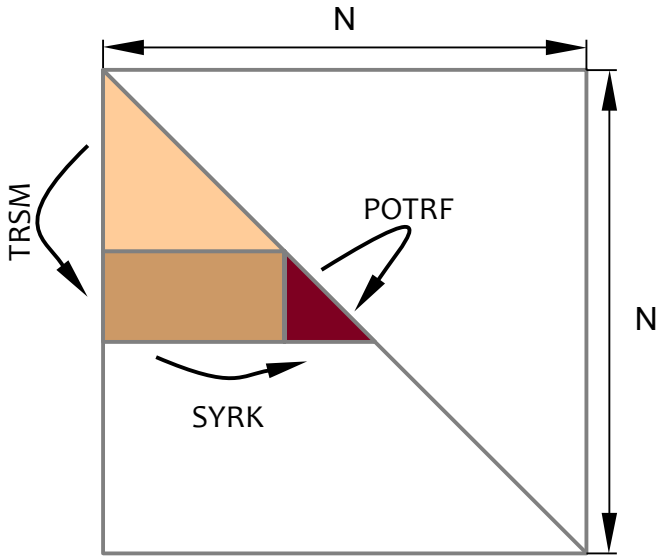


Batched Cholesky

algorithmic variants

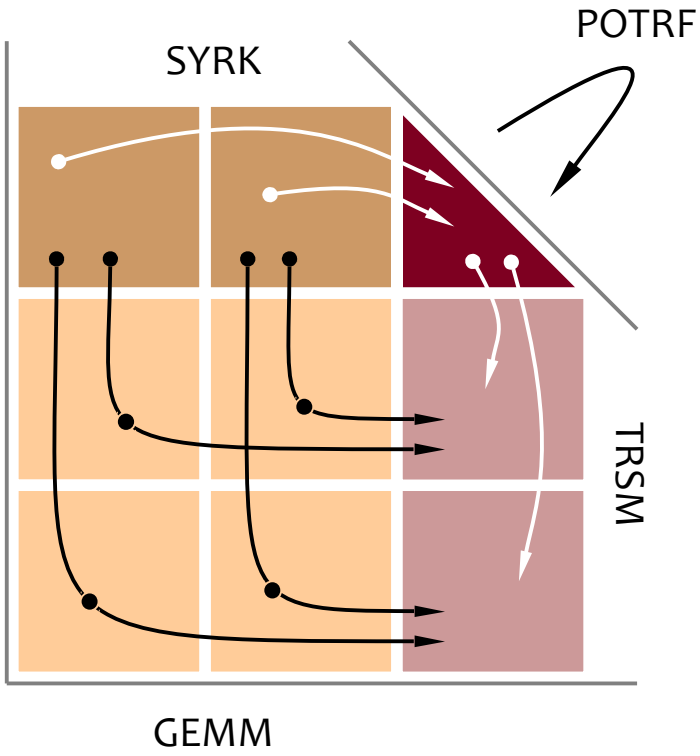






Batched Cholesky

tiling



Batched Cholesky

tuning parameters

Algorithmic Variant

- right-looking
- left-looking
- top-looking

Tiling Factor (NB)

Chunking

- no chunking
- chunking
  - chunking factor (DIM)

Unrolling

- partial (tile operation only)
- full (the entire routine)

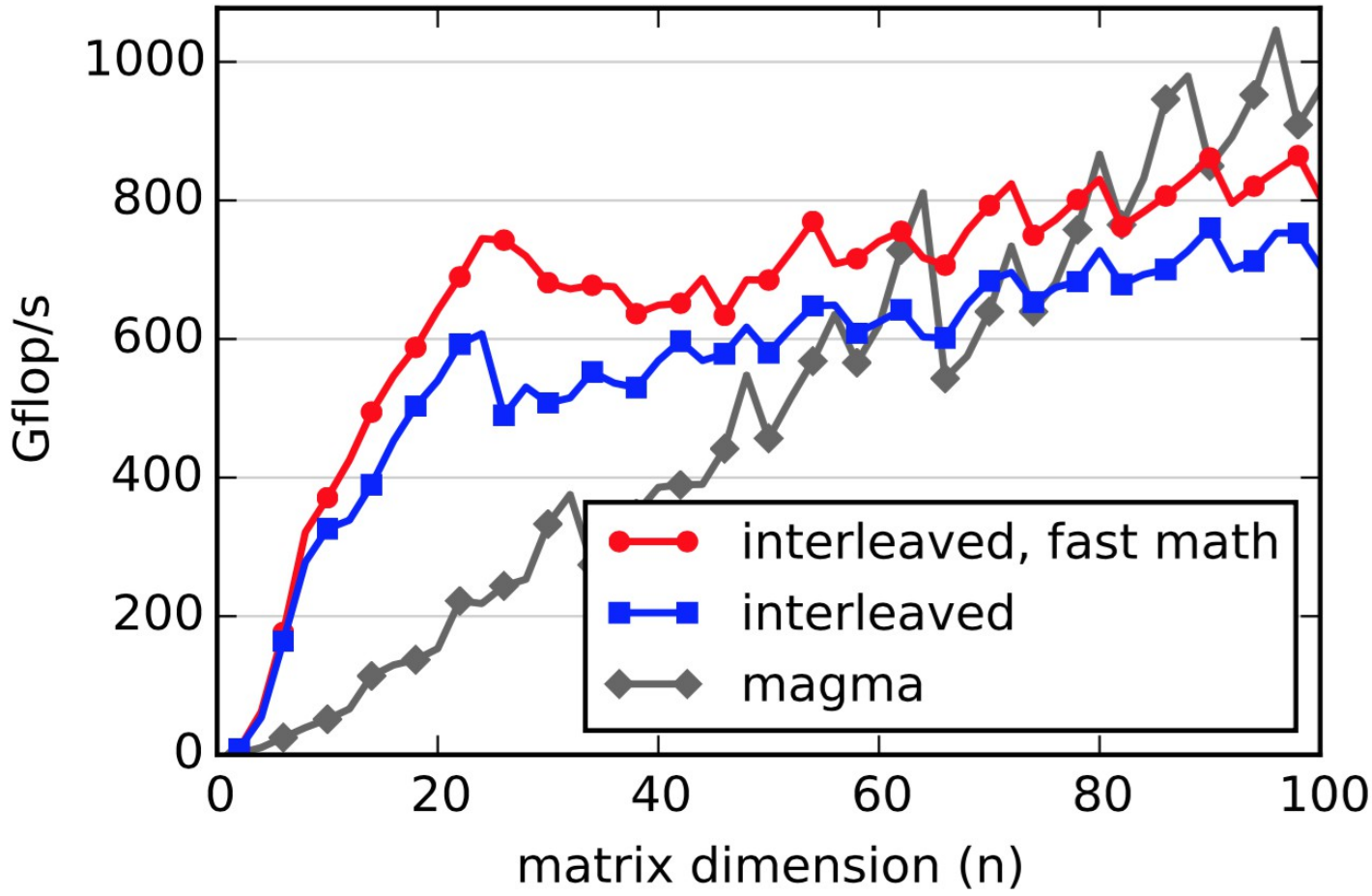
target: NVIDIA Pascal GPU

span sizes up to 100

~14,000 runs

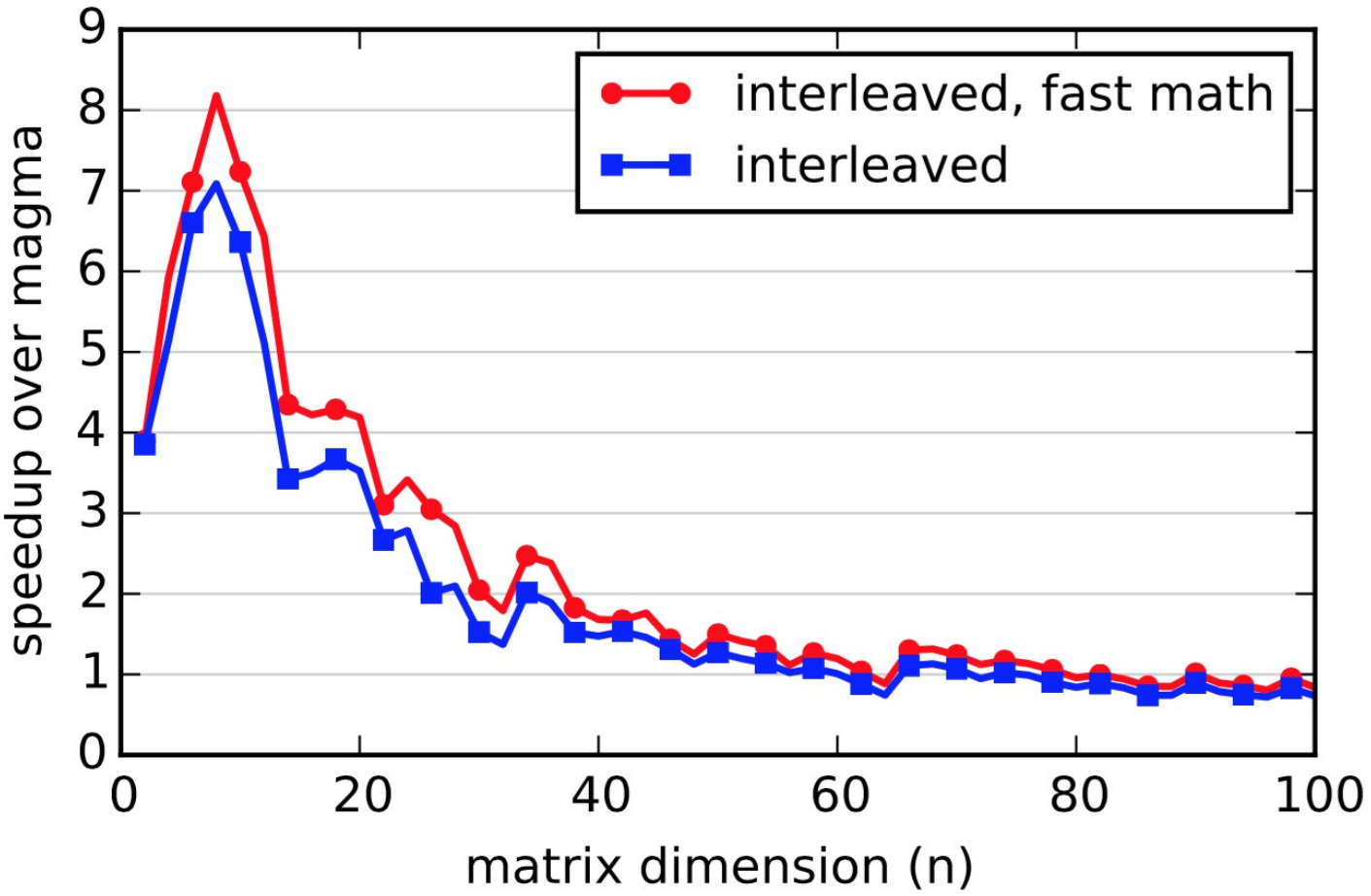
Batched Cholesky

overall performance



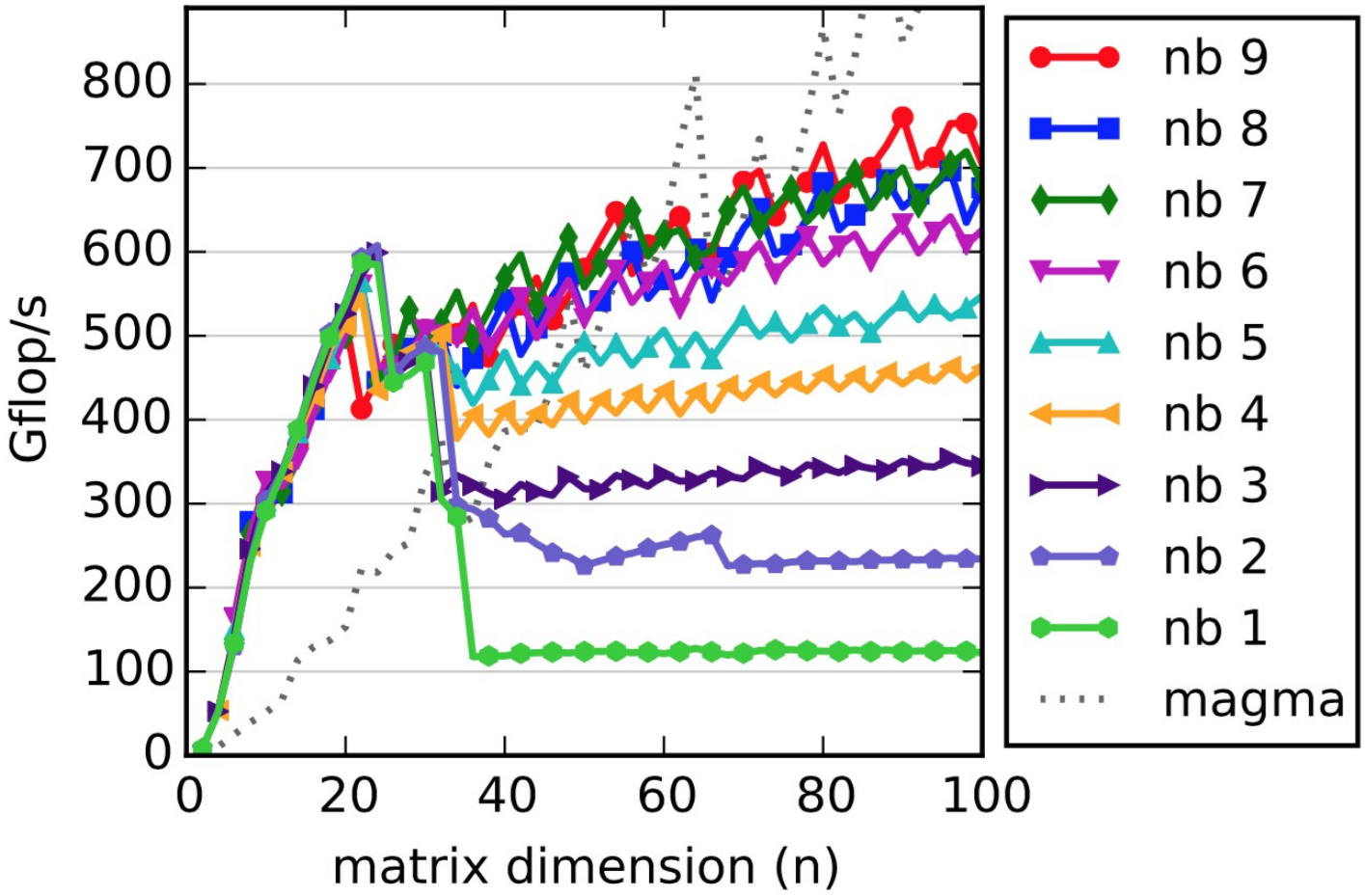
Batched Cholesky

overall performance



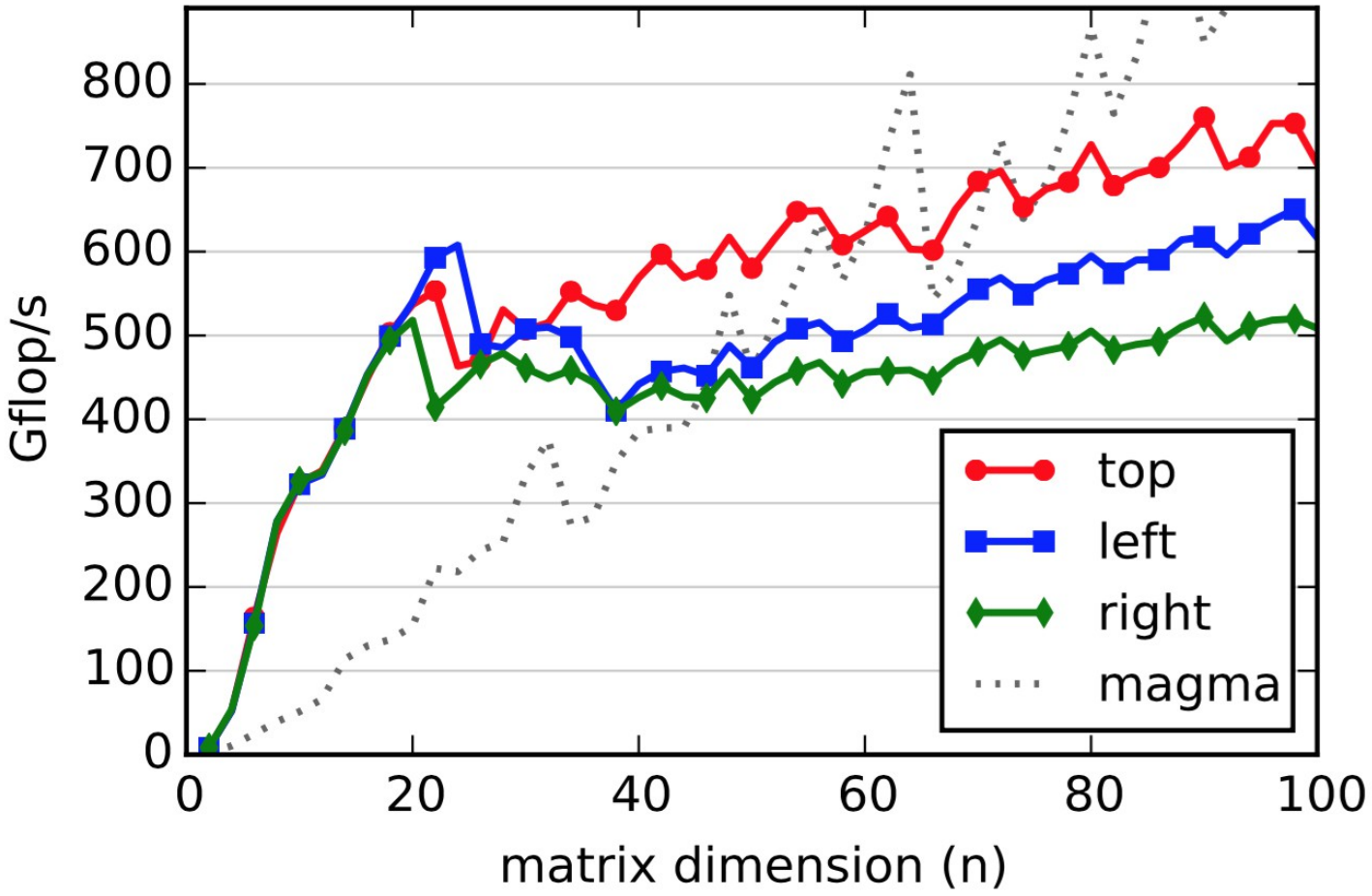
Batched Cholesky

tiling



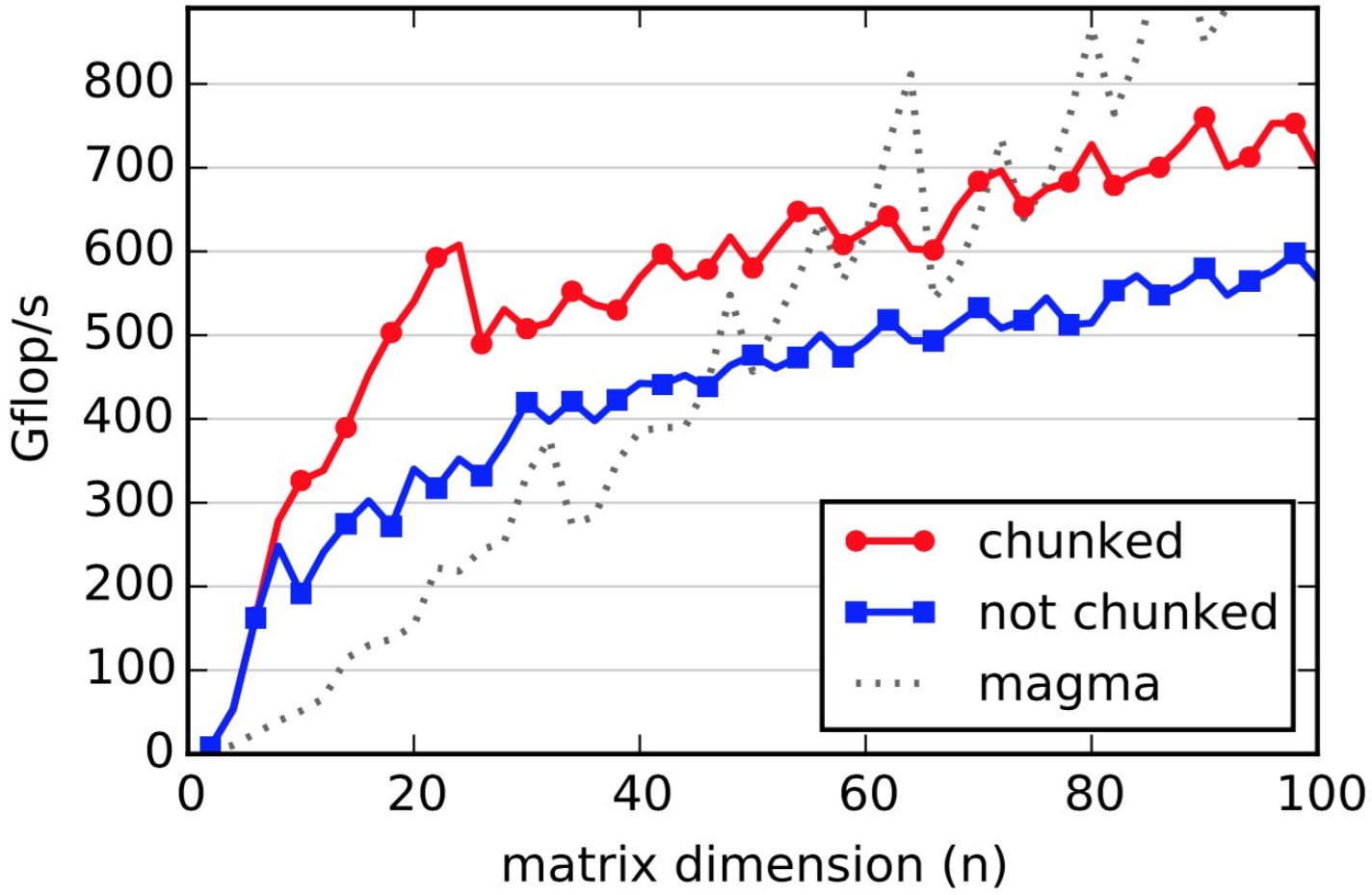
Batched Cholesky

algorithmic variants



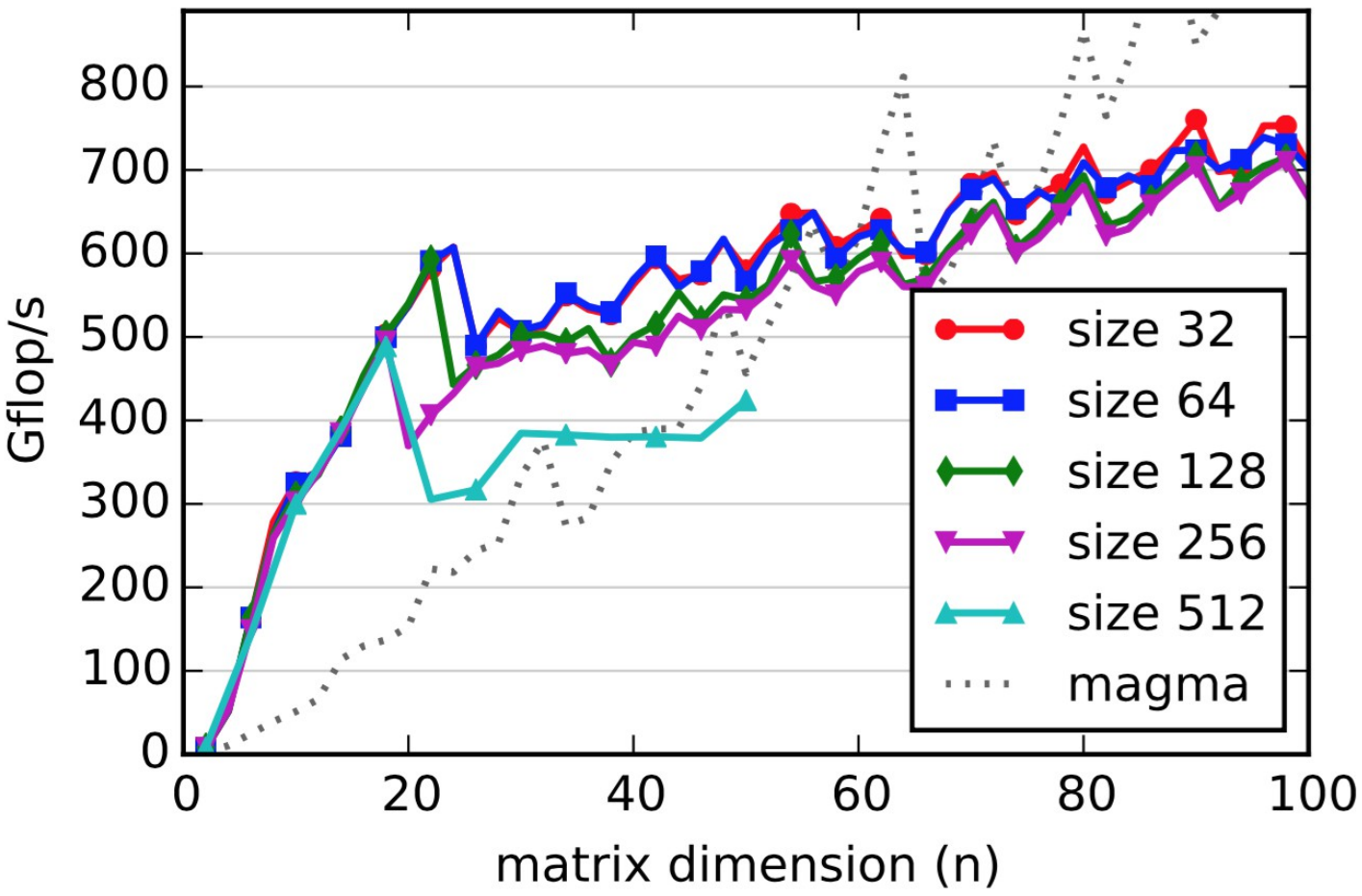
Batched Cholesky

chunking



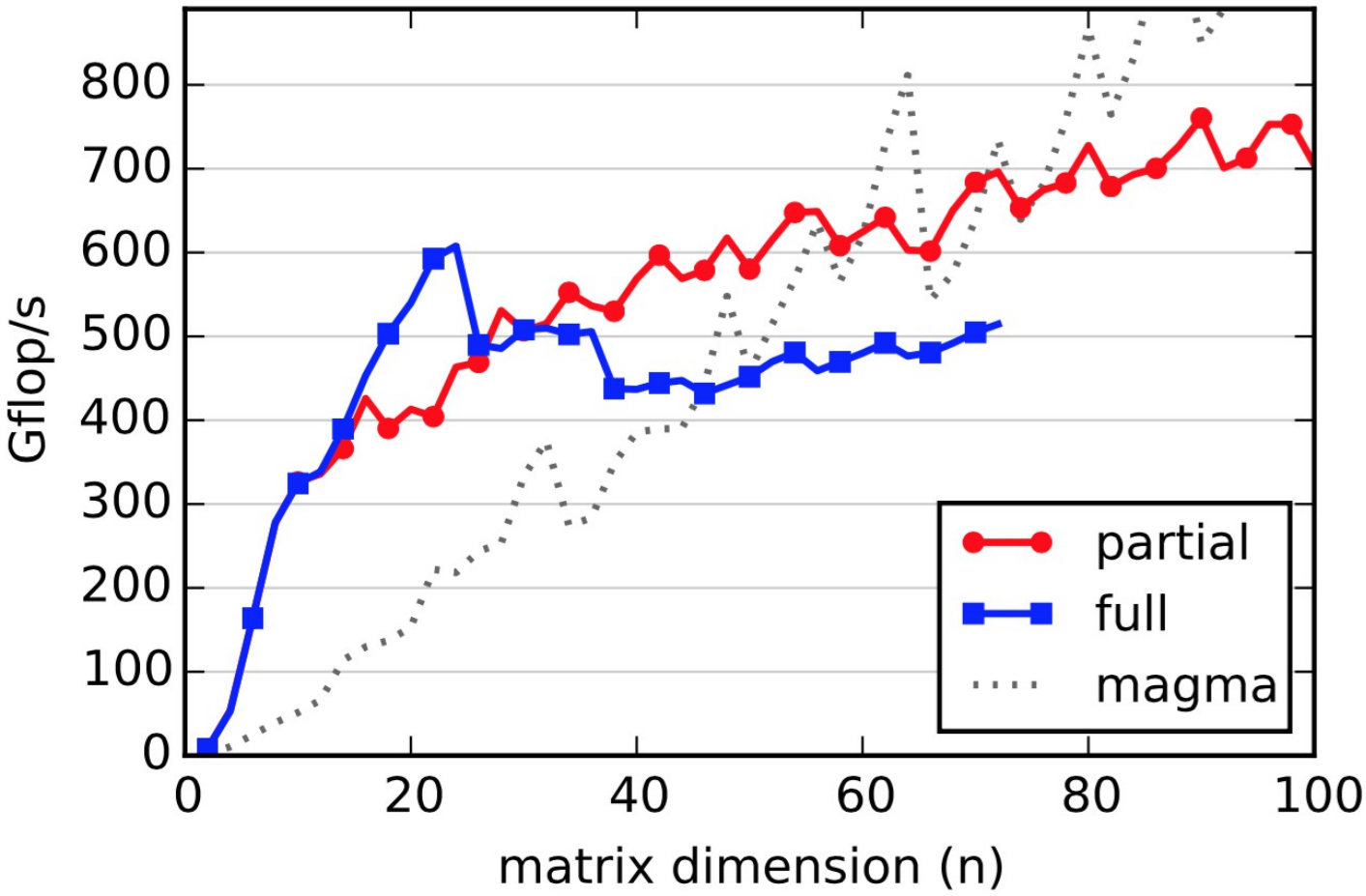
Batched Cholesky

chunk size



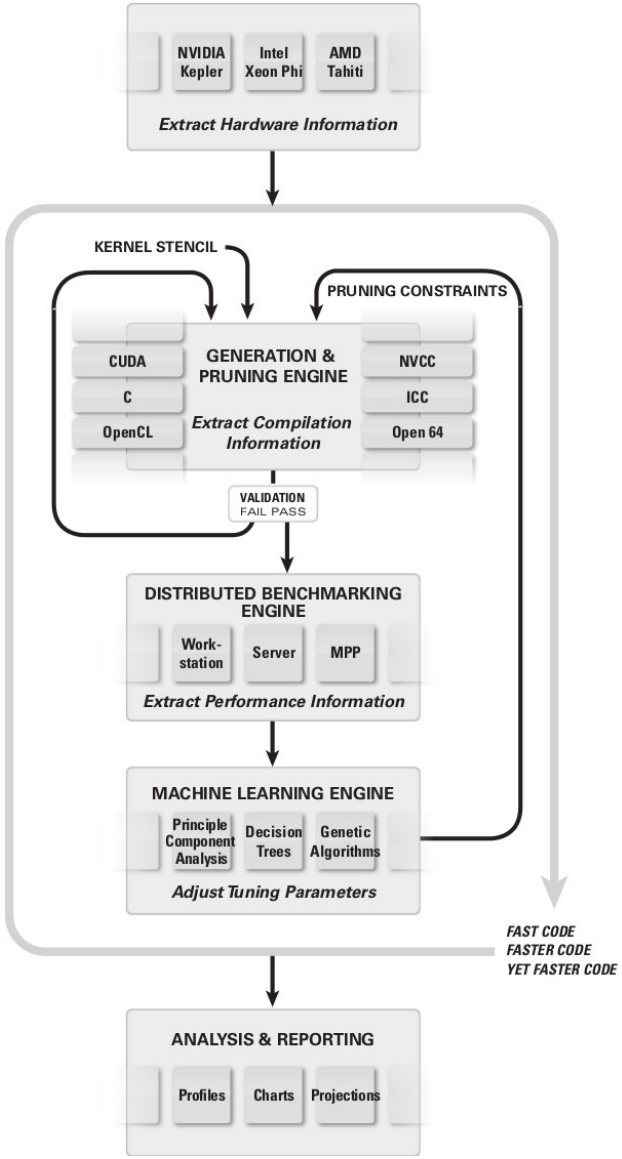
Batched Cholesky

unrolling



BEAST

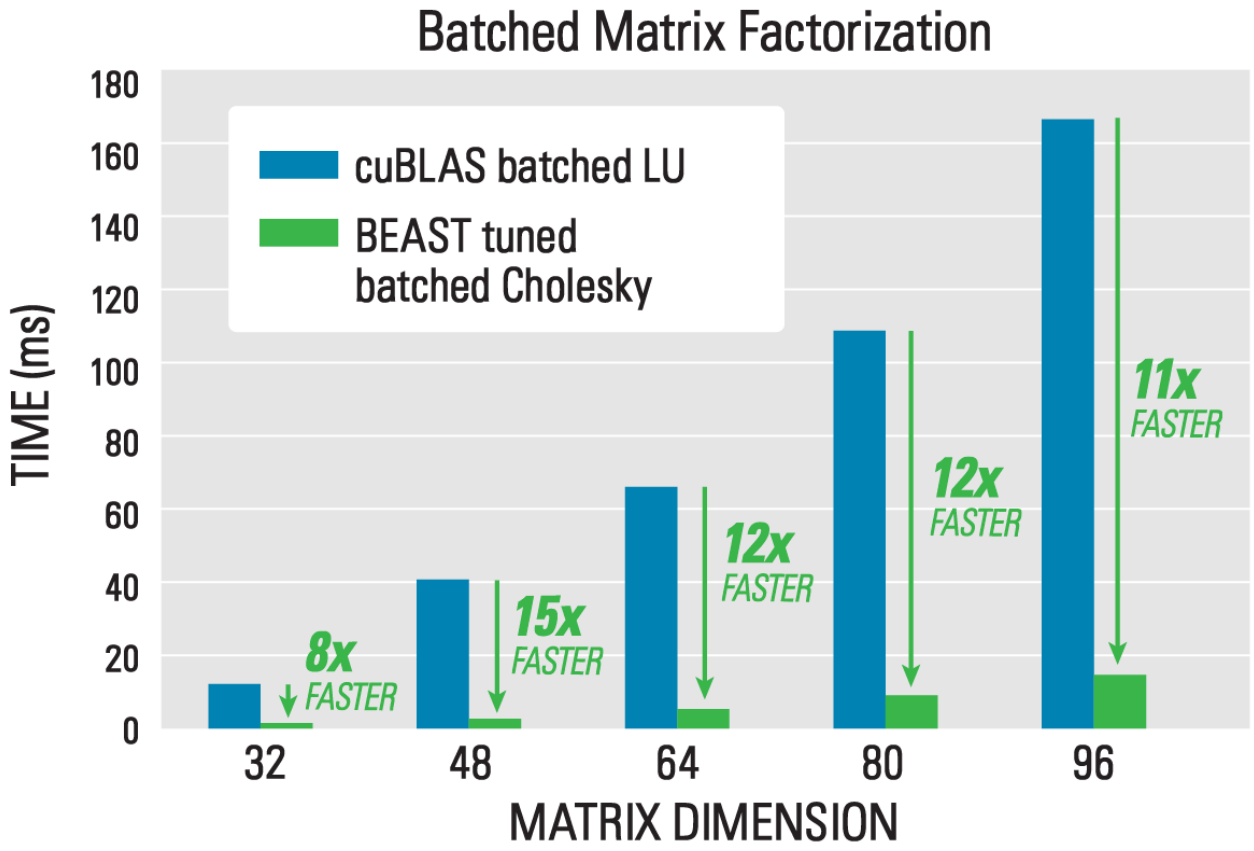
Bench-testing Environment for Automated Software Tuning



- generate large search space
- prune in a smart way
- collect results
- iterate

BEAST

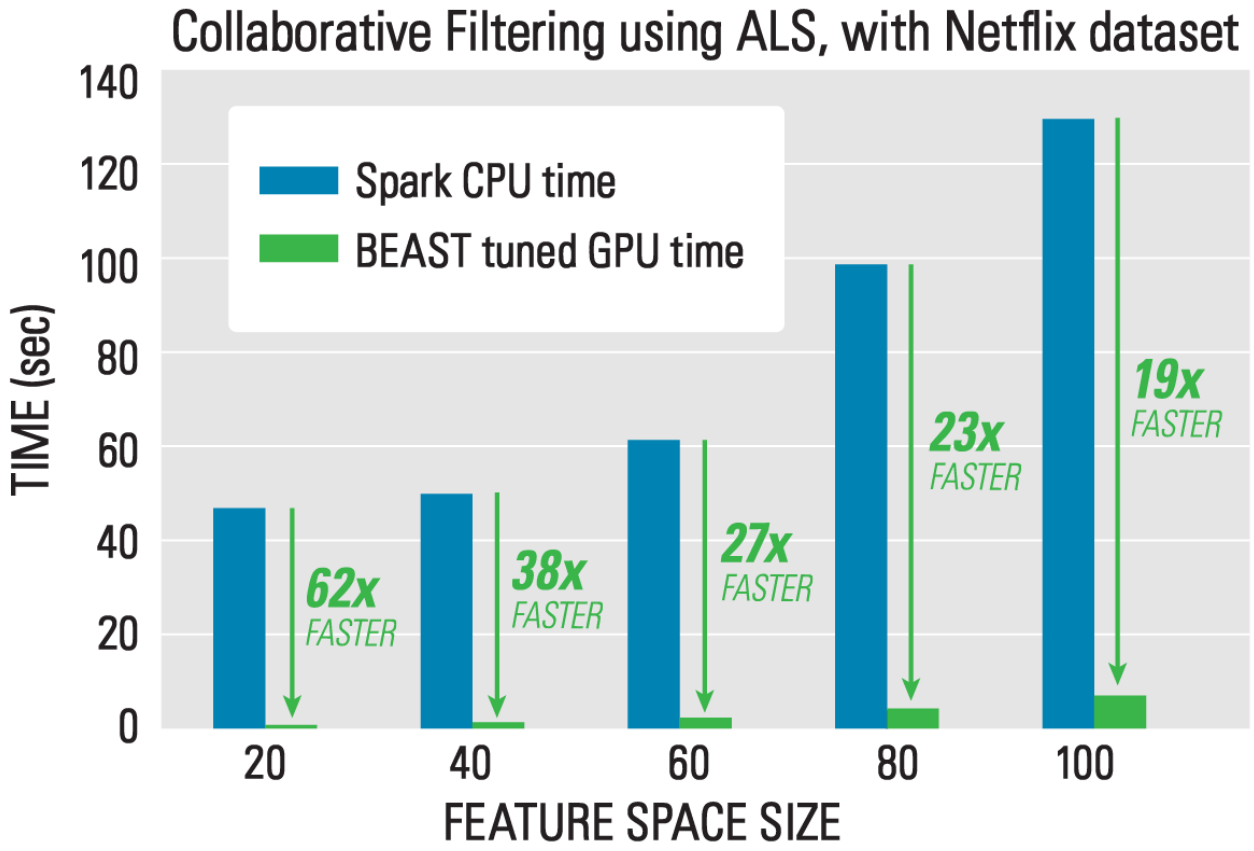
traditional batched Cholesky



Kurzak, Jakub, Hartwig Anzt, Mark Gates, and Jack Dongarra.
"Implementation and tuning of batched Cholesky factorization and solve for NVIDIA GPUs."
IEEE Transactions on Parallel and Distributed Systems 27, no. 7 (2016): 2036-2048.

BEAST

alternating least squares



Gates, Mark, Hartwig Anzt, Jakub Kurzak, and Jack Dongarra.
"Accelerating collaborative filtering using concepts from high performance computing."
In Big Data (Big Data), 2015 IEEE International Conference on, pp. 667-676. IEEE, 2015.

BEAST

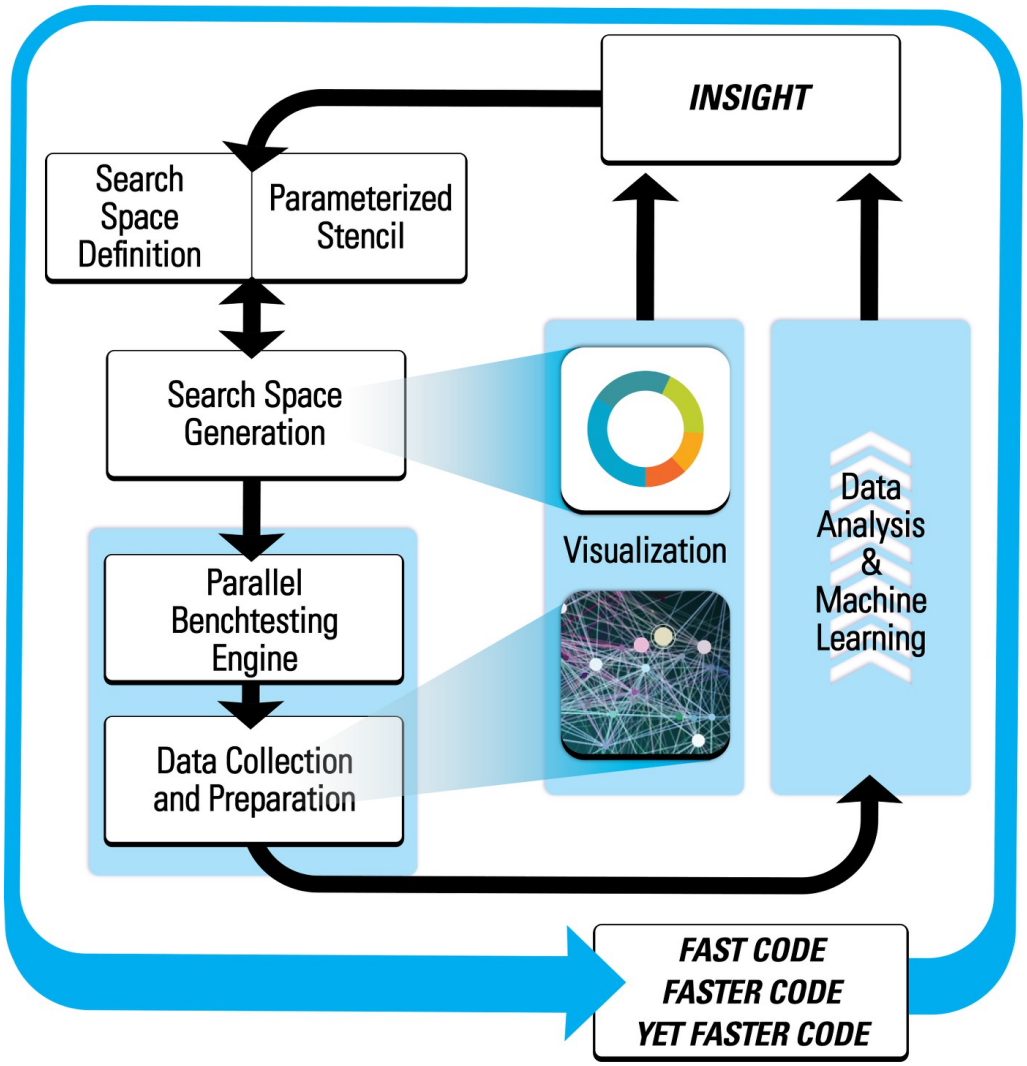
LANAI: LANguage for Autotuning Infrastructure

```
dim_m = range(1, max_threads_dim_x+1)
dim_n = range(1, max_threads_dim_y+1)
@iterator
def blk_m(dim_m):
  return range(dim_m, max_threads_dim_x+1, dim_m)
@iterator
def blk_n(dim_n):
  return range(dim_n, max_threads_dim_y+1, dim_n)
blk_k = range(1, min(max_threads_dim_x, max_threads_dim_y)+1)
@iterator
def dim_vec(arithmetic, precision):
  if arithmetic == "double":
    if precision == "real":
      return range (1, 3)
    else:
      return 1
  else:
    if precision == "real":
      return range(1, 5, 3)
    else:
      return range(1, 3)
```

Luszczek, Piotr, Mark Gates, Jakub Kurzak, Anthony Danalis, and Jack Dongarra.
"Search space generation and pruning system for autotuners."
In *Parallel and Distributed Processing Symposium Workshops, 2016 IEEE International*, pp. 1545-1554. IEEE, 2016.

BONSAI

BEAST OpenN Software Autotuning Infrastructure



- **deploy large autotuning sweeps to supercomputers or the Cloud**
  - distributed compilation
  - distributed benchmarking
  - dynamic load balancing
- exhaustive collection of performance data
- postprocessing of performance data
- data analysis
- visualization


ICL UT
INNOVATIVE
COMPUTING LABORATORY
THE UNIVERSITY of TENNESSEE