Part 1: Avoiding Communication in Eigenvalue and Singular Value Decompositions

Grey Ballard

CS 294/Math 270: Communication-Avoiding Algorithms
UC Berkeley

March 28, 2016

U.S. DEPARTMENT OF ENERGY
National Nuclear Security Administration

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Plan

- Eigenvalue and singular value decompositions of dense matrices
  - want to compute most/all of the values and (possibly) the vectors
  - iterative solvers for large sparse matrices another CA topic

Plan

- Eigenvalue and singular value decompositions of dense matrices
  - want to compute most/all of the values and (possibly) the vectors
  - iterative solvers for large sparse matrices another CA topic
- We're seeking comm-optimal sequential and parallel algorithms
  - known lower bounds (in most cases)
  - standard LAPACK/ScaLAPACK algorithms not optimal

Plan

- Eigenvalue and singular value decompositions of dense matrices
  - want to compute most/all of the values and (possibly) the vectors
  - iterative solvers for large sparse matrices another CA topic
- We're seeking comm-optimal sequential and parallel algorithms
  - known lower bounds (in most cases)
  - standard LAPACK/ScaLAPACK algorithms not optimal
- Many flavors of problems
  - EVD of symmetric matrix
  - EVD of nonsymmetric square matrix
  - SVD of general (rectangular) matrix
  - generalized versions
  - subsets of values/vectors desired

Plan

- Eigenvalue and singular value decompositions of dense matrices
  - want to compute most/all of the values and (possibly) the vectors
  - iterative solvers for large sparse matrices another CA topic
- We're seeking comm-optimal sequential and parallel algorithms
  - known lower bounds (in most cases)
  - standard LAPACK/ScaLAPACK algorithms not optimal
- Many flavors of problems
  - EVD of symmetric matrix
  - EVD of nonsymmetric square matrix
  - SVD of general (rectangular) matrix
  - generalized versions
  - subsets of values/vectors desired
- Overall approach is two-phase reduction to condensed form
  - based on orthogonal (similarity) transformations
  - alternative "divide-and-conquer" approaches another CA topic

Outline

1. EVD/SVD via Reduction to Condensed Form
2. Standard Algorithms
3. Two-Phase Approach
   - Full to Band(-Hessenberg)
   - Band Reduction
4. Open Problems

Canonical Forms for Real Matrices

- EVD of symmetric matrix

$$A = V \Lambda V^T$$

  - Λ is diagonal (eigenvalues) and V is orthogonal (eigenvectors)
- SVD of general rectangular matrix

$$A = U \Sigma V^T$$

  - Σ is diagonal (sing. values) and U, V are orthogonal (sing. vectors)
- EVD of nonsymmetric square matrix

$$A = U T U^T$$

  - Real Schur form: T is almost triangular and U is orthogonal

Computational and Numerical Considerations

- Canonical forms are based on orthogonal transformations
  - algorithms that apply orthogonal (similarity) transformations remain numerically stable
- Computing canonical forms requires iterative algorithms
  - unlike LU or QR decomposition, for example
  - computation depends on numerical values
- Reducing to condensed forms before applying iterative methods saves computation (and usually communication)
  - iterating on dense matrices is expensive
  - there are direct methods for reducing to simpler/sparser forms

Condensed Forms for Real Matrices

- EVD of symmetric matrix

$$A = QTQ^T \qquad \leftarrow \qquad A = V\Lambda V^T$$

  - T is tridiagonal and Q is orthogonal
- SVD of general rectangular matrix

$$A = Q_U B Q_V^T \qquad \leftarrow \qquad A = U\Sigma V^T$$

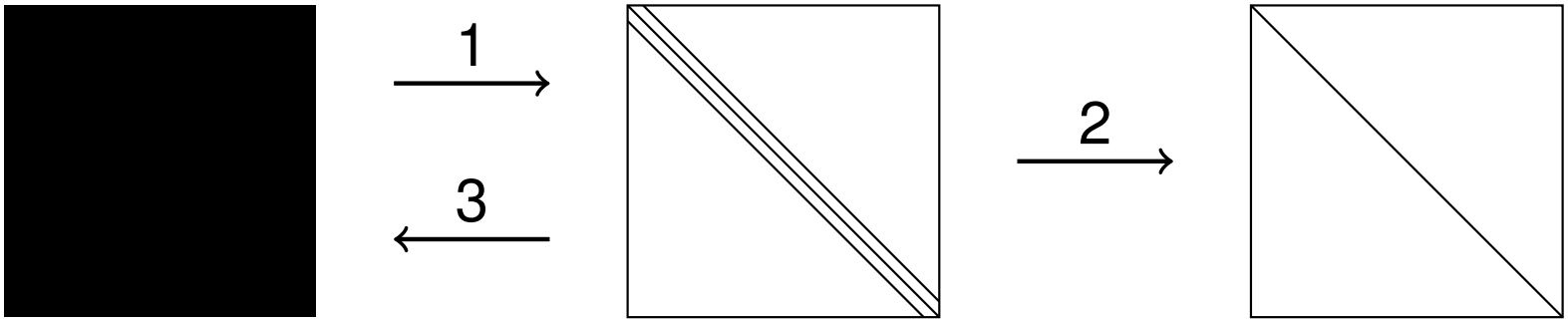
  - B is bidiagonal and Q_U, Q_V are orthogonal
- EVD of nonsymmetric square matrix

$$A = QHQ^T \qquad \leftarrow \qquad A = UTU^T$$

  - H is Hessenberg and Q is orthogonal

Symmetric EVD via Tridiagonalization

We typically compute the symmetric EVD with 3 steps:



1. Reduction-to-tridiagonal via orthogonal similarity transformations

$$A = QTQ^T$$

2. Solve the symmetric tridiagonal eigenvalue problem

$$T = V\Lambda V^T$$

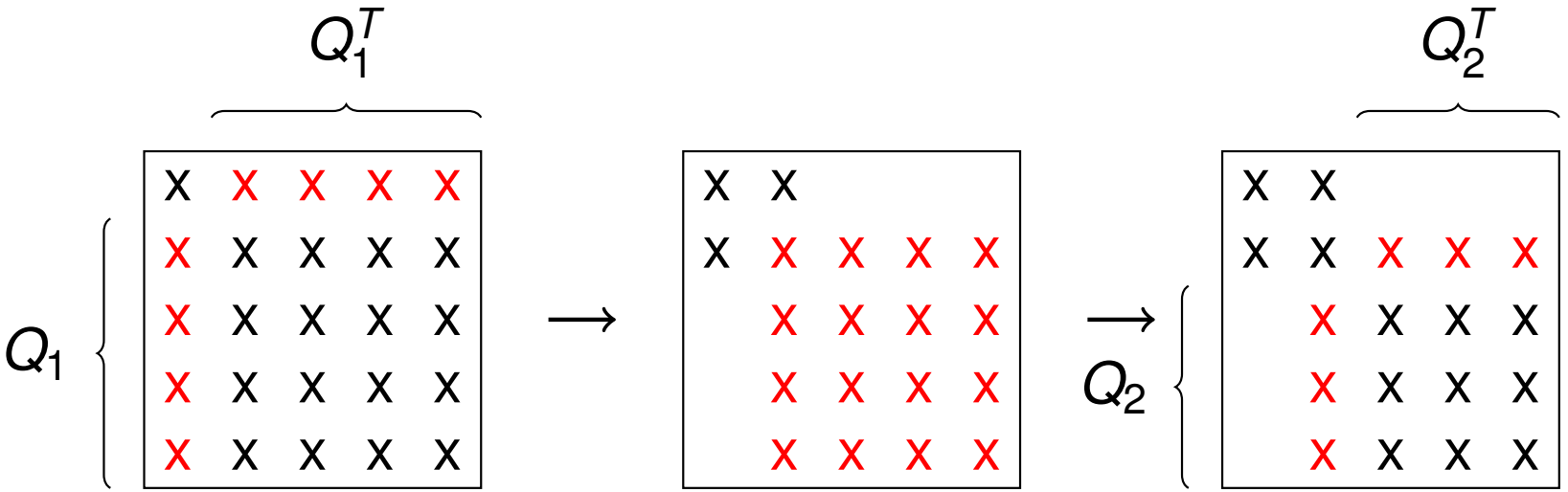
3. Back-transformation of eigenvectors (if desired)

$$A = (QV)\Lambda(QV)^T$$

Outline

1. EVD/SVD via Reduction to Condensed Form
2. Standard Algorithms
3. Two-Phase Approach
   - Full to Band(-Hessenberg)
   - Band Reduction
4. Open Problems

Householder Tridiagonalization



For $i = 1$ **to** $n - 2$

1. compute Householder vector y_i to annihilate column i
2. apply two-sided symmetric update

$$\tilde{A} = (I - \tau_i y_i y_i^T) \cdot A \cdot (I - \tau_i y_i y_i^T)$$

cast as symmetric rank-2 update

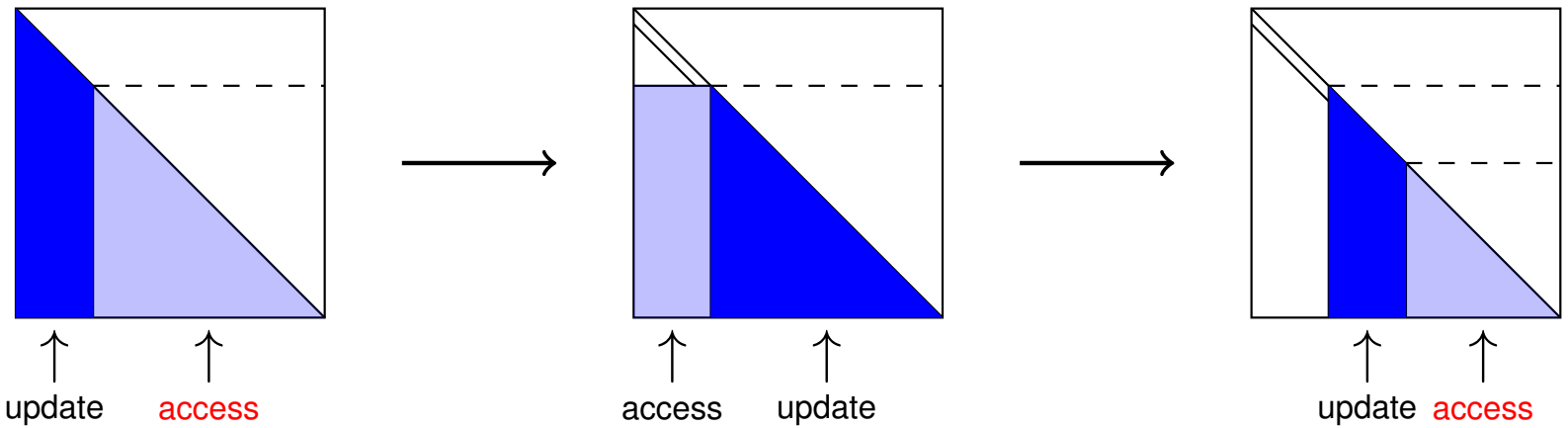
$$\tilde{A} = A - y_i v_i^T - v_i y_i^T \quad \text{(BLAS 2)}$$

End

LAPACK Householder Tridiagonalization

Direct tridiagonalization performed with blocked algorithm:

- panel factorization + (two-sided symmetric) trailing matrix update



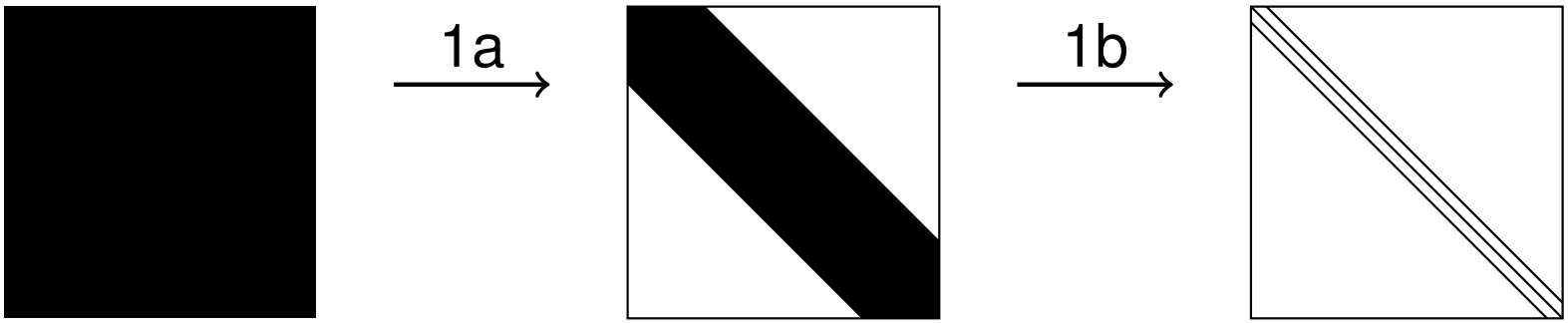
- Panel factorization requires BLAS 2 (matrix-vector) operations
  - total of $O(n^3)$ operations
- Trailing matrix update uses BLAS 3 (matrix-matrix) operations
  - total of $O(n^3)$ operations

Outline

1. EVD/SVD via Reduction to Condensed Form
2. Standard Algorithms
3. Two-Phase Approach
   - Full to Band(-Hessenberg)
   - Band Reduction
4. Open Problems

Two-Phase Tridiagonalization (SBR)

Symmetric tridiagonalization can be done over two (or more) phases in procedure known as Successive Band Reduction (SBR) [BLS00]:

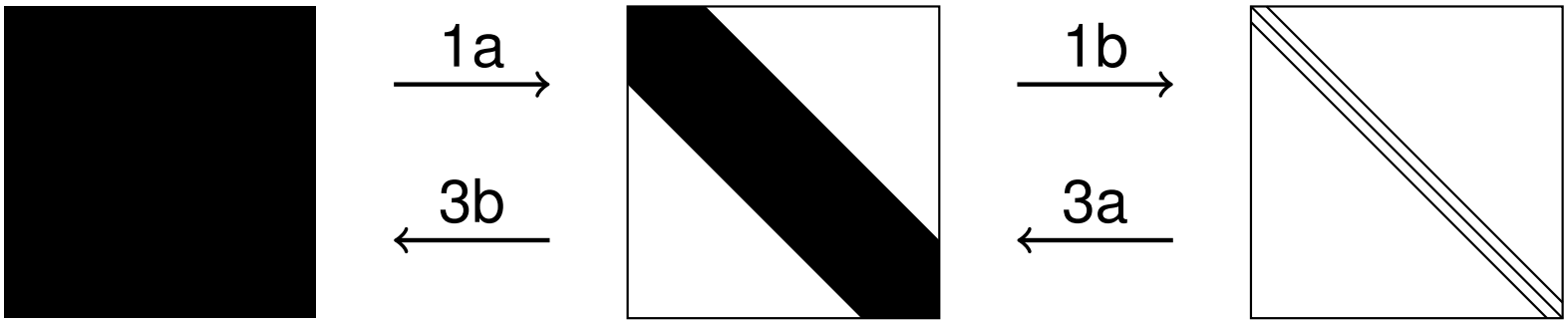


1a Full-to-band via orthogonal similarity transformations

1b Band-to-tridiagonal using bulge-chasing transformations

Two-Phase Tridiagonalization (SBR)

Symmetric tridiagonalization can be done over two (or more) phases in procedure known as Successive Band Reduction (SBR) [BLS00]:



1a Full-to-band via orthogonal similarity transformations

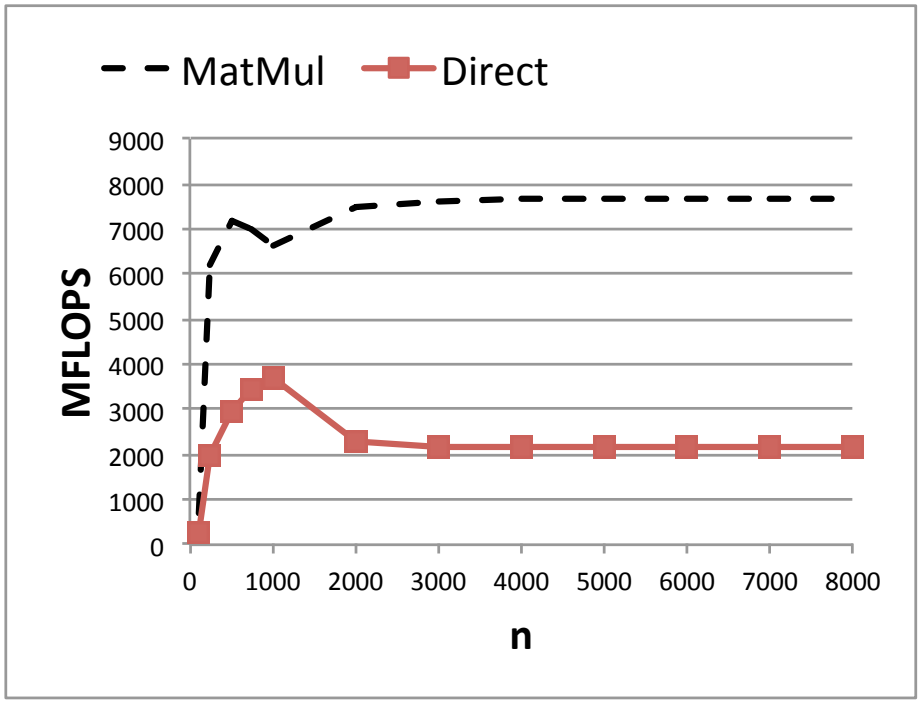
1b Band-to-tridiagonal using bulge-chasing transformations

- Two-phase back-transformation required for eigenvectors

Two-Phase Performance Benefits

Two-phase tridiagonalization avoids communication bottlenecks of direct approach

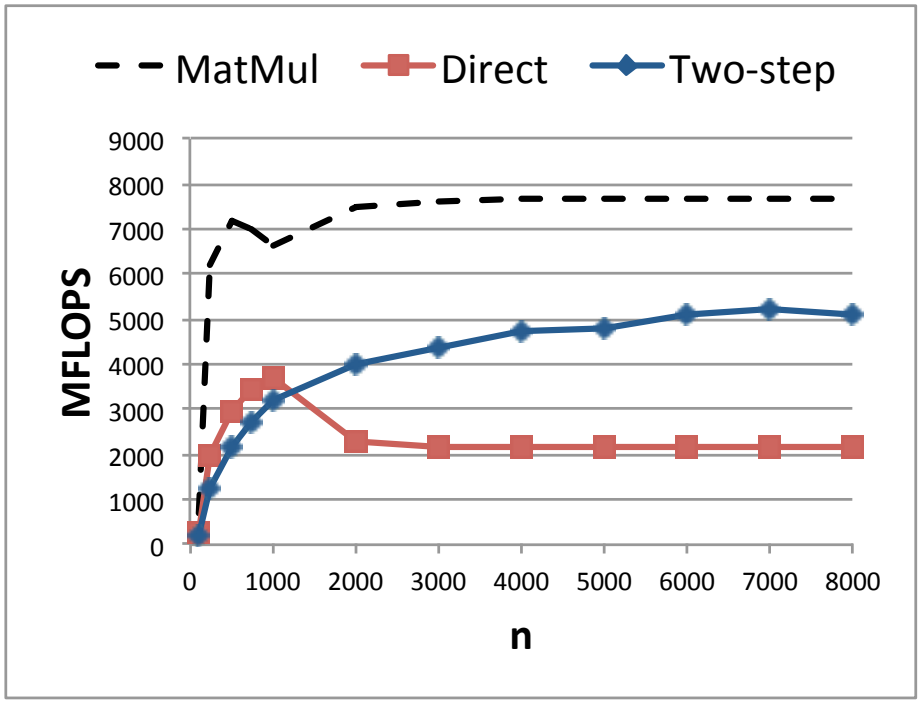
- Sequential performance example
- Direct approach suffers poor cache performance



Two-Phase Performance Benefits

Two-phase tridiagonalization avoids communication bottlenecks of direct approach

- Sequential performance example
- Direct approach suffers poor cache performance
- Two-phase approach already available in MKL



Software for Two-Phase Tridiagonalization

Two-phase tridiagonalization is proven to be effective in practice, achieving better performance than direct tridiagonalization

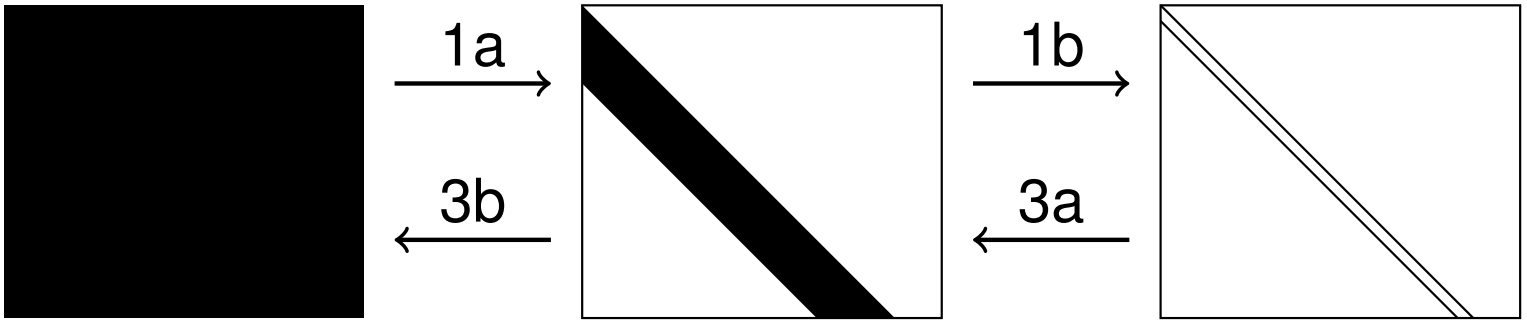
- despite requiring more flops for eigenvectors

- Sequential
  - Successive Band Reduction [BLS00], Intel MKL
- Multicore
  - PLASMA [LLD11], CA-SBR [BDK12]
- GPU
  - MAGMA [HSG+13], Eigen-G [IYM14]
- Distributed-memory parallel
  - ELPA [MBJ+14], Eigen-Exa [IYM11]

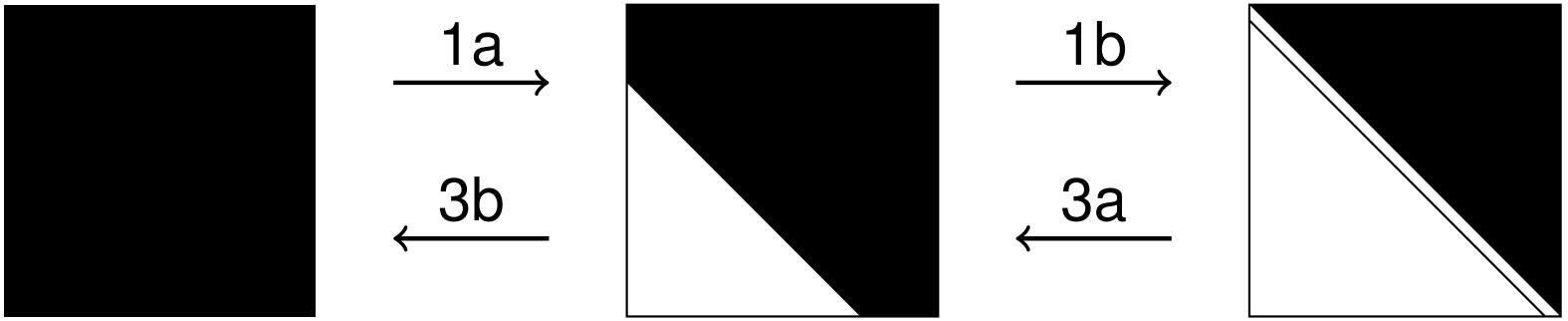
- other work on two-phase reductions for SVD and nonsym. EVD
  - bidiagonal reduction [HKL13], Hessenberg reduction [KK11]

Two-Phase Variants

Bidiagonalization can be done over two phases in a similar procedure:



As can reduction to Hessenberg:

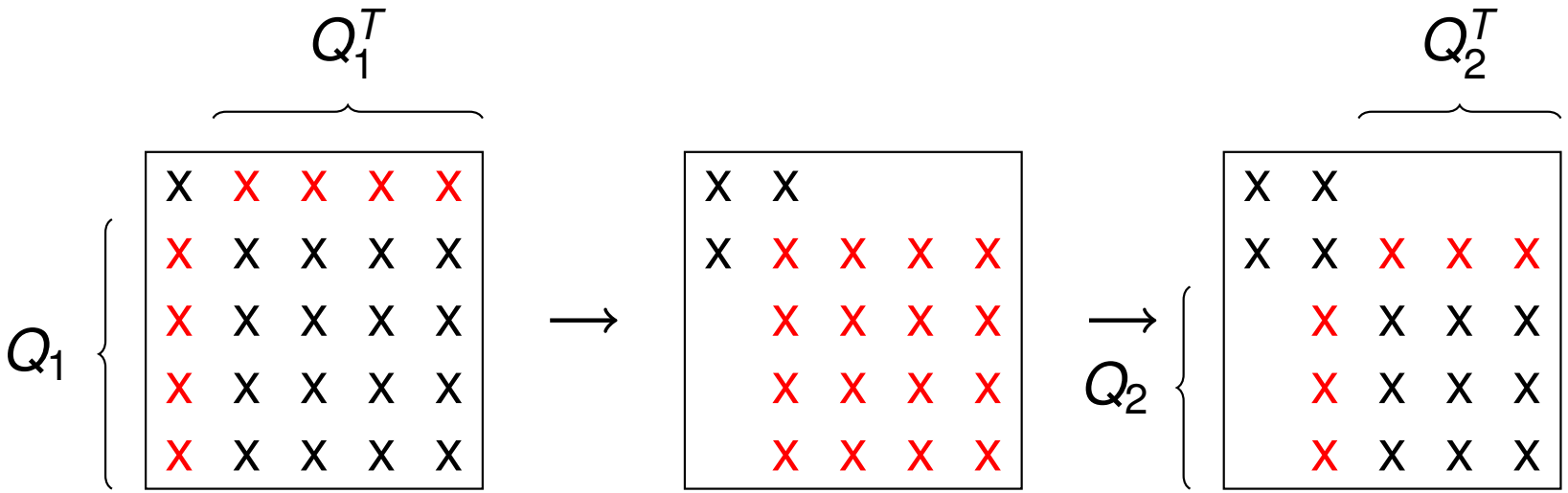


Outline

1. EVD/SVD via Reduction to Condensed Form
2. Standard Algorithms
3. Two-Phase Approach
   - Full to Band(-Hessenberg)
   - Band Reduction
4. Open Problems

Basic Idea

Start with the Householder tridiagonalization algorithm:



Interpret x's as blocks (submatrices) instead of scalars

Blocked Full-to-Band Algorithm

For $i = 1$ **to** $\frac{n}{b} - 2$

1. QR factorization to generate Y_i and annihilate block column i
2. apply two-sided symmetric update

$$\tilde{A} = (I - Y_i T_i Y_i^T) \cdot A \cdot (I - Y_i T_i^T Y_i^T)$$

cast as symmetric rank-$2b$ update

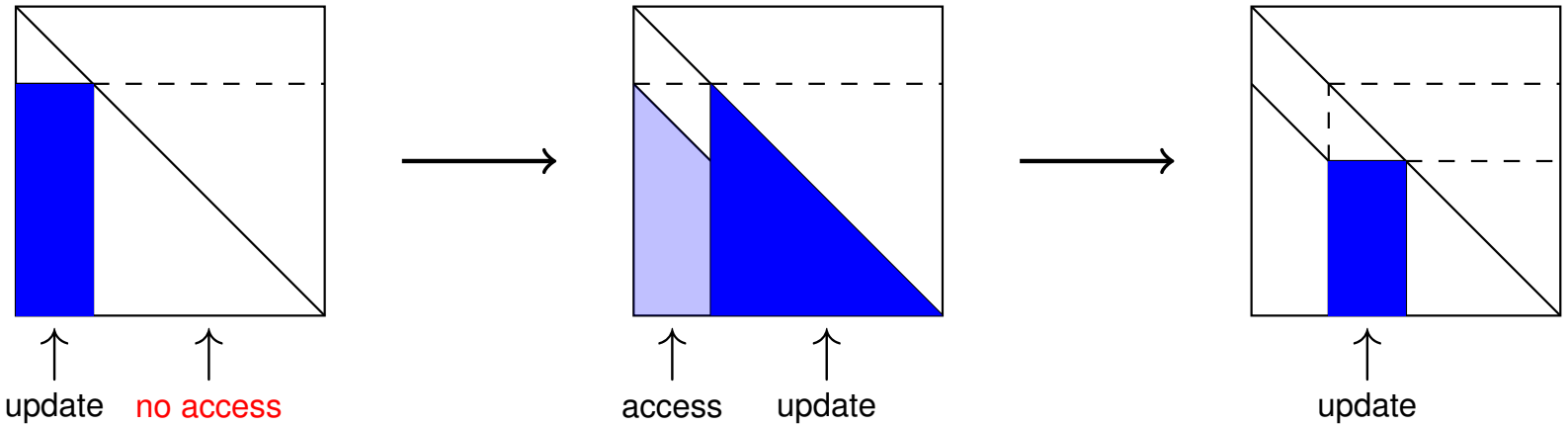
$$\tilde{A} = A - Y_i V_i^T - V_i Y_i^T$$

End

Blocked Full-to-Band Algorithm

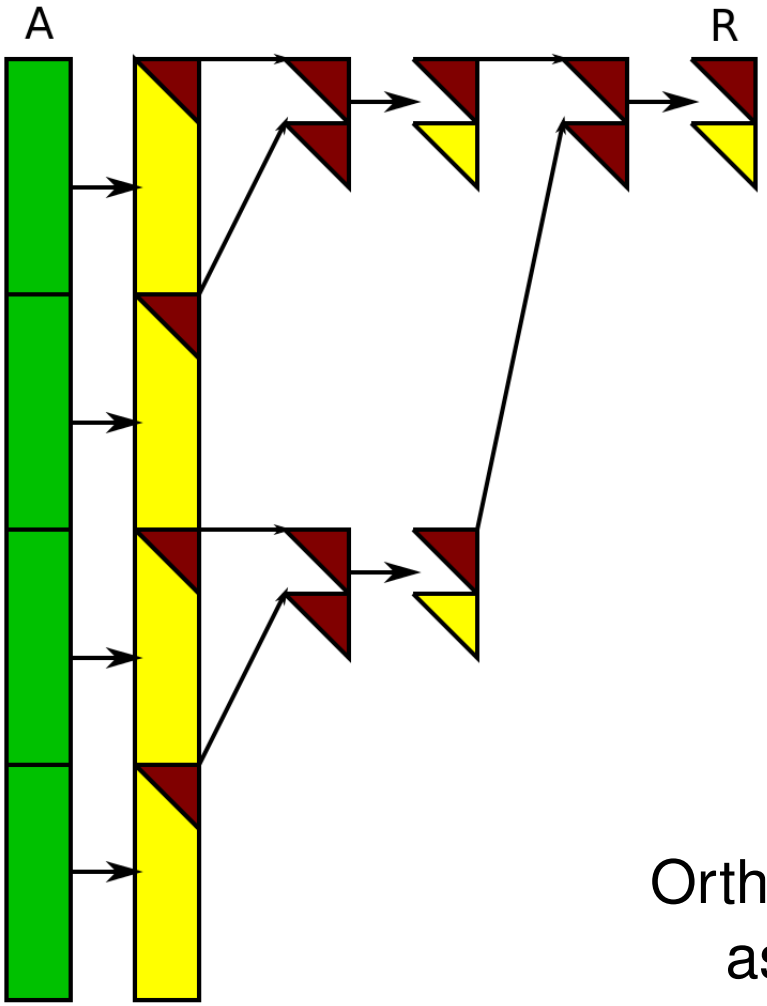
Full-to-band also performed with blocked algorithm:

- panel factorization + (two-sided symmetric) trailing matrix update



- Panel factorization is tall-skinny QR factorization
  - total of $O(n^2 b)$ operations
- Trailing matrix update uses BLAS 3 (matrix-matrix) operations
  - total of $O(n^3)$ operations

Tall-Skinny QR (TSQR) Algorithm [DGHL12]



Key benefit of TSQR:
one parallel reduction

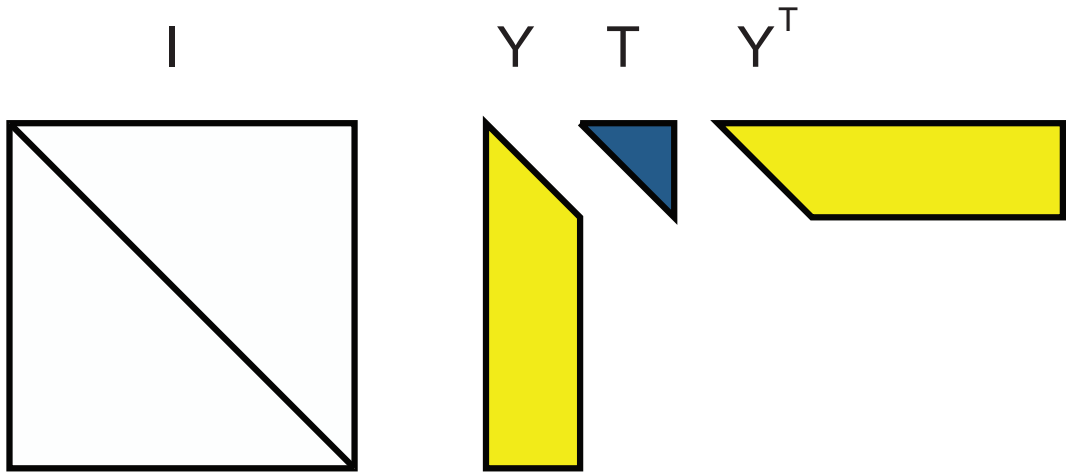
Householder QR:
one reduction *per column*

Orthogonal factor stored implicitly
as tree of Householder vectors

Reconstructing Householder Vectors (TSQR-HR)

1. Perform TSQR
2. Form Q explicitly (tall-skinny orthonormal factor)
3. Perform LU decomposition: $Q - Sgn = LU$
4. Set $Y = L$
5. Set $T = -U \cdot Sgn \cdot Y_1^{-T}$

$$I - YTY^T = I - \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix} [T] \begin{bmatrix} Y_1^T & Y_2^T \end{bmatrix}$$



Leading Order Sequential Costs for Full-to-Band

Panel Factorization	Flops	Words	Messages
Householder QR [BLS00]	$\frac{4}{3}n^3$	$O\left(\frac{n^3}{\sqrt{M}}\right)$	$O\left(\frac{n^3}{M^1}\right)$
TSQR [LLD11]			$O\left(\frac{n^3}{M^{3/2}}\right)$
TSQR-HR [BDK15]			$O\left(\frac{n^3}{M^{3/2}}\right)$
Lower Bound [BDHS11]	—	$\Omega\left(\frac{n^3}{\sqrt{M}}\right)$	$\Omega\left(\frac{n^3}{M^{3/2}}\right)$

Sequential costs of full-to-band reduction of $n \times n$ matrix
to band matrix with bandwidth $b \ll n$,
where M is the size of the fast memory.

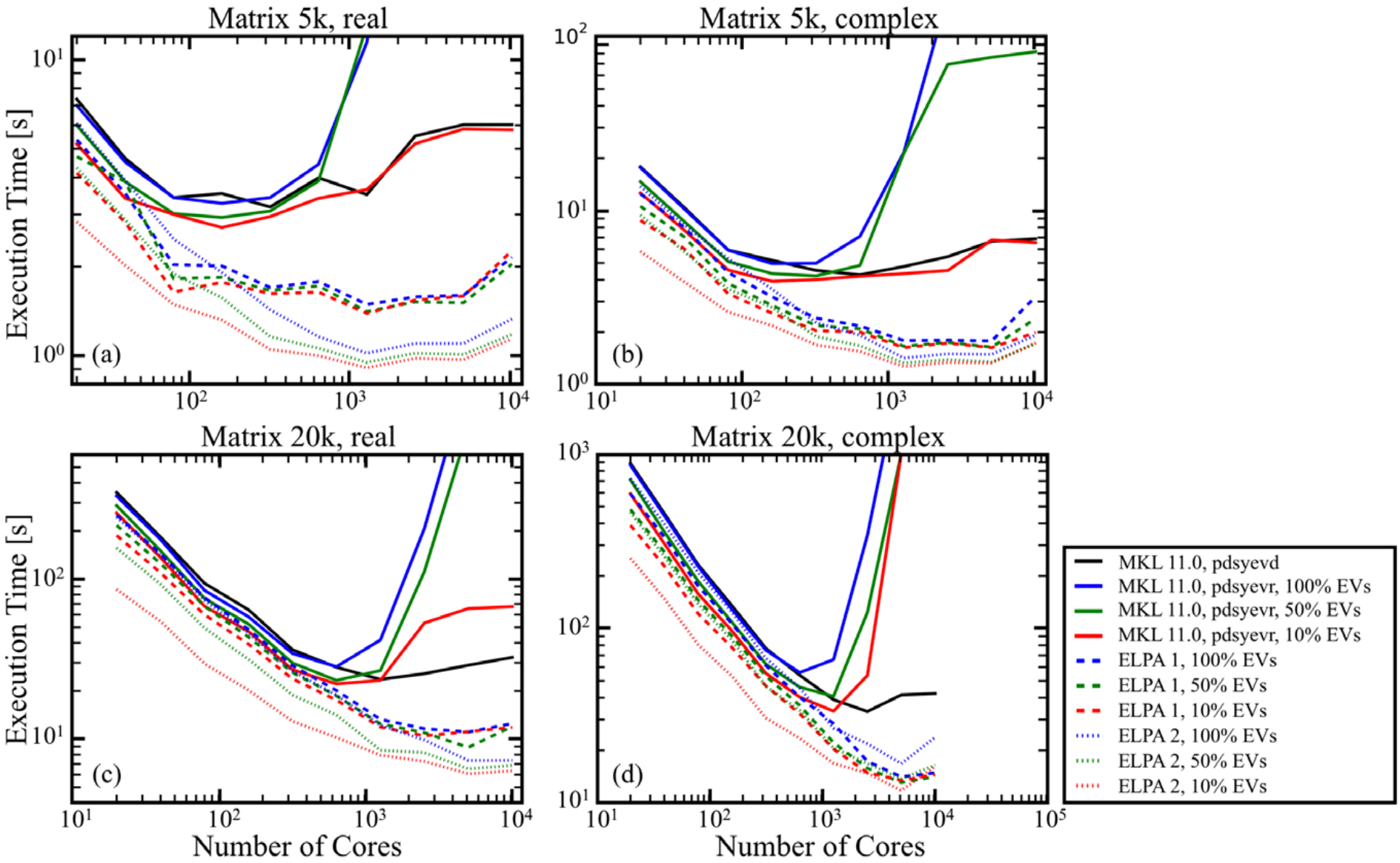
Leading Order Parallel Costs for Full-to-Band

Panel Factorization	Flops	Words	Messages
Householder QR [MBJ+14]	$\frac{4}{3}\frac{n^3}{p}$	$O\left(\frac{n^2}{\sqrt{p}}\right)$	$O\left(n \log p\right)$
TSQR		$O\left(\frac{n^2}{\sqrt{p}} \log p\right)$	$O\left(\sqrt{p} \log^3 p\right)$
TSQR-HR [BDK15]		$O\left(\frac{n^2}{\sqrt{p}}\right)$	$O\left(\sqrt{p} \log^2 p\right)$
Lower Bound* [BDHS11]	—	$\Omega\left(\frac{n^2}{\sqrt{p}}\right)$	$\Omega\left(\sqrt{p}\right)$

Parallel costs of full-to-band reduction of $n \times n$ matrix
to band matrix with bandwidth $b \ll n$
distributed over p processors in 2D fashion.

* assumes local memory restricted to $O(n^2/p)$

ELPA Library Parallel Performance [MBJ+14]



Outline

1. EVD/SVD via Reduction to Condensed Form
2. Standard Algorithms
3. Two-Phase Approach
   - Full to Band(-Hessenberg)
   - Band Reduction
4. Open Problems

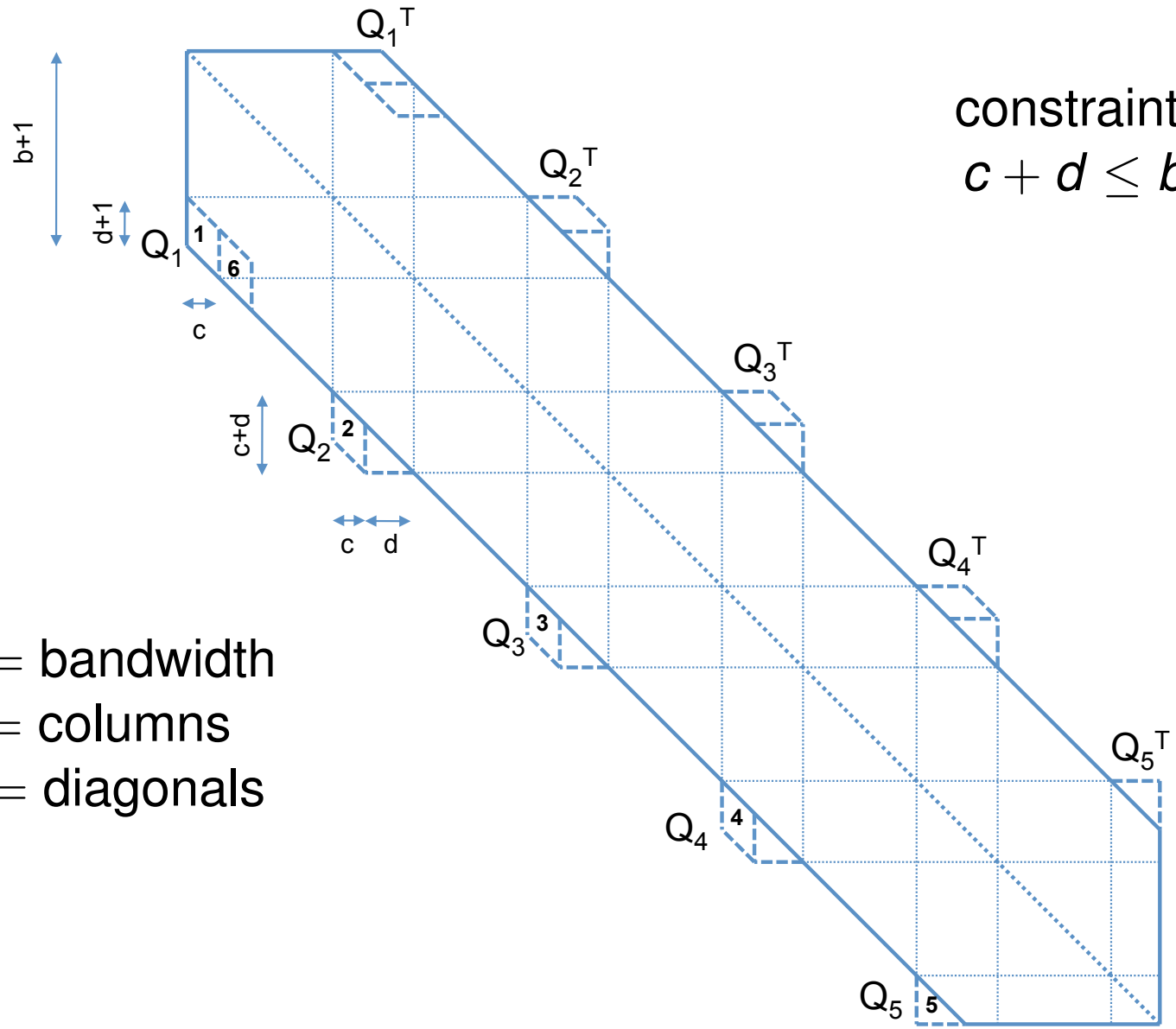
Band-to-Tridiagonal Reduction

Maintaining band structure during orthogonal similarity transformations is trickier than with full-to-band phase

- Annihilating entries within band causes fill-in outside the band
- Bulge-chasing process is required to maintain band structure

- Ideas go back a long way:
  - Rutishauser [Rut63]
  - Schwarz [Sch63]
  - Murata and Horikoshi [MH75]
  - Kaufman [Kau84]
  - Bischof, Lang, and Sun [BLS00]

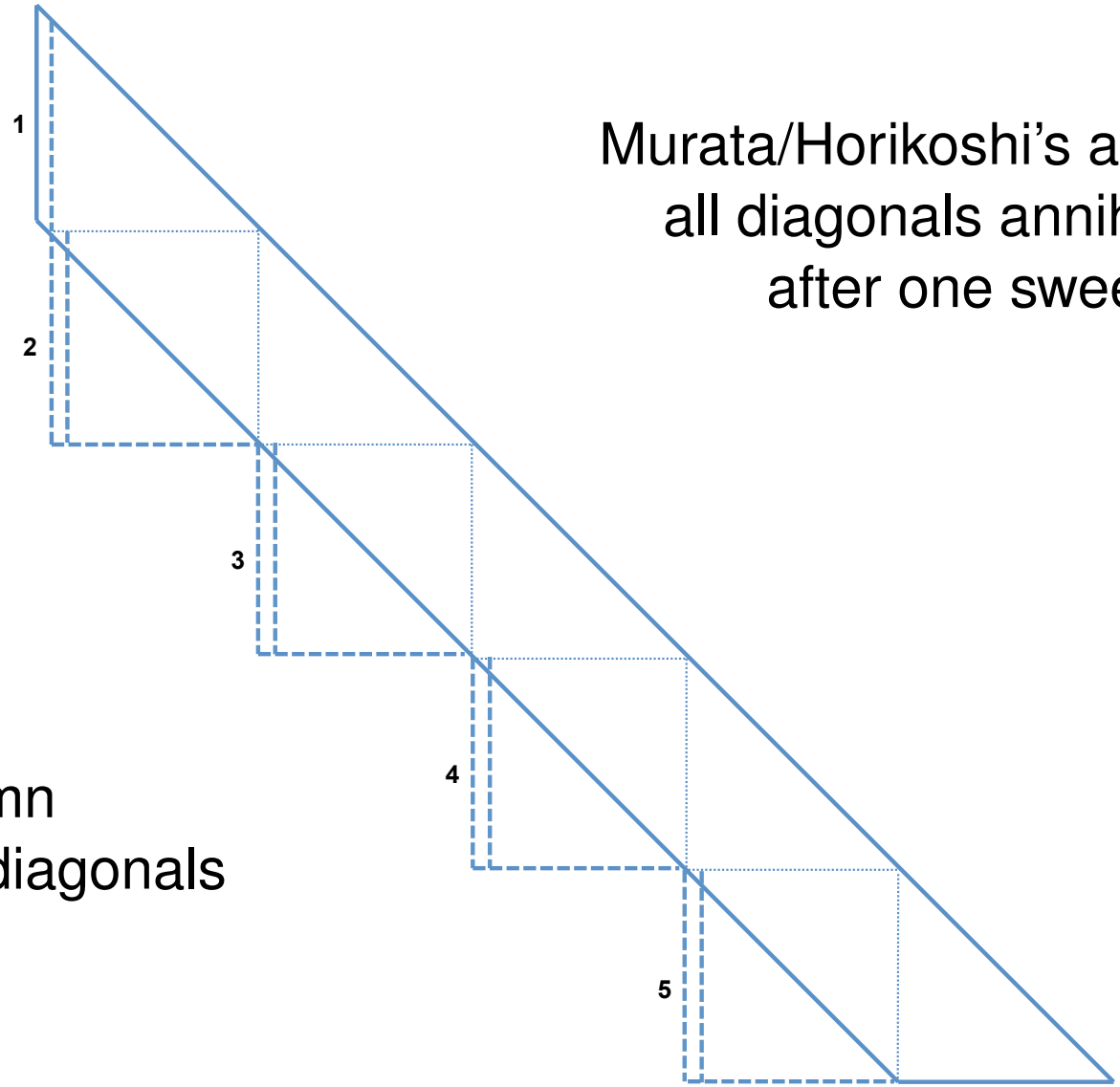
Band-to-Tridiagonal Bulge Chasing



constraint:
$c + d \leq b$

$b =$ bandwidth
$c =$ columns
$d =$ diagonals

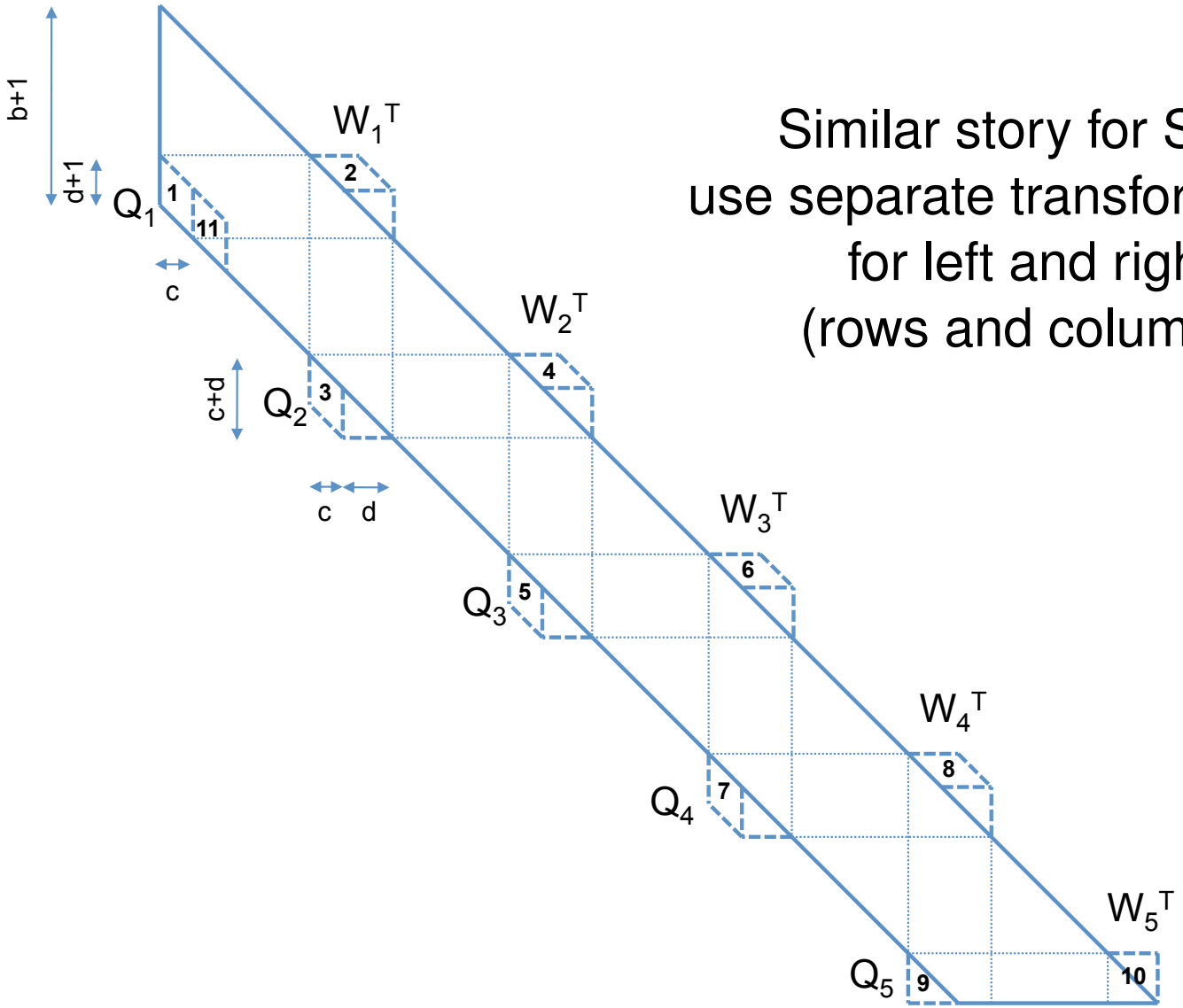
1-Sweep Band-to-Tridiagonal [MH75]



Murata/Horikoshi's algorithm:
all diagonals annihilated
after one sweep

$c = 1$ column
$d = b - 1$ diagonals

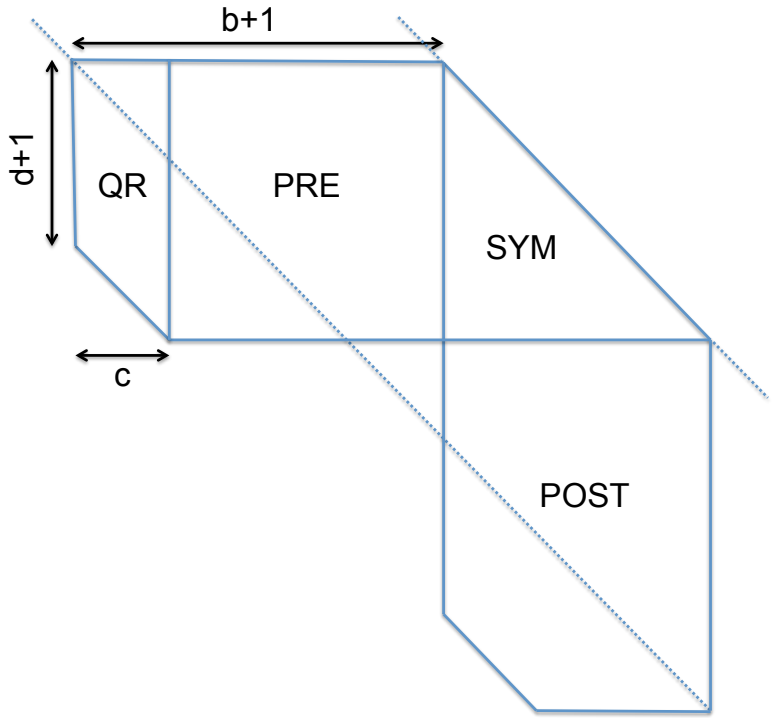
Band-to-Bidiagonal Bulge Chasing

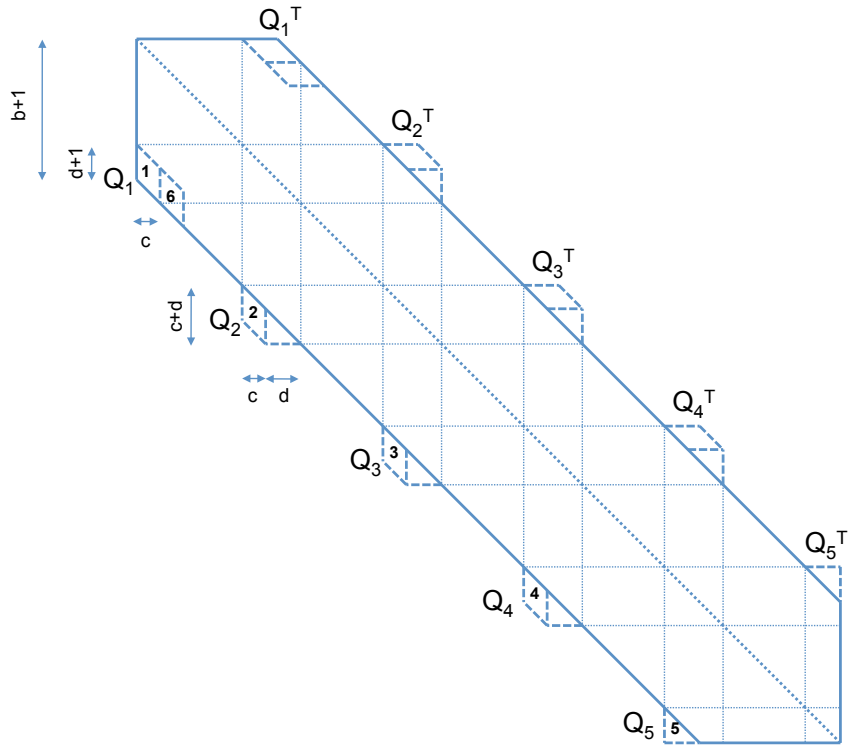


Similar story for SVD:
use separate transformations
for left and right
(rows and columns)

How do we get data re-use?

1. Increase number of columns in parallelogram (c)
   - permits blocking Householder updates: $O(c)$ re-use
   - constraint $c + d \leq b \implies$ trade-off between re-use and progress
   - requires multiple "sweeps"
2. Chase multiple bulges at a time (ω)
   - apply several updates to band while it's in local memory: $O(\omega)$ re-use
   - bulges cannot overlap, need working set to fit in local memory





Communication-Avoiding SBR

CA-SBR balances the two techniques for getting data re-use

Theoretically optimal approach: cut bandwidth in half at every sweep

- $\log b$ sweeps

Sequential and shared-memory implementation exist [BDK12]

- number of sweeps is tuning parameter

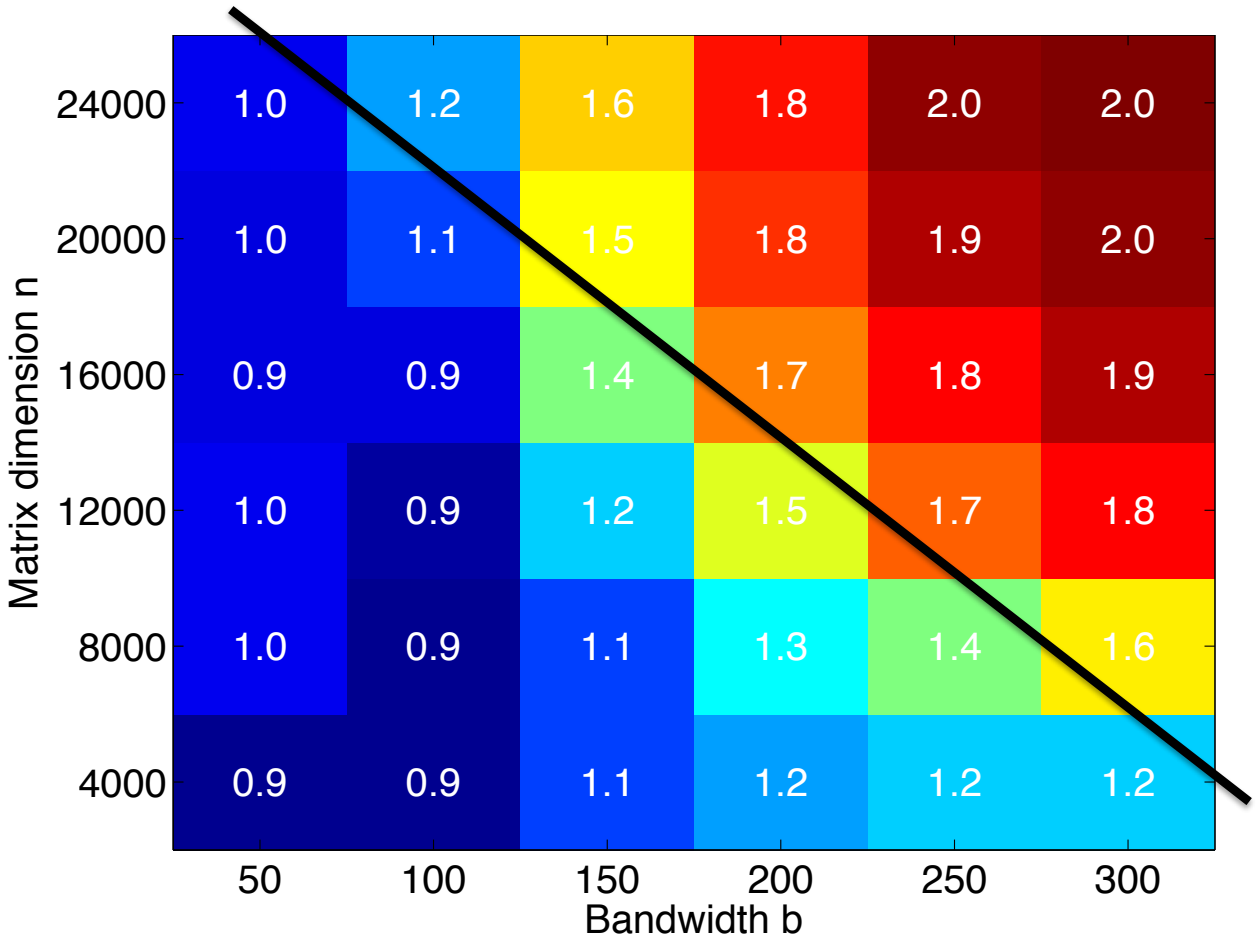
Leading Order Costs for Seq. Band-to-Tridiagonal

Algorithm	Flops	Words	Messages
LAPACK [Kau00]	$4n^2b$	$O(n^2b)$	$O(n^2b)$
CA-SBR	$5n^2b$	$O\left(\frac{n^2b^2}{M}\right)$	$O\left(\frac{n^2b^2}{M^2}\right)$
Lower Bound	—	?	?

Sequential costs of band-to-tridiagonal reduction of $n \times n$ band matrix with bandwidth $b < \sqrt{M}/3$, where M is the size of the fast memory.

Performance Results in Shared Memory

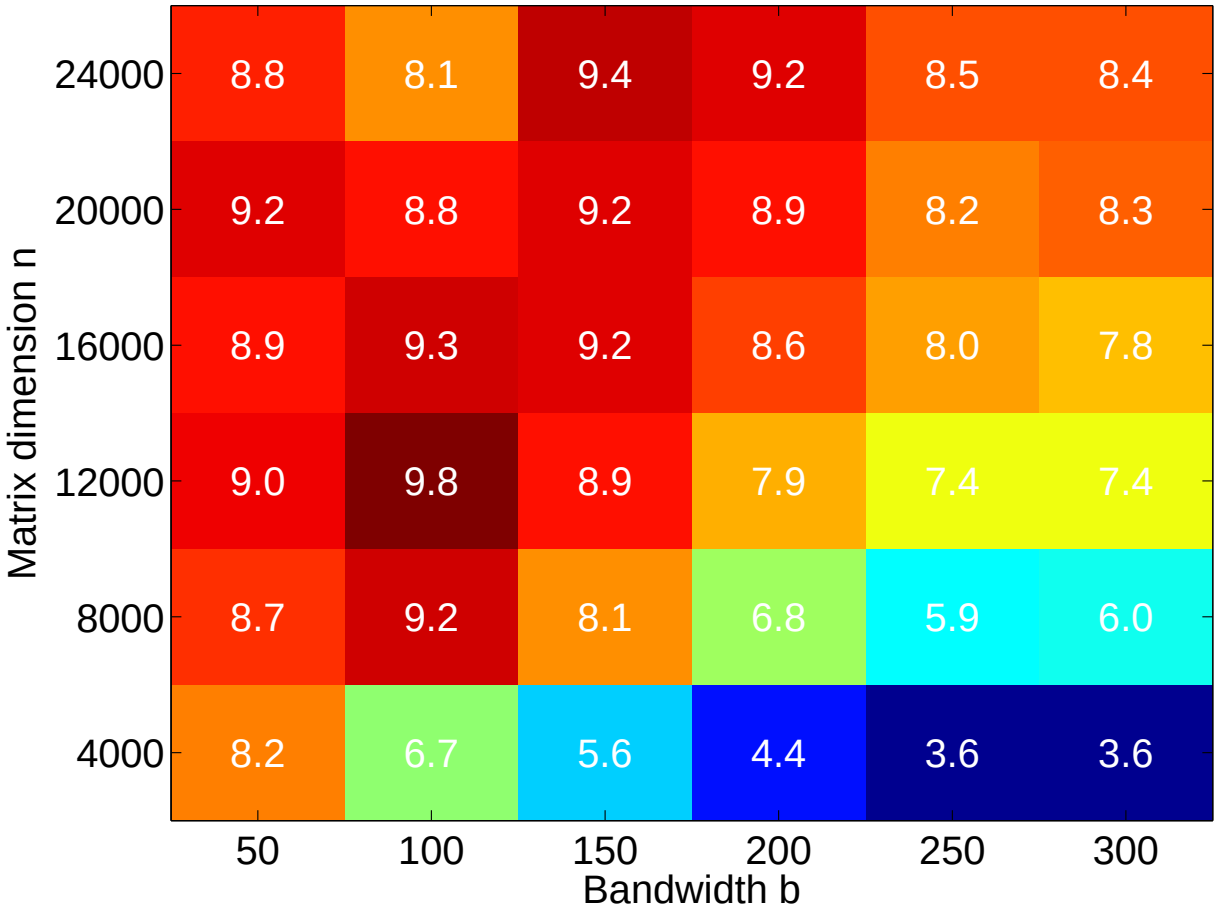
Speedup of sequential CASBR over Intel's Math Kernel Library



Benchmarked on 10-core Intel Westmere [BDK12]

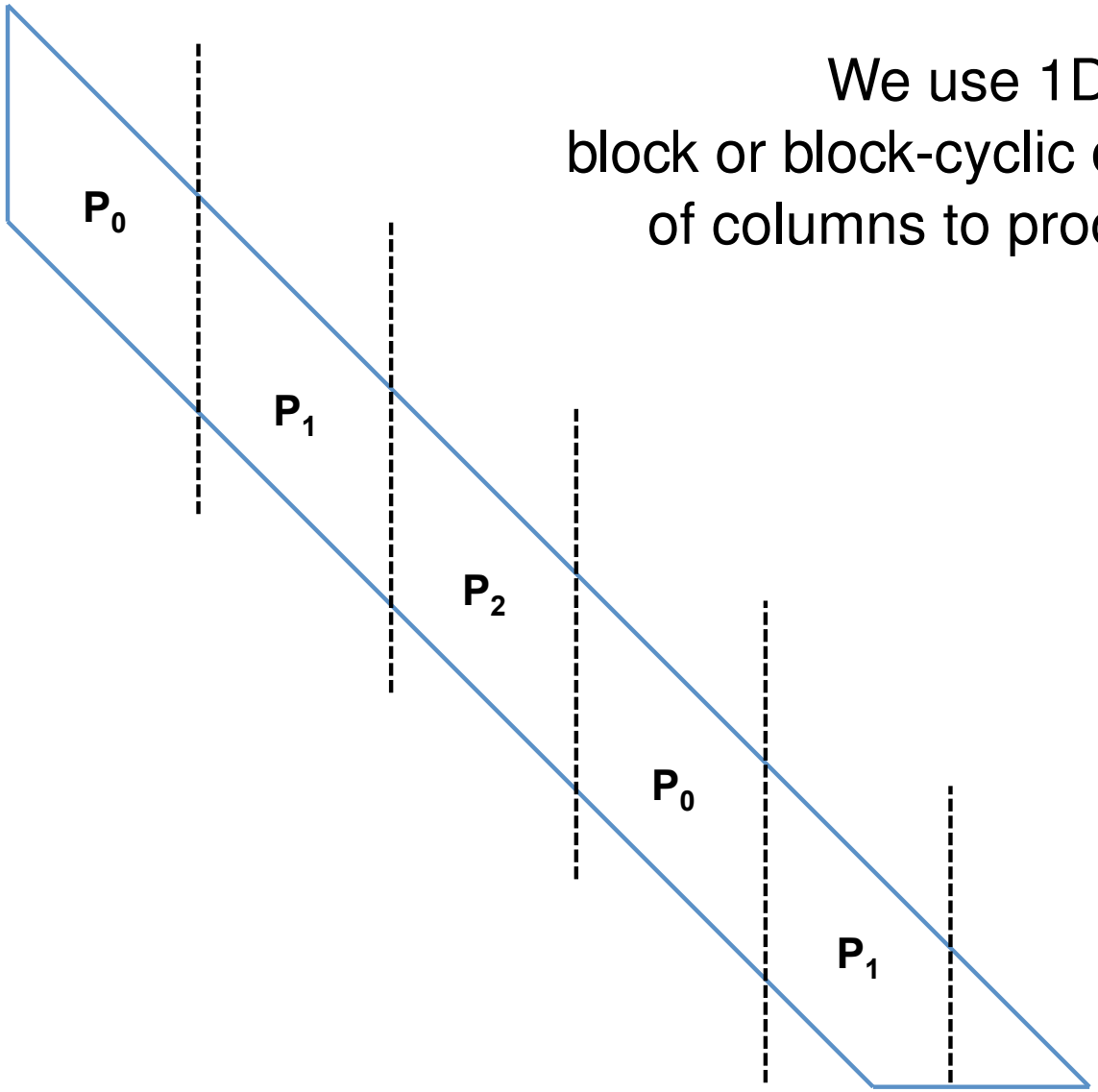
Performance Results in Shared Memory

Speedup of parallel CASBR (10 threads) over sequential CASBR



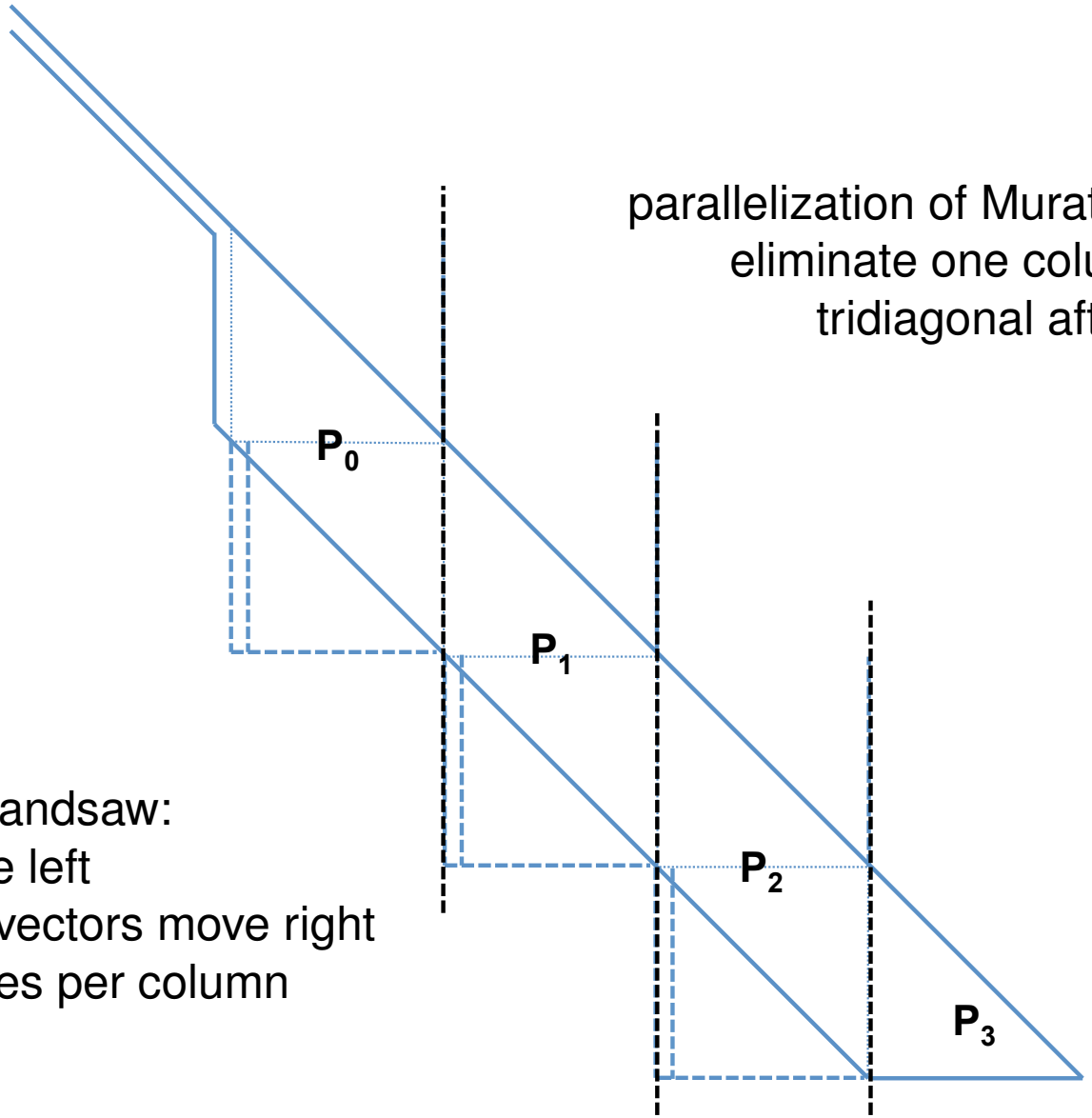
Benchmarked on 10-core Intel Westmere [BDK12]

Distributed-Memory Parallel Distribution



We use 1D
block or block-cyclic distribution
of columns to processors

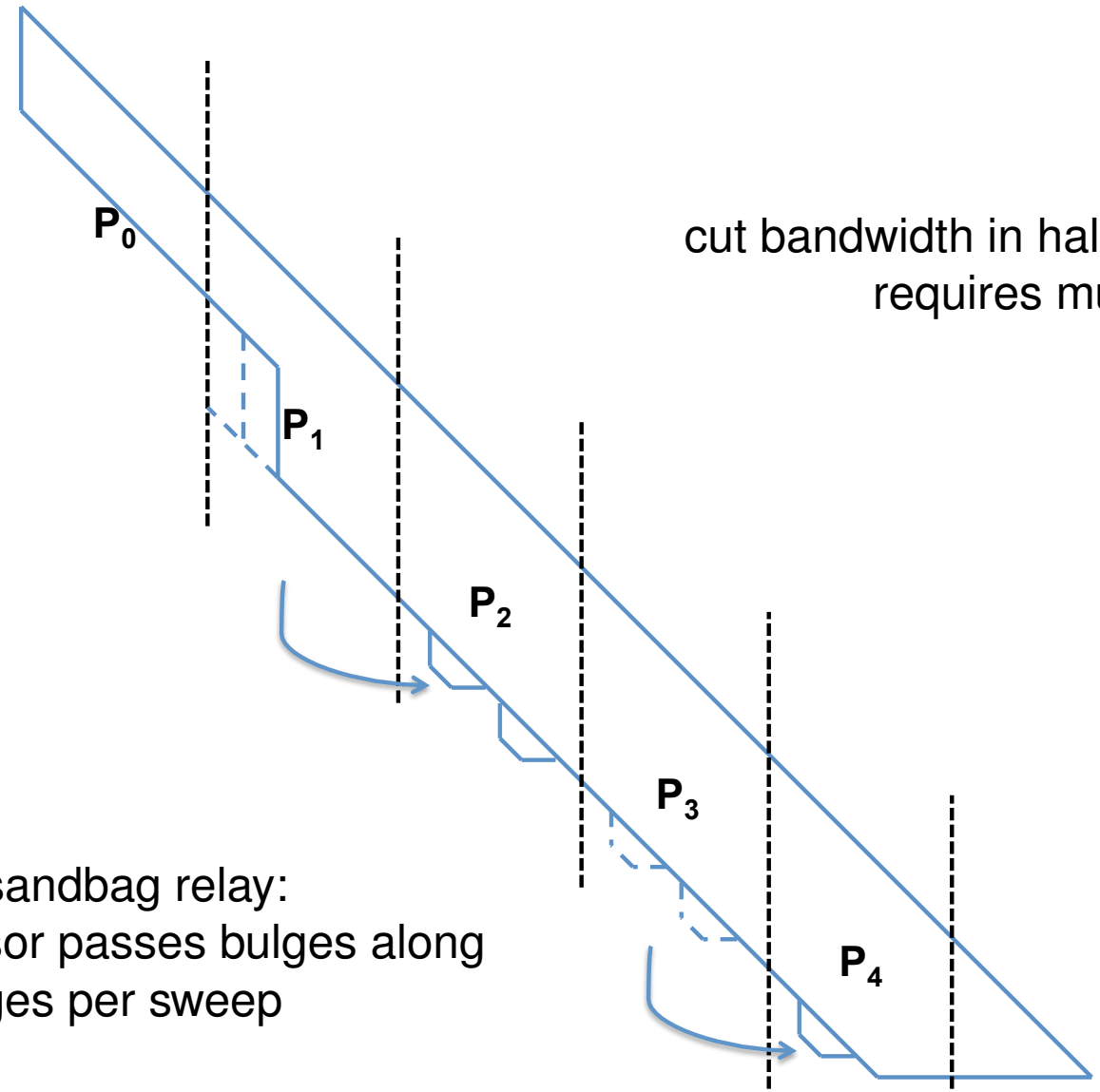
Lang's Algorithm [Lan93, Auc12]



parallelization of Murata/Horikoshi's:
eliminate one column at a time,
tridiagonal after one sweep

works like a bandsaw:
columns move left
Householder vectors move right
$O(1)$ messages per column

Communication-Avoiding SBR [BDK15]



cut bandwidth in half each sweep;
requires multiple sweeps

works like a sandbag relay:
each processor passes bulges along
$O(p)$ messages per sweep

Leading Order Costs for Parallel Band-to-Tridiagonal

Algorithm	Flops	Words	Messages
Lang's [Lan93, Auc12]	$O\left(\frac{n^2 b}{p}\right)$	$O(nb)$	$O(n)$
CA-SBR [BDK15]			$O(p \log b)$
Lower Bound	—	?	?

Costs of band-to-tridiagonal reduction of $n \times n$ band matrix with bandwidth $b \ll n$ distributed over p processors in 1D fashion.

Outline

1. EVD/SVD via Reduction to Condensed Form
2. Standard Algorithms
3. Two-Phase Approach
   - Full to Band(-Hessenberg)
   - Band Reduction
4. Open Problems

Back-Transformation for Eigenvectors

If only eigenvalues are desired, CA-SBR gives a theoretical net win, but . . .

Back-Transformation for Eigenvectors

If only eigenvalues are desired, CA-SBR gives a theoretical net win, but . . .

. . . what if we want eigenvectors?

- we must accumulate all the orthogonal transformations from the band-to-tridiagonal reduction
- we generate $O(n^2)$ data per sweep
- naively, we need $O(n^3)$ computation per sweep

Back-Transformation for Eigenvectors

If only eigenvalues are desired, CA-SBR gives a theoretical net win, but . . .

. . . what if we want eigenvectors?

- we must accumulate all the orthogonal transformations from the band-to-tridiagonal reduction
- we generate $O(n^2)$ data per sweep
- naively, we need $O(n^3)$ computation per sweep

In order to achieve $O(p \log b) \ll O(n)$ messages,
we need to take $O(\log b)$ sweeps instead of 1 sweep

- tradeoff between latency cost and flops/bandwidth cost

Open Mathematical Problem

In two-phase tridiagonalization, we compute the following matrices:

$$A = Q_1 B {Q_1}^T = Q_1 (Q_2 T Q_2^T) Q_1^T = Q_1 Q_2 (V \Lambda V^T) {Q_2}^T Q_1^T$$

where A is dense, B is band, T is tridiagonal, Λ is diagonal

We can compute A, B, T, Λ, V stably and efficiently, we seek $Q_1 Q_2 V$

Open Mathematical Problem

In two-phase tridiagonalization, we compute the following matrices:

$$A = Q_1 B {Q_1}^T = Q_1 (Q_2 T Q_2^T) Q_1^T = Q_1 Q_2 (V \Lambda V^T) {Q_2}^T Q_1^T$$

where A is dense, B is band, T is tridiagonal, Λ is diagonal

We can compute A, B, T, Λ, V stably and efficiently, we seek $Q_1 Q_2 V$

Can we compute

- $Q_1 Q_2$ from A and T; or
- Q_2 from B and T

stably (as stable as direct tridiagonalization) and efficiently ($\ll O(n)$ messages)?

Open Mathematical Problem

Problem: Given A and (similar) T, compute Q such that $A = QTQ^T$

Householder reconstruction established a connection between $A = QR$ and $A = LU$

Is there an analogous connection between $A = QTQ^T$ and $A = L\tilde{T}L^T$ (or $A = LDL^T$)?

- $\tilde{T}$ is tridiagonal from Aasen's factorization
- D is block-diagonal from Bunch-Kaufman's factorization

Open CA Problem

Two-phase tridiagonalization attains the communication lower bound assuming local memory of $O(n^2/p)$

Can you exploit the memory-communication tradeoff in the case of EVD/SVD (like 2.5D matrix multiplication)?

We're currently working on this

Open Implementation Problems

Communication-optimal algorithms have not all been implemented

Shared-memory parallel implementation of CA band reduction showed promising results

There are preliminary implementations of Householder reconstruction for full-to-band in distributed memory

CA band reduction not implemented in distributed memory

Even less work on two-phase SVD and nonsymmetric EVD

For more details:

Avoiding Communication in Successive Band Reduction
Grey Ballard, Jim Demmel, and Nick Knight
ACM Transactions on Parallel Computing 2015
http://doi.acm.org/10.1145/2686877

References I

T. Auckenthaler.

Highly Scalable Eigensolvers for Petaflop Applications.

PhD thesis, Fakultät für Informatik, Technische Universität München, 2012.

G. Ballard, J. Demmel, O. Holtz, and O. Schwartz.

Minimizing communication in numerical linear algebra.

SIAM Journal on Matrix Analysis and Applications, 32(3):866–901, September 2011.

G. Ballard, J. Demmel, and N. Knight.

Communication avoiding successive band reduction.

In *Proceedings of the 17th ACM SIGPLAN symposium on Principles and Practice of Parallel Programming*, PPoPP '12, pages 35–44, New York, NY, USA, 2012. ACM.

References II

Grey Ballard, James Demmel, and Nicholas Knight.

Avoiding communication in successive band reduction.

ACM Transactions on Parallel Computing, 1(2):11:1–11:37, February 2015.

C. Bischof, B. Lang, and X. Sun.

A framework for symmetric band reduction.

ACM Transactions on Mathematical Software, 26(4):581–601, December 2000.

J. Demmel, L. Grigori, M. Hoemmen, and J. Langou.

Communication-optimal parallel and sequential QR and LU factorizations.

SIAM Journal on Scientific Computing, 34(1):A206–A239, 2012.

References III

Azzam Haidar, Jakub Kurzak, and Piotr Luszczek.

An improved parallel singular value algorithm and its implementation for multicore hardware.

In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, SC '13, pages 90:1–90:12, New York, NY, USA, 2013. ACM.

Azzam Haidar, Raffaele Solcá , Mark Gates, Stanimire Tomov, Thomas Schulthess, and Jack Dongarra.

Leading edge hybrid multi-GPU algorithms for generalized eigenproblems in electronic structure calculations.

In J.M. Kunkel, T. Ludwig, and H. W. Meuer, editors, *Supercomputing*, volume 7905 of *Lecture Notes in Computer Science*, pages 67–80. Springer Berlin Heidelberg, 2013.

References IV

Toshiyuki Imamura, Susumu Yamada, and Masahiko Machida.

Development of a high performance eigensolver on the petascale next generation supercomputer system.

In *Proceedings of Joint International Conference on Supercomputing in Nuclear Applications and Monte Carlo 2010 (SNA+ MC2010)*, 2011.

Toshiyuki Imamura, Susumu Yamada, and Masahiko Machida.

Eigen-G: GPU-based eigenvalue solver for real-symmetric dense matrices.

In Roman Wyrzykowski, Jack Dongarra, Konrad Karczewski, and Jerzy Waśniewski, editors, *Parallel Processing and Applied Mathematics*, volume 8384 of *Lecture Notes in Computer Science*, pages 673–682. Springer Berlin Heidelberg, 2014.

L. Kaufman.

Banded eigenvalue solvers on vector machines.

ACM Transactions on Mathematical Software, 10:73–86, 1984.

References V

L. Kaufman.

Band reduction algorithms revisited.

ACM Transactions on Mathematical Software, 26(4):551–567, 2000.

L. Karlsson and B. Kågström.

Parallel two-stage reduction to hessenberg form using dynamic scheduling on shared-memory architectures.

Parallel Computing, 37(12):771–782, 2011.

6th International Workshop on Parallel Matrix Algorithms and Applications (PMAA'10).

B. Lang.

A parallel algorithm for reducing symmetric banded matrices to tridiagonal form.

SIAM Journal on Scientific Computing, 14(6):1320–1338, November 1993.

References VI

P. Luszczek, H. Ltaief, and J. Dongarra.

Two-stage tridiagonal reduction for dense symmetric matrices using tile algorithms on multicore architectures.

In *Proceedings of the 2011 IEEE International Parallel & Distributed Processing Symposium*, IPDPS '11, pages 944–955, Washington, DC, USA, 2011. IEEE Computer Society.

A. Marek, V. Blum, R. Johanni, V. Havu, B. Lang, T. Auckenthaler, A. Heinecke, H.-J. Bungartz, and H. Lederer.

The ELPA library: scalable parallel eigenvalue solutions for electronic structure theory and computational science.

Journal of Physics: Condensed Matter, 26(21), 2014.

K. Murata and K. Horikoshi.

A new method for the tridiagonalization of the symmetric band matrix.

Information Processing in Japan, 15:108–112, 1975.

References VII

H. Rutishauser.

On Jacobi rotation patterns.

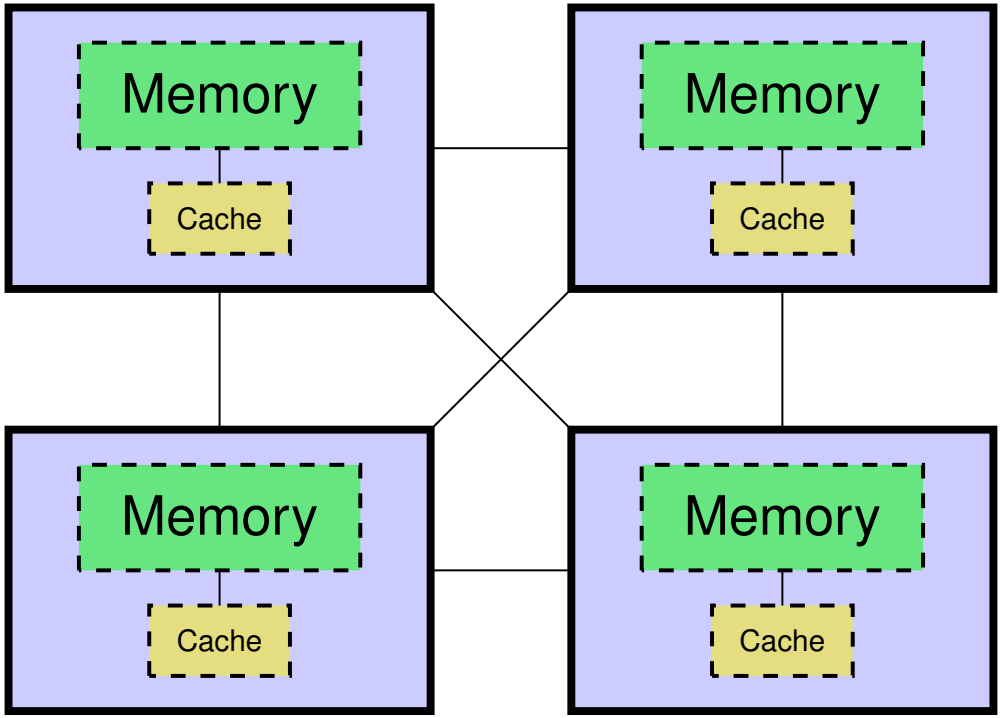
In *Proceedings of Symposia in Applied Mathematics*, volume 15, pages 219–239. AMS, 1963.

H. Schwarz.

Algorithm 183: reduction of a symmetric bandmatrix to triple diagonal form.

Communications of the ACM, 6(6):315–316, June 1963.

Model of Distributed-Memory Parallel Computation



To analyze algorithms, we are interested in the following quantities

- **flops** — floating point operations
- **memory bandwidth cost** — words moved between memory and cache
- **interprocessor bandwidth cost** — words communicated between processors
- **latency cost** — messages communicated between processors