Communication avoiding QR factorization and orthogonalization of a set of vectors

Laura Grigori

EPFL and PSI

October 3, 2023

EPFL

PAUL SCHERRER INSTITUT PSI

Plan

Orthogonalization processes

- QR factorization
- Gram-Schmidt (GS) orthogonalization process
- CholeskyQR
- Block QR factorization

Communication avoiding QR factorization

- TSQR: QR factorization of a tall skinny matrix
- Reconstruct Householder vectors from TSQR

Plan

Orthogonalization processes
- QR factorization
- Gram-Schmidt (GS) orthogonalization process
- CholeskyQR
- Block QR factorization

Communication avoiding QR factorization

The QR factorization

Given a matrix $A \in \mathbb{R}^{m \times n}$, $m \geq n$, its QR factorization is

$$A = QR = \begin{pmatrix} \tilde{Q} & \bar{Q} \end{pmatrix} \begin{pmatrix} \tilde{R} \\ 0 \end{pmatrix} = \tilde{Q}\tilde{R}$$

where $Q \in \mathbb{R}^{m \times m}$ is orthogonal and $R \in \mathbb{R}^{m \times n}$ is upper triangular.
The thin factorization has $\tilde{Q} \in \mathbb{R}^{m \times n}$ and $\tilde{R} \in \mathbb{R}^{n \times n}$.

If A has full rank, the factorization $\tilde{Q}\tilde{R}$ is essentialy unique (modulo signs of diagonal elements of $\tilde{R}$).

- $A^T A = \tilde{R}^T \tilde{R}$ is a Cholesky factorization and $A = A\tilde{R}^{-1}\tilde{R}$ is a QR factorization.
- $A = \tilde{Q}D \cdot D\tilde{R}$, $D = diag(\pm 1)$ is a QR factorization.

Gram-Schmidt (GS) orthogonalization process

Given set of linearly independent vectors $W = [w_1, \ldots, w_b]$, $W \in \mathbb{R}^{m \times b}$, $m \gg b$.
Construct an orthogonal basis $\tilde{Q} = [q_1, \ldots, q_b]$ such that $W = \tilde{Q}\tilde{R}$, $\tilde{Q} \in \mathbb{R}^{m \times b}$, $\tilde{R} \in \mathbb{R}^{b \times b}$.

For each w_j **do**

1. Given P_{j-1} projector on $\mathrm{span}(\tilde{Q}_{j-1})^{\perp}$, compute $q_j \perp q_1, \ldots, q_{j-1}$ as

$$q_j = P_{j-1} w_j$$

2. Normalize q_j

End For

Projectors P_j used in Gram-Schmidt

- **Classical Gram-Schmidt (CGS)**: BLAS-2 matrix-vector operations
  - one synchronization for each new vector w_j

$$P_{j-1} = I - \tilde{Q}_{j-1}\tilde{Q}_{j-1}^T$$

  - Numerical stability: assume $c_2(m,b)\kappa^2(W)u < 1$,

$$\|I - \tilde{Q}^T\tilde{Q}\| = c_1(m,b)\kappa^2(W)u, \quad c_1, c_2 = \mathcal{O}(mb^2)$$

- **CGS2**: BLAS-2 matrix-vector operations
  - CGS2: two synchronizations for each vector w_j

$$P_{j-1} = (I - \tilde{Q}_{j-1}\tilde{Q}_{j-1}^T)(I - \tilde{Q}_{j-1}\tilde{Q}_{j-1}^T),$$

  - Numerical stability: assume $c_4(m,b)\kappa(W)u < 1, c_4 = \mathcal{O}(m^2b^3)$

$$\|I - \tilde{Q}^T\tilde{Q}\| = c_3(m,n)u, c_3 = \mathcal{O}(mb^{3/2})$$

$\rightarrow$ good efficiency, but stability issues

Projectors P_j used in Gram-Schmidt

- **Modified Gram-Schmidt (MGS)** : BLAS-1 vector-vector operations
  - $(j-1)$ synchronizations for each vector w_j

$$P_{j-1} = (I - q_{j-1}q_{j-1}^T)\dots(I - q_1 q_1^T)$$

  - Numerical stability: assume $c_3(m,b)\kappa(W)u < 1$,

$$\|I - \tilde{Q}^T\tilde{Q}\| = c_3(m,b)\kappa(W)u, c_3 = \mathcal{O}(mb)$$

  $\rightarrow$ better numerical stability, but poor efficiency

- Orthogonal projector

$$P_{j-1} = I - \tilde{Q}_{j-1}(\tilde{Q}_{j-1}^T\tilde{Q}_{j-1})^{-1}\tilde{Q}_{j-1} = I - \tilde{Q}_{j-1}\tilde{Q}_{j-1}^{\dagger}$$

Note: u machine precision, $W \in \mathbb{R}^{m\times b}$, $\kappa(W)$ is condition number of W, bounds from [Giraud et al., 2005].

Projectors P_j used in Gram-Schmidt

- **Modified Gram-Schmidt (MGS)** : BLAS-1 vector-vector operations
  - $(j-1)$ synchronizations for each vector w_j

$$P_{j-1} = (I - q_{j-1}q_{j-1}^T)\dots(I - q_1 q_1^T)$$

  - Numerical stability: assume $c_3(m,b)\kappa(W)u < 1$,

$$\|I - \tilde{Q}^T\tilde{Q}\| = c_3(m,b)\kappa(W)u, c_3 = \mathcal{O}(mb)$$

  $\rightarrow$ better numerical stability, but poor efficiency

- Orthogonal projector

$$P_{j-1} = I - \tilde{Q}_{j-1}(\tilde{Q}_{j-1}^T\tilde{Q}_{j-1})^{-1}\tilde{Q}_{j-1} = I - \tilde{Q}_{j-1}\tilde{Q}_{j-1}^{\dagger}$$

Note: u machine precision, $W \in \mathbb{R}^{m\times b}$, $\kappa(W)$ is condition number of W, bounds from [Giraud et al., 2005].

Cholesky QR

- **CholeskyQR** : BLAS-3 matrix-matrix operations
  Compute $W = \tilde{Q}\tilde{R}$ as:
  1. Compute the Cholesky factorization of $W^T W$ as $W^T W = \tilde{R}^T \tilde{R}$
  2. Compute the orthogonal factor as $\tilde{Q} = W\tilde{R}^{-1}$
- Numerical stability: assume $O(u)\kappa^2(W) < 1$, then

$$\|I - \tilde{Q}^T \tilde{Q}\| = O(u)\kappa^2(W)$$

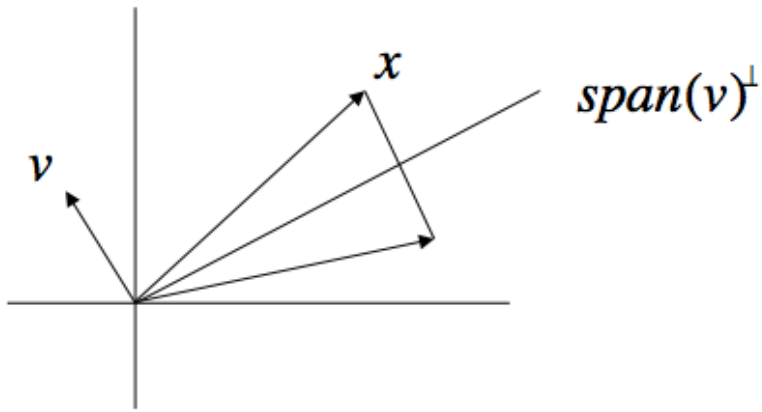
Householder transformation

The Householder matrix

$$P = I - \frac{2}{v^T v} v v^T$$

has the following properties:

- is symmetric and orthogonal, $P^2 = I$,
- is independent of the scaling of v,
- it reflects x about the hyperplane $span(v)^{\perp}$



$$Px = x - \frac{2v^T x}{v^T v} v = x - \alpha v$$

Presentation of Householder transformations and stability analysis from [N.J.Higham, 2002].

Householder for the QR factorization

We look for a Householder matrix that allows to annihilate the elements of a vector x, except first one.

$$Px = y, \quad \|x\|_2 = \|y\|_2, \quad y = \sigma e_1, \quad \sigma = \pm\|x\|_2$$

With the choice of sign made to avoid cancellation when computing $v_1 = x_1 - \sigma$ (where v_1, x_1 are the first elements of vectors v, x respectively), we have

$$\begin{aligned} v &= x - y = x - \sigma e_1, \\ \sigma &= -sign(x_1)\|x\|_2, v = x - \sigma e_1, \\ P &= I - \beta v v^T, \beta = \frac{2}{v^T v} \end{aligned}$$

Householder based QR factorization

$$A = \begin{pmatrix} x & x & x \\ x & x & x \\ x & x & x \end{pmatrix}$$

$$P_1 A = \begin{pmatrix} x & x & x \\ 0 & x & x \\ 0 & x & x \end{pmatrix}, \begin{pmatrix} 1 & \\ & \tilde{P}_2 \end{pmatrix} P_1 = \begin{pmatrix} x & x & x \\ 0 & x & x \\ 0 & 0 & x \end{pmatrix} = R$$

So we have

$$\begin{aligned} Q^T A &= P_n P_{n-1} \dots P_1 A = R, \\ Q &= (I - \beta_1 v_1 v_1^T) \dots (I - \beta_{n-1} v_{n-1} v_{n-1}^T)(I - \beta_n v_n v_n^T) \end{aligned}$$

$\#flops = 2n^2(m - n/3)$

Error analysis of the QR factorization

The following result follows

Theorem ([N.J.Higham, 2002])

Let $\hat{R} \in \mathbb{R}^{m\times n}$ be the computed factor of $A \in \mathbb{R}^{m\times n}$ obtained by using Householder transformations. Then there is an orthogonal $Q \in \mathbb{R}^{m\times m}$ such that

$$A + \Delta A = Q\hat{R}, \text{ where } \|\Delta a_j\|_2 \leq \tilde{\gamma}_{mn}\|a_j\|_2, \quad j = 1:n,$$

where $\tilde{\gamma}_{mn} = cmnu/(1 - cmnu)$, c is a constant, u is machine precision, $mnu < 1$, a_j denotes the j-th column of A.

Computational complexity

- Flops per iterations
  - Dot product $w = v_k^T A(k:m, k+1:n) : 2(m-k)(n-k)$
  - Outer product $v_k w : (m-k)(n-k)$
  - Subtraction $A(k:m, k+1:n) - \ldots : (m-k)(n-k)$
- Flops of Householder-QR

$$\sum_{k=1}^{n} 4(m-k)(n-k) = 4\sum_{k=1}^{n}(mn - k(m+n) + k^2)$$
$$\approx 4mn^2 - 4(m+n)n^2/2 + 4n^3/3 = 2mn^2 - 2n^3/3$$

Algebra of block QR

Storage efficient representation for Q [Schreiber and Loan, 1989]

$$Q = P_1 P_2 \dots P_k = (I - \beta_1 v_1 v_1^T) \dots (I - \beta_k v_k v_k^T) = I - YTY^T$$

Example for $k = 2$

$$Y = (v_1 | v_2), \quad T = \begin{pmatrix} \beta_1 & -\beta_1 v_1^T v_2 \beta_2 \\ 0 & \beta_2 \end{pmatrix}$$

Example for combining two compact representations

$$\begin{aligned} Q &= (I - Y_1 T_1 Y_1^T)(I - Y_2 T_2 Y_2^T) \\ T &= \begin{pmatrix} T_1 & -T_1 Y_1^T Y_2 T_2 \\ 0 & T_2 \end{pmatrix} \end{aligned}$$

Block algorithm for computing the QR factorization

Partitioning of matrix A of size $m \times n$

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$$

where A_{11} is of size $b \times b$, A_{21} is of size $(m-b) \times b$, A_{12} is of size $b \times (n-b)$ and A_{22} is of size $(m-b) \times (n-b)$.

Block QR algebra

The first step of the block QR factorization algorithm computes:

$$Q_1^T A = \begin{pmatrix} R_{11} & R_{12} \\ & A^1 \end{pmatrix}$$

The algorithm continues recursively on the trailing matrix A^1.

Algebra of block QR factorization

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} = Q_1 \begin{pmatrix} R_{11} & R_{12} \\ & A^1 \end{pmatrix}$$

Block QR algebra

1. Compute the factorization

$$\begin{pmatrix} A_{11} \\ A_{12} \end{pmatrix} = Q_1 R_{11}$$

2. Compute the compact representation $Q_1 = I - YTY^T$
3. Apply Q_1^T on the trailing matrix

$$(I - YT^T Y^T) \begin{pmatrix} A_{12} \\ A_{22} \end{pmatrix} = \begin{pmatrix} A_{12} \\ A_{22} \end{pmatrix} - Y \left(T^T \left(Y^T \begin{pmatrix} A_{12} \\ A_{22} \end{pmatrix} \right) \right)$$

4. The algorithm continues recursively on the trailing matrix A^1.

Parallel implementation of the QR factorization

QR factorization on a $P = P_r \times P_c$ grid of processors
For ib $= 1$ to n-1 step b

1. Compute panel factorization on P_r processors

$$\begin{pmatrix} A_{11} \\ A_{12} \end{pmatrix} = Q_1 R_{11} = (I - YTY^T) R_{11}$$

2. The P_r processors broadcast along the rows their parts of Y and T
3. Apply Q_1^T on the trailing matrix:
   - All processors compute their local part of
   $$W_l = Y_l^T (A_{12l}; A_{22l})$$
   - The processors owning block row ib compute the sum over W_l, that is
   $$W = Y^T (A_{12}; A_{22})$$
   and then compute $W' = T^T W$
   - The processors owning block row ib broadcast along the columns their part of W'
4. All processors compute
$$(A_{12}^1; A_{22}^1) = (A_{12}; A_{22}) - Y * W'$$

Cost of parallel QR factorization

$$
\begin{aligned}
& \gamma \cdot \left(\frac{6mnb - 3n^2 b}{2p_r} + \frac{n^2 b}{2p_c} + \frac{2mn^2 - 2n^3/3}{p} \right) \\
+ \quad & \beta \cdot \left(nb \log p_r + \frac{2mn - n^2}{p_r} + \frac{n^2}{p_c} \right) \\
+ \quad & \alpha \cdot \left(2n \log p_r + \frac{2n}{b} \log p_c \right) .
\end{aligned}
$$

Plan

Orthogonalization processes

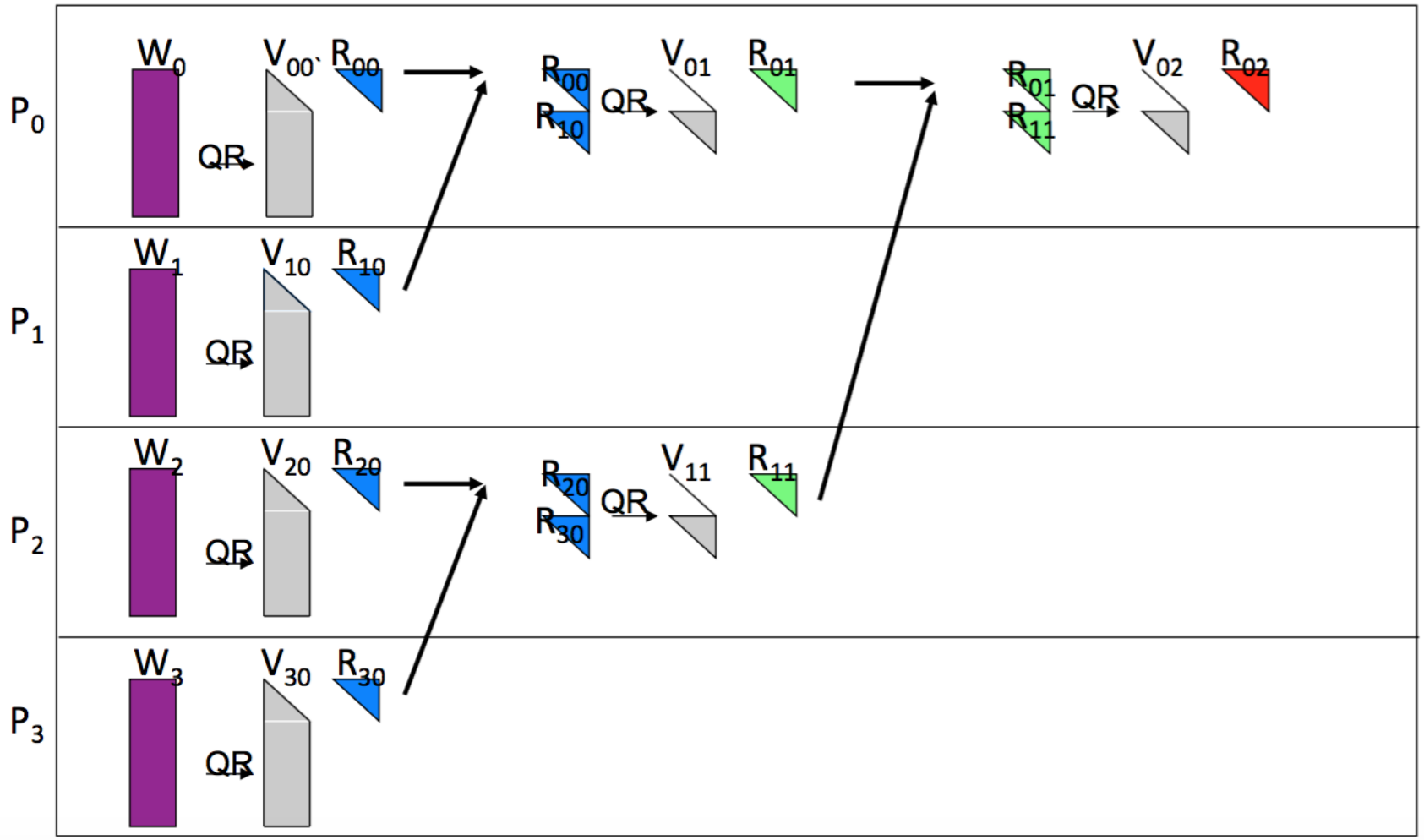
Communication avoiding QR factorization

- TSQR: QR factorization of a tall skinny matrix
- Reconstruct Householder vectors from TSQR

TSQR: QR factorization of a tall skinny matrix

- QR decomposition of m x b matrix W, $m \gg b$ using Householder transformations
  - P processors, block row layout
- Classic Parallel Algorithm
  - Compute Householder vector for each column
  - Number of messages $\approx b \cdot log_2 P$
- Communication Avoiding Algorithm
  - Reduction operation, with QR as operator
  - Number of messages $log_2 P$

TSQR: QR factorization of a tall skinny matrix



J. Demmel, LG, M. Hoemmen, J. Langou, 08

References: Golub, Plemmons, Sameh 88, Pothen, Raghavan, 89, Da Cunha, Becker, Patterson, 02

Algebra of TSQR

$$W = \begin{pmatrix} W_0 \\ W_1 \\ W_2 \\ W_3 \end{pmatrix} = \begin{pmatrix} Q_{00}R_{00} \\ Q_{10}R_{10} \\ Q_{20}R_{20} \\ Q_{30}R_{30} \end{pmatrix} = \left(\begin{array}{c|c|c|c} Q_{00} & & & \\ \hline & Q_{10} & & \\ \hline & & Q_{20} & \\ \hline & & & Q_{30} \end{array}\right) \left(\begin{array}{c} R_{00} \\ R_{10} \\ \hline R_{20} \\ R_{30} \end{array}\right)$$

$$\left(\begin{array}{c} R_{00} \\ R_{10} \\ \hline R_{20} \\ R_{30} \end{array}\right) = \left(\begin{array}{c} \begin{pmatrix} R_{00} \\ R_{10} \end{pmatrix} \\ \hline \begin{pmatrix} R_{20} \\ R_{30} \end{pmatrix} \end{array}\right) = \left(\begin{array}{c} Q_{01}R_{01} \\ \hline Q_{11}R_{11} \end{array}\right) = \left(\begin{array}{c|c} Q_{01} & \\ \hline & Q_{11} \end{array}\right) \left(\begin{array}{c} R_{01} \\ \hline R_{11} \end{array}\right)$$

$$\left(\begin{array}{c} R_{01} \\ \hline R_{11} \end{array}\right) = Q_{02} \cdot R_{02}$$

Algebra of TSQR (contd)

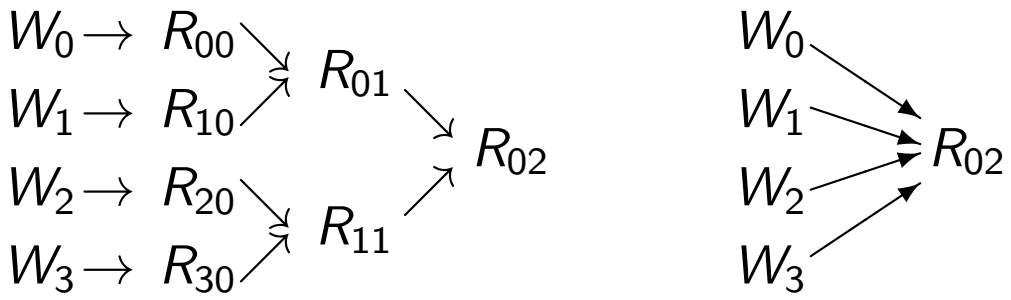
- Q is represented implicitly as a product of orthogonal matrices

$$W = \begin{pmatrix} W_0 \\ W_1 \\ W_2 \\ W_3 \end{pmatrix} = \left(\begin{array}{c|c|c|c} Q_{00} & & & \\ \hline & Q_{10} & & \\ \hline & & Q_{20} & \\ \hline & & & Q_{30} \end{array}\right) \cdot \left(\begin{array}{c|c} Q_{01} & \\ \hline & Q_{11} \end{array}\right) \cdot Q_{02} \cdot R_{02}. \quad (1)$$

- For the products to make sense, dimensions of intermediate Q factors must be chosen consistently. They can all be square, or they can all have b columns (in which case each R factor will be $b \times b$)

Flexibility of TSQR and CAQR algorithms

- Reduction tree will depend on the underlying architecture, could be chosen dynamically



(a) Parallel TSQR (b) Householder QR

Figure: Graphical comparison of parallel TSQR and Householder QR.

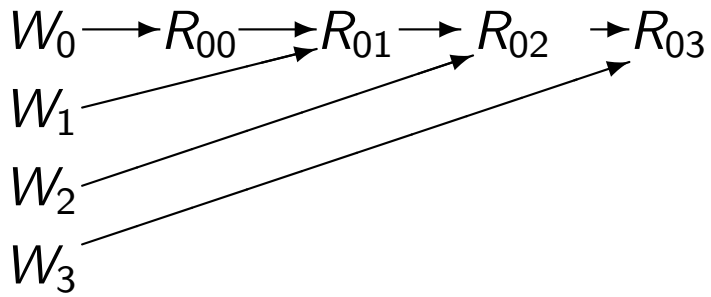
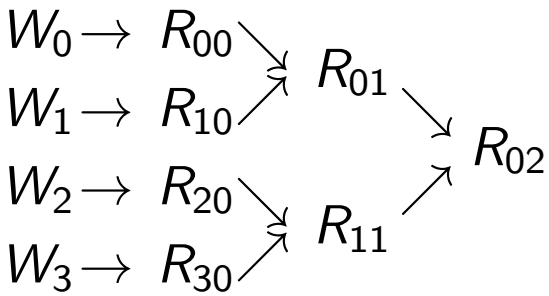
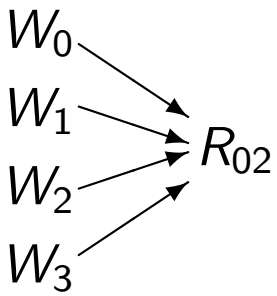


Figure: Graphical representation of sequential TSQR.

CAQR based on TSQR



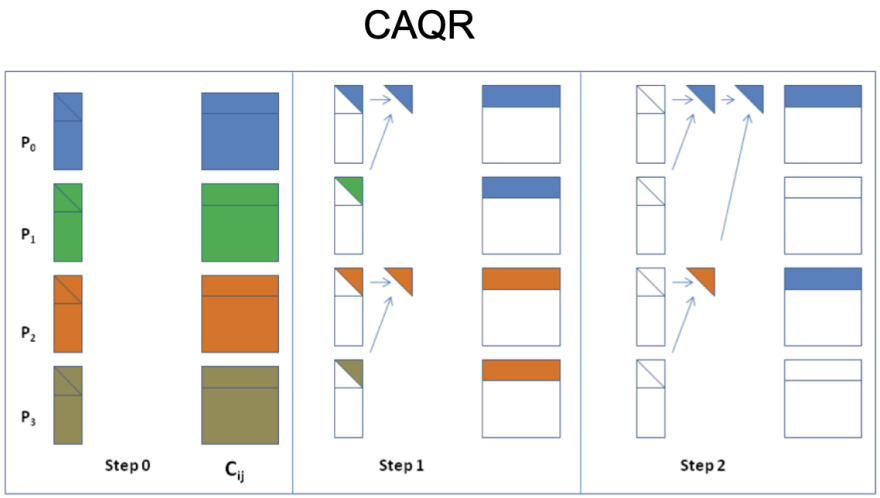
(a) Parallel TSQR



(b) Householder QR

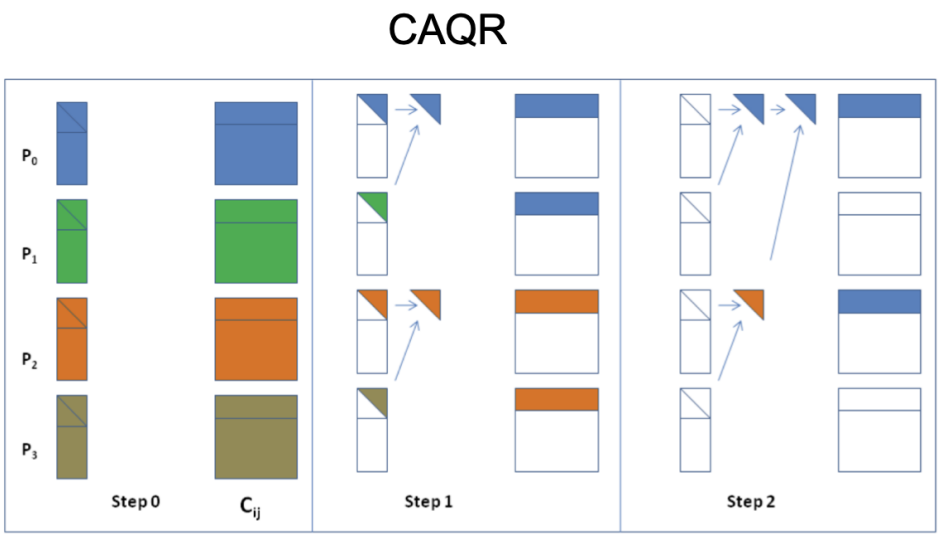
CAQR: communication avoiding QR factorization of $A \in \mathbb{R}^{m \times n}$

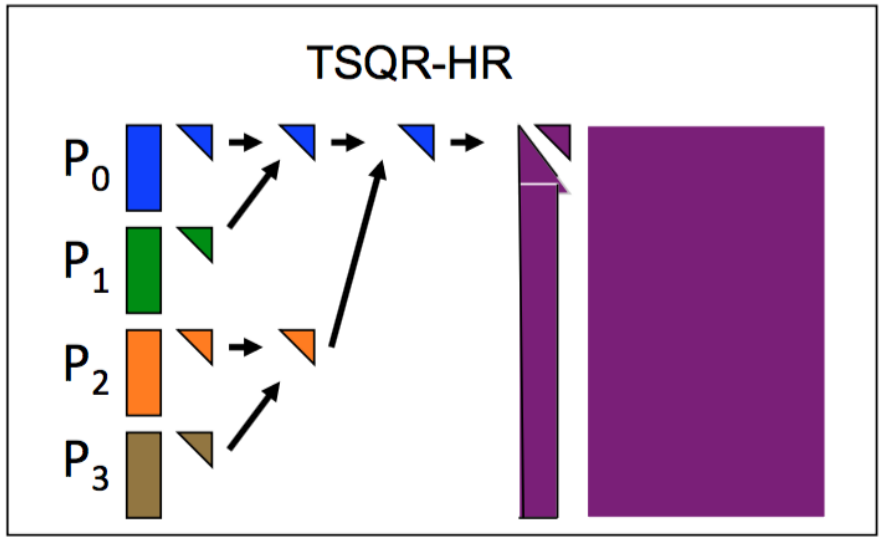
- TSQR used to compute the panel factorization
- Update of trailing matrix follows same reduction tree structure as TSQR



TSQR: QR factorization of a tall skinny matrix

Goal: to simplify the usage of TSQR in CAQR, reconstruct Householder vectors from TSQR to provide the same output as Householder QR.





J. Demmel, LG, M. Hoemmen, J. Langou, 08

Ballard, Demmel, LG, Jacquelin, Nguyen, Solomonik, 14

Reconstruct Householder vectors from TSQR

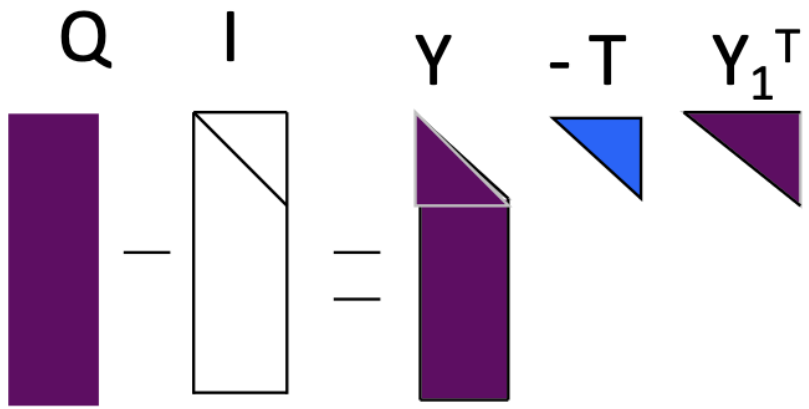
- The QR factorization using Householder vectors

$$W = \tilde{Q}\tilde{R} = (I - YTY_1^T)$$

where $Y = \begin{pmatrix} Y_1 \\ Y_2 \end{pmatrix}$

- can be re-written as an LU factorization

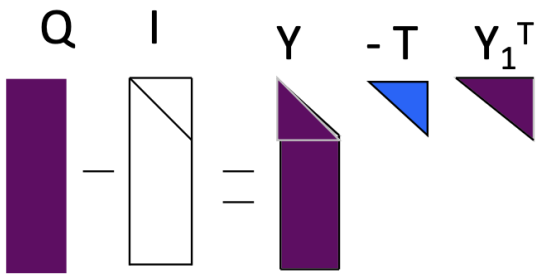
$$\begin{aligned} W - \tilde{R} &= Y(-TY_1^T)\tilde{R} \\ \tilde{Q} - I &= Y(-TY_1^T) \end{aligned}$$



NB: in the figure Q should be the thin $\tilde{Q}$

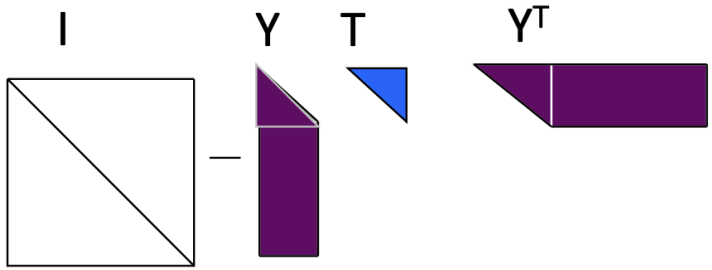
Reconstruct Householder vectors from TSQR

1. Perform TSQR
2. Form $\tilde{Q}$ explicitely (tall-skinny orthonormal factor)
3. Perform LU decomposition $\tilde{Q} - I = LU$
4. Set $Y = L$
5. Set $T = -UY_1^{-T}$

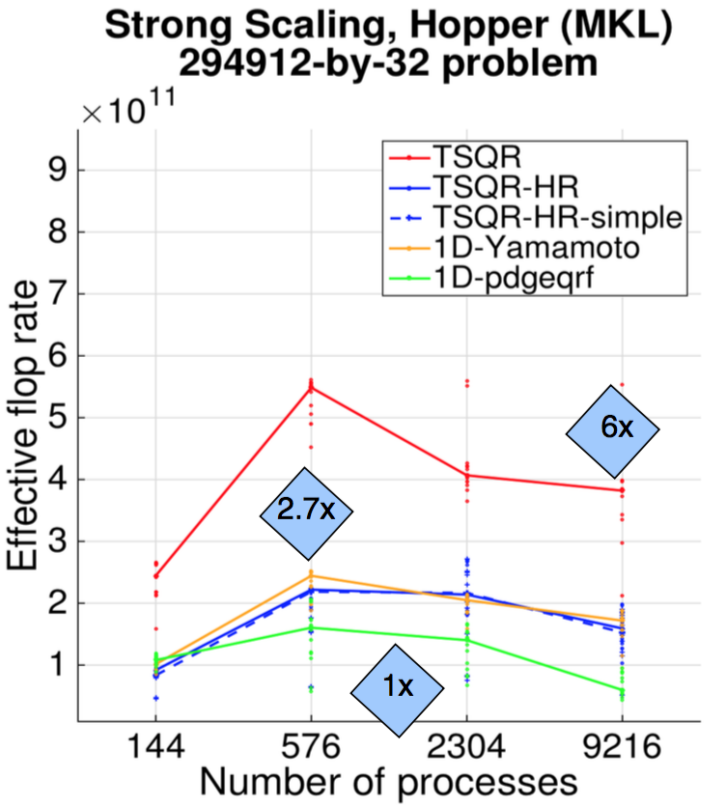


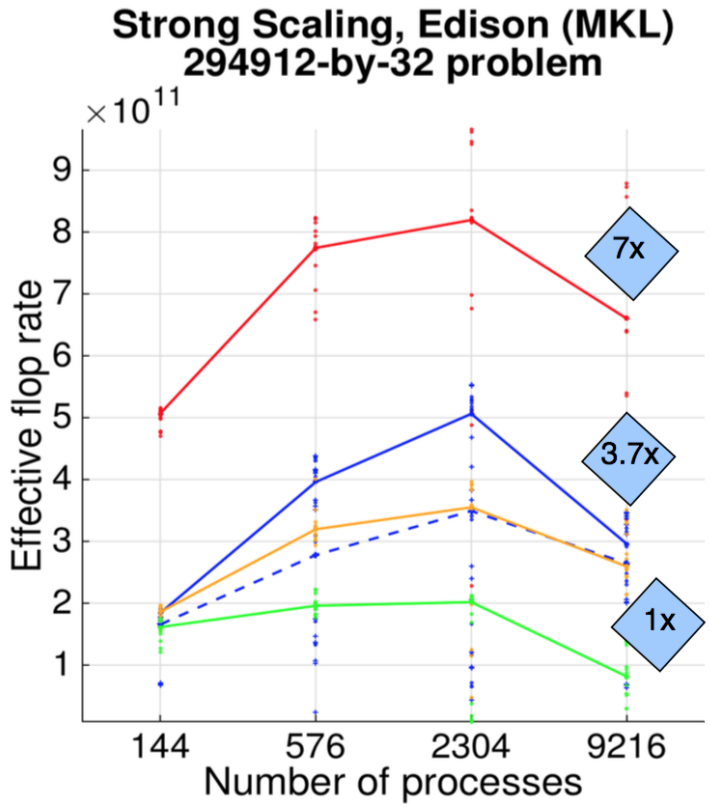
Obtain the compact representation:

$$Q = I - YTY^T = I - \begin{pmatrix} Y_1 \\ Y_2 \end{pmatrix} T \left(Y_1^T Y_2^T \right)$$



Strong scaling of TSQR





- Hopper: Cray XE6 (NERSC) – 2 x 12-core AMD Magny-Cours (2.1 GHz)
- Edison: Cray CX30 (NERSC) – 2 x 12-core Intel Ivy Bridge (2.4 GHz)
- Effective flop rate, computed by dividing $2mn^2 - 2n^3/3$ by measured runtime

Ballard, Demmel, LG, Jacquelin, Knight, Nguyen, and Solomonik, 2015

CAQR based on TSQR

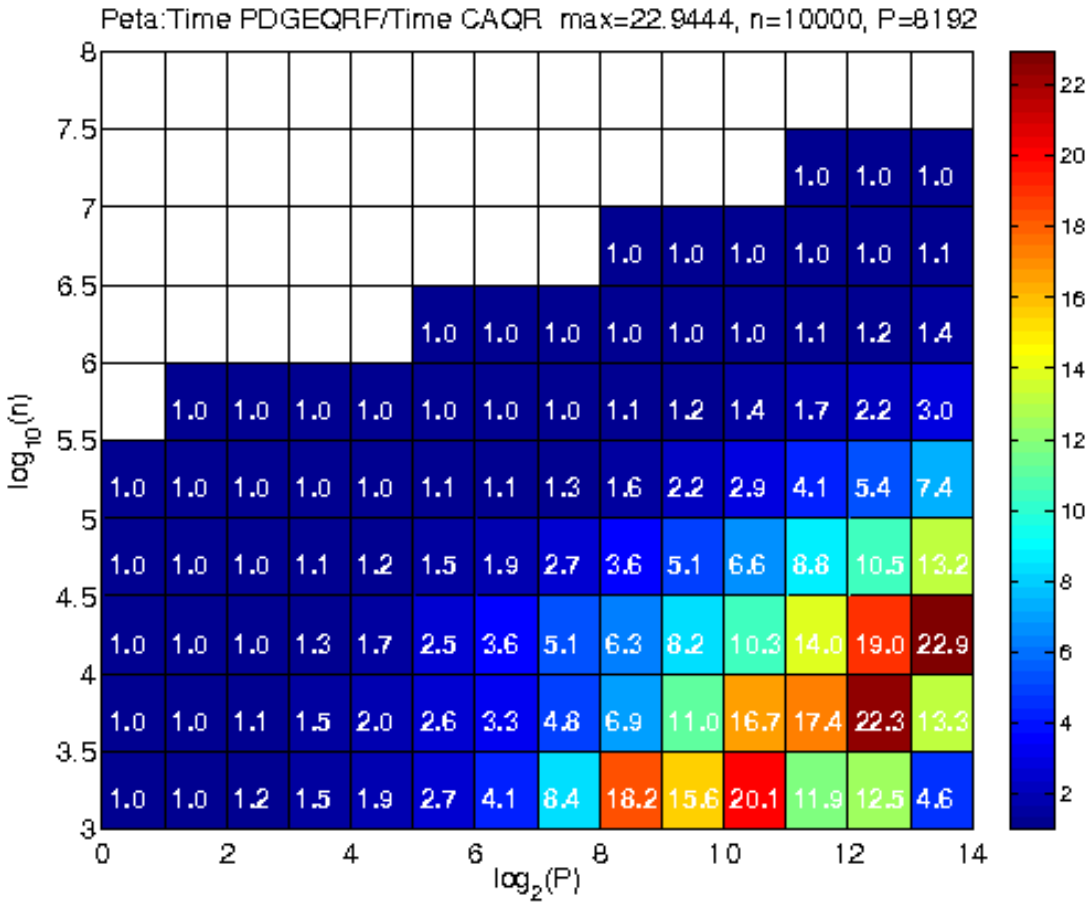
Cost of CAQR vs ScaLAPACK's PDGEQRF

- $n \times n$ matrix on $\sqrt{P} \times \sqrt{P}$ processor grid, block size b
- Flops: $(4/3)n^3/P + 3/4n^2 b \log_2 P/\sqrt{P}$ vs $(4/3)n^3/P$
- Bandwidth: $(3/4)n^2 \log_2 P/\sqrt{P}$ vs same
- Latency: $2.5n \log_2 P/b$ vs $1.5n \log_2 P$

Close to optimal (modulo log P factors)

- Assume $O(n^2/P)$ memory/processor, $O(n^3)$ algorithm
- Choose b near $n/\sqrt{P}$ (its upper bound)
- Bandwidth lower bound: $\Omega(n^2/\sqrt{P})$ – just $\log_2 P$ smaller
- Latency lower bound: $\Omega(\sqrt{P})$ – just polylog(P) smaller

Modeled Speedups of CAQR vs ScaLAPACK



Petascale machine with 8192 procs, each at 500 GFlops/s, a bandwidth of 4 GB/s, $\gamma = 2 \cdot 10^{-12}, \alpha = 10^{-5}, \beta = 2 \cdot 10^{-9}$ sec/word

Impact

- TSQR/CAQR implemented in
  - Intel Data analytics library
  - GNU Scientific Library
  - ScaLAPACK
  - Spark for data mining
- CALU (introduced in next lecture) implemented in
  - Cray's libsci
  - To be implemented in lapack/scapalack

Solving least squares problems

Given matrix $A \in \mathbb{R}^{m \times n}$, $rank(A) = n$, vector $b \in \mathbb{R}^{m \times 1}$, the unique solution to $\min_x \|Ax - b\|_2$ is

$$x = A^+ b, \quad A^+ = (A^T A)^{-1} A^T$$

Using the QR factorization of A

$$A = QR = \begin{pmatrix} \tilde{Q} & \bar{Q} \end{pmatrix} \begin{pmatrix} \tilde{R} \\ 0 \end{pmatrix} \tag{2}$$

We obtain

$$\begin{aligned} ||r||_2^2 &= ||b - Ax||_2^2 = ||b - \begin{pmatrix} \tilde{Q} & \bar{Q} \end{pmatrix} \begin{pmatrix} \tilde{R} \\ 0 \end{pmatrix} x||_2^2 \\ &= || \begin{pmatrix} \tilde{Q}^T \\ \bar{Q}^T \end{pmatrix} b - \begin{pmatrix} \tilde{R} \\ 0 \end{pmatrix} x||_2^2 = || \begin{pmatrix} \tilde{Q}^T b - \tilde{R}x \\ \bar{Q}^T b \end{pmatrix} ||_2^2 \\ &= ||\tilde{Q}^T b - \tilde{R}x||_2^2 + ||\bar{Q}^T b||_2^2 \end{aligned}$$

Solve $\tilde{R}x = \tilde{Q}^T b$ to minimize $||r||_2$.

Acknowledgement

- Some of the examples taken from [Golub and Van Loan, 1996]

References (1)


Giraud, L., Langou, J., Rozložník, M., and Eshof, J. v. d. (2005).
Rounding error analysis of the classical gram-schmidt orthogonalization process.
Numerische Mathematik, 101(1):87–100.


Golub, G. H. and Van Loan, C. F. (1996).
Matrix Computations (3rd Ed.).
Johns Hopkins University Press, Baltimore, MD, USA.


N.J.Higham (2002).
Accuracy and Stability of Numerical Algorithms.
SIAM, second edition.


Schreiber, R. and Loan, C. V. (1989).
A storage efficient *WY* representation for products of Householder transformations.
SIAM J. Sci. Stat. Comput., 10(1):53–57.