

A novel parallel QR algorithm

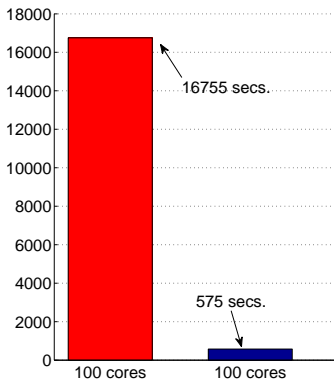
D. Kressner

ETH Zurich, Seminar for Applied Mathematics

Joint work with [R. Granat](#), [B. Kågström](#), U Umeå, Sweden.

23rd Biennial Conference on Numerical Analysis





Overall execution time of the QR algorithm for a $16\,000 \times 16\,000$ matrix on 25 Intel Xeon quadcore nodes:

ScaLAPACK vs. **new implementation**.

Outline

- Chasing multiple bulges in parallel.
- Aggressive early deflation in parallel.
- Overview of new implementation.
- Numerical experiments.

QR algorithm in a nutshell

Step 1 (non-iterative). Reduction to Hessenberg form:

$$Q_1^T A Q_1 = H = \begin{array}{|c} \square \\ \hline \end{array}$$

Step 2 (iterative). Reduction from Hessenberg to Schur form:

$$Q_2^T H Q_2 = T = \begin{array}{|c} \square \\ \hline \end{array}$$

Optional. Balancing (pre-processing step) and eigenvalue reordering (post-processing step).

Focus of this talk on parallel algorithms for performing Step 2.

Chasing multiple bulges in parallel

An implicit shifted QR iteration

Assume H is Hessenberg (after having performed Step 1).
Iteration of QR algorithm chooses $k \ll n$ shifts

$$\text{sigma} = \text{eig}(H(n-k+1:n, n-k+1:n))$$

Usually $k = 2$. Construct 1st col of shift polynomial

$$v = (H - \sigma_1 I)(H - \sigma_2 I)e_1 = \begin{bmatrix} X \\ X \\ X \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

An implicit shifted QR iteration

Q_0 = Householder reflection with $Q_0 v = \gamma e_1$.

$$H \rightarrow Q_0^T H Q_0 = \begin{bmatrix} \hat{X} & \hat{X} & \hat{X} & \hat{X} & \hat{X} & \hat{X} & \hat{X} & \dots \\ \hat{X} & \hat{X} & \hat{X} & \hat{X} & \hat{X} & \hat{X} & \hat{X} & \dots \\ \hat{X} & \hat{X} & \hat{X} & \hat{X} & \hat{X} & \hat{X} & \hat{X} & \dots \\ \hat{X} & \hat{X} & \hat{X} & X & X & X & X & \dots \\ 0 & 0 & 0 & X & X & X & X & \dots \\ 0 & 0 & 0 & 0 & X & X & X & \dots \\ 0 & 0 & 0 & 0 & 0 & X & X & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Hessenberg structure of H disturbed by **bulge**.

An implicit shifted QR iteration

Chase the bulge by Householder reflections:

$$H \rightarrow \begin{bmatrix}
 X & \widehat{X} & \widehat{X} & \widehat{X} & X & X & X & \dots \\
 \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \dots \\
 0 & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \dots \\
 0 & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \dots \\
 0 & \widehat{X} & \widehat{X} & \widehat{X} & X & X & X & \dots \\
 0 & 0 & 0 & 0 & X & X & X & \dots \\
 0 & 0 & 0 & 0 & 0 & X & X & \dots \\
 \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots
 \end{bmatrix} \rightarrow \begin{bmatrix}
 X & X & \widehat{X} & \widehat{X} & \widehat{X} & X & X & \dots \\
 X & X & \widehat{X} & \widehat{X} & \widehat{X} & X & X & \dots \\
 0 & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \dots \\
 0 & 0 & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \dots \\
 0 & 0 & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \dots \\
 0 & 0 & \widehat{X} & \widehat{X} & \widehat{X} & X & X & \dots \\
 0 & 0 & 0 & 0 & 0 & X & X & \dots \\
 \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots
 \end{bmatrix} \rightarrow \dots$$

Chasing multiple bulges

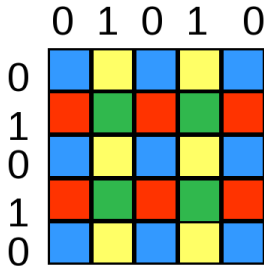
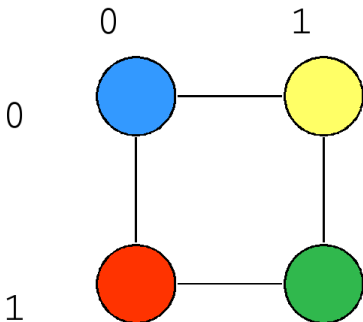
Key to efficient serial and parallel implementations: **introduce another bulge before chasing off the first one.**

$$H \rightarrow \begin{bmatrix} \hat{X} & \hat{X} & \hat{X} & X & X & X & X & \dots \\ \hat{X} & \hat{X} & \hat{X} & \hat{X} & \hat{X} & \hat{X} & \hat{X} & \dots \\ \hat{X} & \hat{X} & \hat{X} & \hat{X} & \hat{X} & \hat{X} & \hat{X} & \dots \\ \hat{X} & \hat{X} & \hat{X} & \hat{X} & \hat{X} & \hat{X} & \hat{X} & \dots \\ 0 & 0 & 0 & X & X & X & X & \dots \\ 0 & 0 & 0 & X & X & X & X & \dots \\ 0 & 0 & 0 & X & X & X & X & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

- Serial: [Braman/Byers/Mathias'02], [Lang'02].
- Parallel: [van de Geijn'93], [Watkins'94], [Henry/Watkins/Dongarra'02] and others.

Chasing multiple bulges in parallel

On distributed memory machines, assume ScaLAPACK's block cyclic distribution of H across $P_r \times P_c$ MPI process mesh.



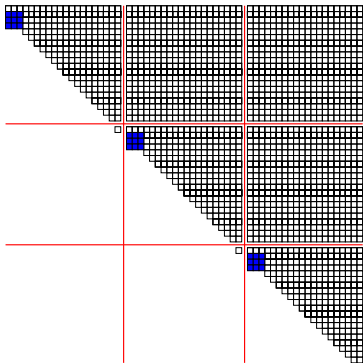
H

Chasing multiple bulges in parallel

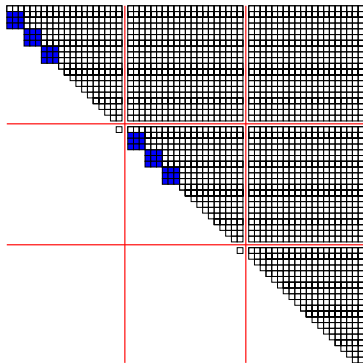
ScaLAPACK: At most one bulge per diagonal block in the process mesh. **Loosely coupled bulges**.

New implementation: Chains of **tightly coupled bulges**.

ScaLAPACK

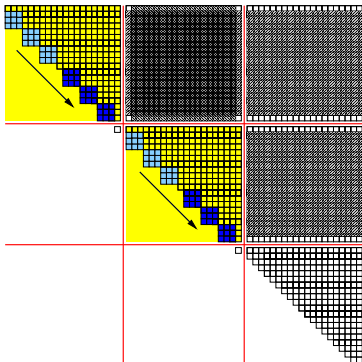


New implementation



Chasing multiple bulges in parallel

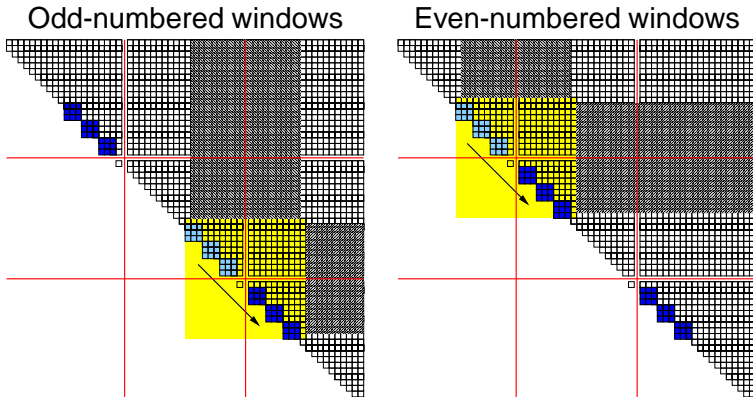
In new implementation, entire chains of bulges are chased.



An intra-block chase of bulge chains.

Chasing multiple bulges in parallel

In new implementation, entire chains of bulges are chased.



An inter-block (cross process border) chase of bulge chains.

Summary

- Chains of bulges (instead of individual bulges) allow to utilize **level 3 BLAS**.
Large message size for cross-border communication \rightsquigarrow less penalty from latencies.
- Multithreading on each node using OpenMP directives.
- Decreases wall clock execution time by a factor of 5–6 for intro example, in comparison to existing ScaLAPACK code (combined with multithreading on each node).

Aggressive early deflation in parallel

Deflation strategies

Classical deflation criterion in **ScaLAPACK**:

$$|h_{i+1,i}| \leq \mathbf{u} \max\{|h_{i,j}|, |h_{i+1,i+1}|\},$$

with \mathbf{u} unit roundoff (double precision $\mathbf{u} \approx 1.1 \times 10^{-16}$).

Aggressive early deflation (AED) strategy by Braman, Byers, and Mathias (2002) in **new implementation**. Often detects convergence much earlier and significantly reduces average #shifts to deflate an eigenvalue.

Illustration of AED

Choose deflation window size $n_{\text{win}} \ll n$ and partition H :

$$H = \begin{matrix} & & n-n_{\text{win}}-1 & 1 & n_{\text{win}} \\ & & & & \\ n-n_{\text{win}}-1 & & \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ 0 & H_{32} & H_{33} \end{bmatrix} \\ 1 & & & & \\ n_{\text{win}} & & & & \end{matrix}.$$

Compute Schur decomposition of H_{33} and update rest of H :

$$H \rightarrow \begin{bmatrix} \vdots & \vdots & & & & & \\ \dots & X & X & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} \\ \dots & X & X & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} \\ \hline \dots & 0 & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} \\ \dots & 0 & \widehat{X} & 0 & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} \\ \dots & 0 & \widehat{X} & 0 & 0 & \widehat{X} & \widehat{X} & \widehat{X} \\ \dots & 0 & \widehat{X} & 0 & 0 & 0 & \widehat{X} & \widehat{X} \\ \dots & 0 & \widehat{X} & 0 & 0 & 0 & 0 & \widehat{X} \end{bmatrix}$$

Illustration of AED

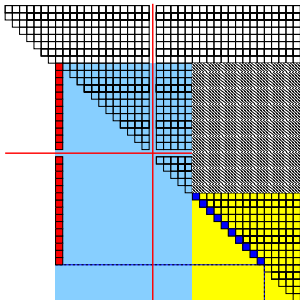
- Search for tiny trailing elements of newly introduced **spike** under different eigenvalue orderings of H_{33} .
- Return undeflatable part back to Hessenberg form.

For example, if 2 eigenvalues could be deflated:

$$H \rightarrow \left[\begin{array}{cc|ccccc} & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \dots & X & X & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} \\ \dots & X & X & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} \\ \hline \dots & 0 & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} \\ \dots & 0 & \widehat{X} & 0 & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} \\ \dots & 0 & \widehat{X} & 0 & 0 & \widehat{X} & \widehat{X} & \widehat{X} \\ \dots & 0 & 0 & 0 & 0 & 0 & \widehat{X} & \widehat{X} \\ \dots & 0 & 0 & 0 & 0 & 0 & 0 & \widehat{X} \end{array} \right] \rightarrow \left[\begin{array}{cc|ccc|cc} & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \dots & X & X & \widehat{X} & \widehat{X} & \widehat{X} & X & X \\ \dots & X & X & \widehat{X} & \widehat{X} & \widehat{X} & X & X \\ \hline \dots & 0 & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} \\ \dots & 0 & \widehat{0} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} \\ \dots & 0 & \widehat{0} & 0 & \widehat{X} & \widehat{X} & \widehat{X} & \widehat{X} \\ \hline \dots & 0 & 0 & 0 & 0 & 0 & X & X \\ \dots & 0 & 0 & 0 & 0 & 0 & 0 & X \end{array} \right]$$

AED in parallel

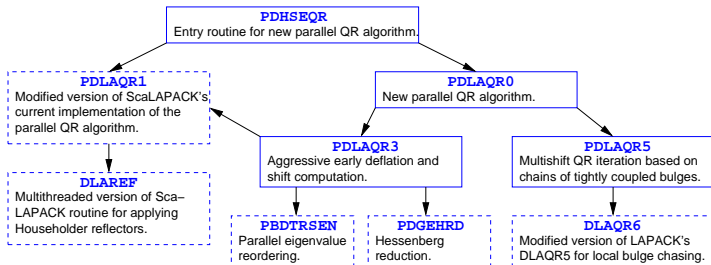
Reorganize AED s.t. groups of eigenvalues are reordered.



- Reordering performed locally in **computational window**.
- Afterwards, entire group of **undeflatable eigenvalues** reordered to top left corner with a parallel algorithm described in [Granat/Kågström/Kressner'09].

Overview of new implementation

Routines and dependencies

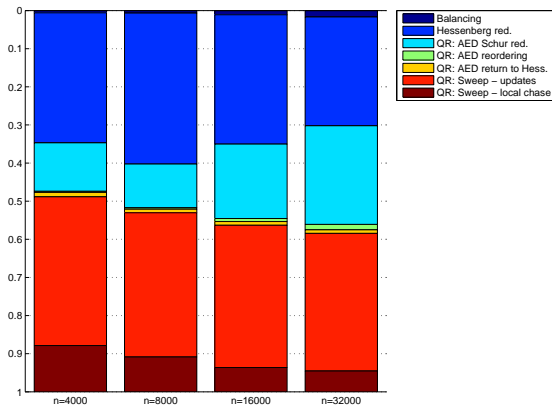


- ScaLAPACK-like implementation based on BLACS and parallel BLAS.
- Local multithreading on each node by mixing parallel MPI with OpenMP directives.

Numerical experiments

$P_r \times P_c \times P_t$	$n =$							
	4 000		8 000		16 000		32 000	
$1 \times 1 \times 1$	9332	226	79943	1478				
$2 \times 2 \times 1$	2761	112	20161	640				
$4 \times 4 \times 1$	1174	69	8582	265	67779	1644		
$6 \times 6 \times 1$	697	60	5192	194	32920	1007	∞	6218
$8 \times 8 \times 1$	418	57	3671	152	20856	595	∞	4164
$10 \times 10 \times 1$	368	63	2589	165	16755	516	∞	3046
$1 \times 1 \times 4$	2592	173	18324	913				
$2 \times 2 \times 4$	1617	126	11032	591				
$3 \times 3 \times 4$	834	102	5692	408	38834	2327		
$4 \times 4 \times 4$	612	76	3986	277	25823	1332	∞	9250
$5 \times 5 \times 4$	474	70	2971	203	18934	1061	∞	6568

Execution times of **ScaLAPACK** vs. **new implementation** (in seconds).



Profile of execution time for new implementation; memory load corresponds to a $4\,000 \times 4\,000$ submatrix per core.

$$n = 100.000$$

- 1024 cores organized in 32×32 process grid
- Total execution time **8h 30min**
- balancing = 20min
- reduction to Hessenberg form = 1h 7min
- QR algorithm = 7h 3min
 - 12 QR iterations with $\approx 3000 - 6000$ shifts
 - $n_{\text{win}} = 6145$
 - **80% of execution time spent on AED!**
 - 0.44 shifts per deflated eigenvalue

Conclusions

- New parallel implementation significantly outperforms existing ScaLAPACK implementation.
- Parallel QR iterations by chasing several bulge chains.
- Parallel AED by algorithmic reformulation.
- Tremendous implementation effort.
- AED becomes a serious bottleneck for $\gg O(10^4)$ cores.
- R. Granat, B. Kågström, and DK. Parallel eigenvalue reordering in real Schur forms. *Concurrency and Computation: Practice and Experience*, 2008. To appear.
- R. Granat, B. Kågström, and DK. A novel parallel QR algorithm for hybrid distributed memory HPC systems. Technical report 2009-15, Seminar for applied mathematics, ETH Zurich, 2009.
- B. Adlerborn, DK, and B. Kågström. Parallel variants of the multishift QZ algorithm with advanced deflation techniques. In B. Kågström et al., editor, *Applied Parallel Computing - State of the Art in Scientific Computing (PARA'06)*, volume 4699, pages 117–126. Lecture Notes in Computer Science, Springer, 2007.