High-Performance Algorithms for Computing The Sign of a Triangular Matrix (updated)

Vadim Stotland and Sivan Toledo
Blavatnik School of Computer Science
Tel-Aviv University

Oded Schwartz
Hebrew University

SIAM Linear Algebra 2015

Schur-Parlett Algorithms for $f(A)$

If A is upper triangular then $f(A)$ is also upper triangular

Can use the Schur decomposition
$A = QTQ^* \longrightarrow f(A) = Qf(T)Q^*$

Also, $Af(A) = f(A)A$ (matrices commute with their functions)

Parlett discovered recurrence for $U = f(T)$ of a triangular T

$$\begin{aligned} u_{ii} &= f(t_{ii}) \\ u_{ij} &= t_{ij}\frac{u_{ii}-u_{jj}}{t_{ii}-t_{jj}} - \frac{\sum_{k=i+1}^{j-1} u_{ik}t_{kj} - t_{ik}u_{kj}}{t_{ii}-t_{jj}} \end{aligned}$$

Breakdown/instability when $t_{ii} \approx t_{jj}$ for some i, j

Higham's Stable Version for the Sign Function

Higham discovered that for the sign function of a triangular matrix, another recurrence is stable exactly when Parlett's is not: $f(T)f(T) = U^2 = I$; this leads to the stable recurrence

$$u_{ij} = \begin{cases} t_{ij}\frac{u_{ii}-u_{jj}}{t_{ii}-t_{jj}} - \frac{\sum_{k=i+1}^{j-1} u_{ik}t_{kj}-t_{ik}u_{kj}}{t_{ii}-t_{jj}} & u_{ii}+u_{jj}=0 \\ -\frac{\sum_{k=i+1}^{j-1} u_{ik}u_{kj}}{f_{ii}+f_{jj}}, & u_{ii}+u_{jj}\neq 0 \end{cases}$$

Flop count is $(1/3)n^3$ to $(2/3)n^3$

Two Different Approaches for Communication-Efficient Matrix Signs

The Block Version of Schur-Parlett Leads to Sylvester Equations

A block version of the same recurrence,

$$\begin{aligned} U_{ii} &= f(T_{ii}) \\ T_{ii}U_{ij} - U_{ij}T_{jj} &= U_{ii}T_{ij} - T_{ij}U_{jj} + \sum_{k=i+1}^{j-1} (U_{ik}T_{kj} - T_{ik}U_{kj}) \end{aligned}$$

The second equation is a Sylvester equation from which we can compute U_{ij}

The Sylvester equation is non-singular iff T_{ii}, T_{jj} have no eigenvalues in common

A framework by Davies and Higham reorders the Schur form and partitions T so that the eigenvalues of T_{ii}, T_{jj} are far enough

The Blocked Approach

(1) Reorder the Schur form $T = P\bar{T}P^*$ such that

$$\mathrm{diag}(\bar{T}) = \begin{bmatrix} 1 & 1 & \cdots & 1 & -1 & \cdots & -1 \end{bmatrix}$$

or $A = (QP)\bar{T}(QP)^*$

(2) $U_{11} = f(\bar{T}_{11}) = I$, $U_{22} = f(\bar{T}_{22}) = -I$

(3) Compute U_{12} by solving the Sylvester equation

$$\bar{T}_{11}U_{12} - U_{12}\bar{T}_{22} = U_{11}\bar{T}_{12} - \bar{T}_{12}U_{22} = 2\bar{T}_{12}$$

Subroutines in the Blocked Algorithm

Schur-form reordering: xTRSEN in LAPACK (Bai and Demmel 1993)
Blocked version by Kressner 2006 (not in LAPACK but he gave us the code)
Flop count is $12nk$, between $12n^2$ and $3n^3$, depending on n_-, n_+, and ordering of diag(U)

xTRSYL (Bartels-Stewart) for the Sylvester equation ($n_- n_+ (n_- + n_+)$ flops, between n^2 and $n^3/4$)
Recursive cache-efficient version (RECSY) by Jonsson and Kagstrom, 2003, 2009

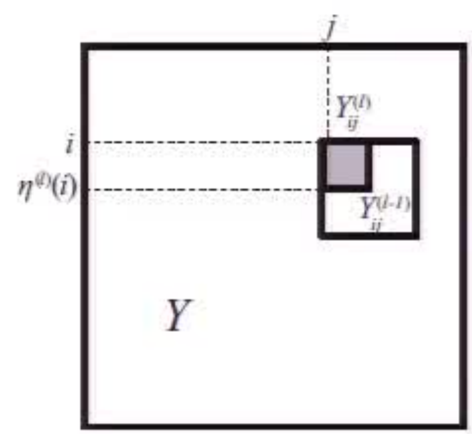
Total is quadratic (!!!) if there are few positive/negative eigenvalues, $3n^3$ in the worst case (4.5X relative to Higham)

A Recursive Higham Recurrence

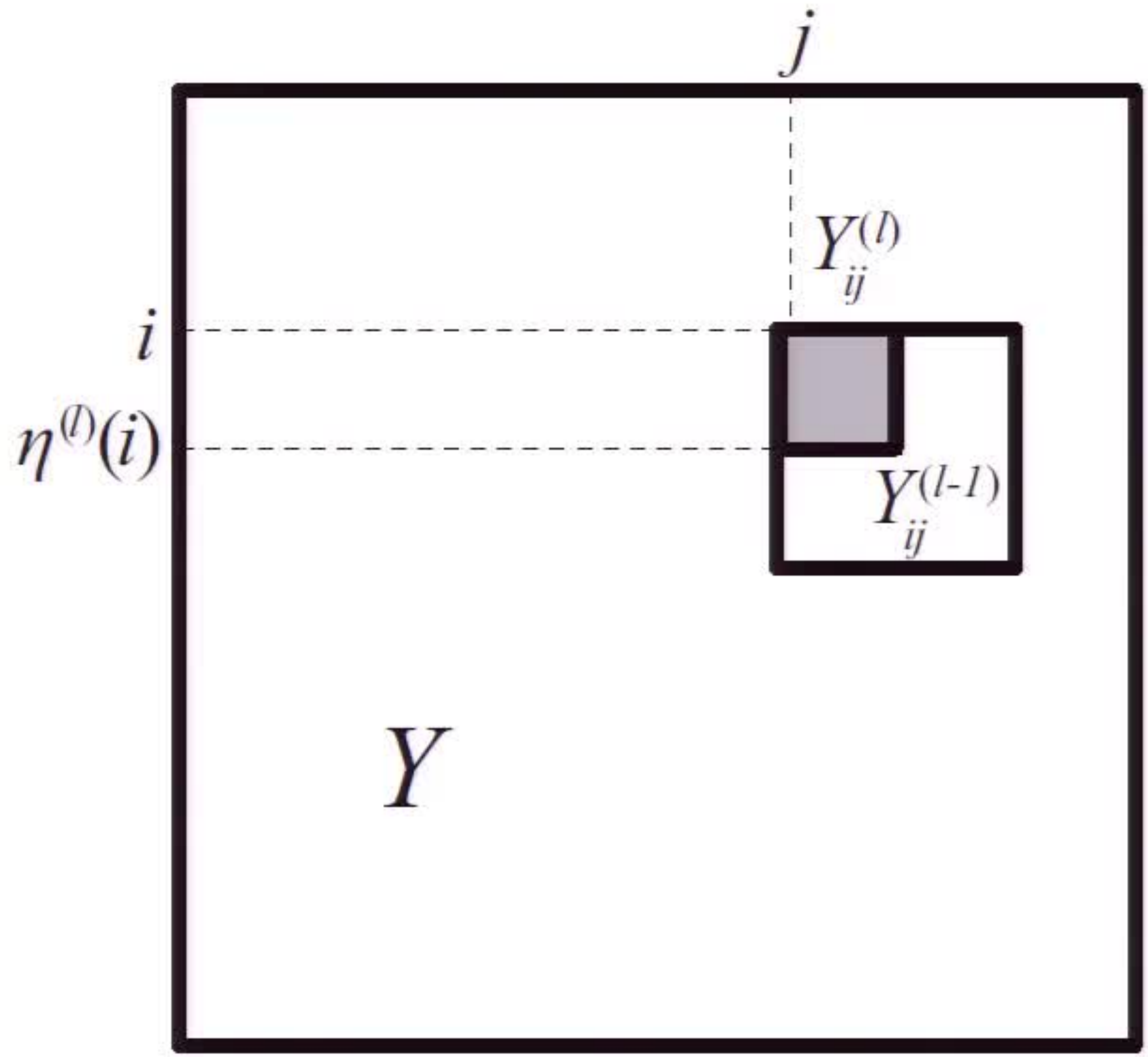
In 2014: a simple looking formulation; apparently correct code; but could not prove correctness

Now: a more complex formulation, but with a correctness proof (and easier to code) based on

$$
\begin{aligned}
(TU)_{ij}^{(\ell)} &= (UT)_{ij}^{(\ell)} \\
(UU)_{ij}^{(\ell)} &= I_{ij}^{(\ell)} .
\end{aligned}
$$



Notation



Block Recurrences

$$
\begin{aligned}
(TU)_{ij}^{(\ell)} &= (UT)_{ij}^{(\ell)} \\
(UU)_{ij}^{(\ell)} &= I_{ij}^{(\ell)} .
\end{aligned}
$$

expand into

$$
\begin{aligned}
T_{ii}^{(\ell)} U_{ij}^{(\ell)} - U_{ij}^{(\ell)} T_{jj}^{(\ell)} &= U_{ii}^{(\ell)} T_{ij}^{(\ell)} - T_{ij}^{(\ell)} U_{jj}^{(\ell)} \\
&\quad + \sum_{k=\eta^{(\ell)}(i)}^{j-1} \left(U_{ik}^{(\ell)} T_{kj}^{(\ell)} - T_{ik}^{(\ell)} U_{kj}^{(\ell)} \right) \\
U_{ii}^{(\ell)} U_{ij}^{(\ell)} + U_{ij}^{(\ell)} U_{jj}^{(\ell)} &= I_{ij}^{(\ell)} - \sum_{k=\eta^{(\ell)}(i)}^{j-1} U_{ik}^{(\ell)} U_{kj}^{(\ell)} .
\end{aligned}
$$

Auxiliary Matrices to Represent the Sums

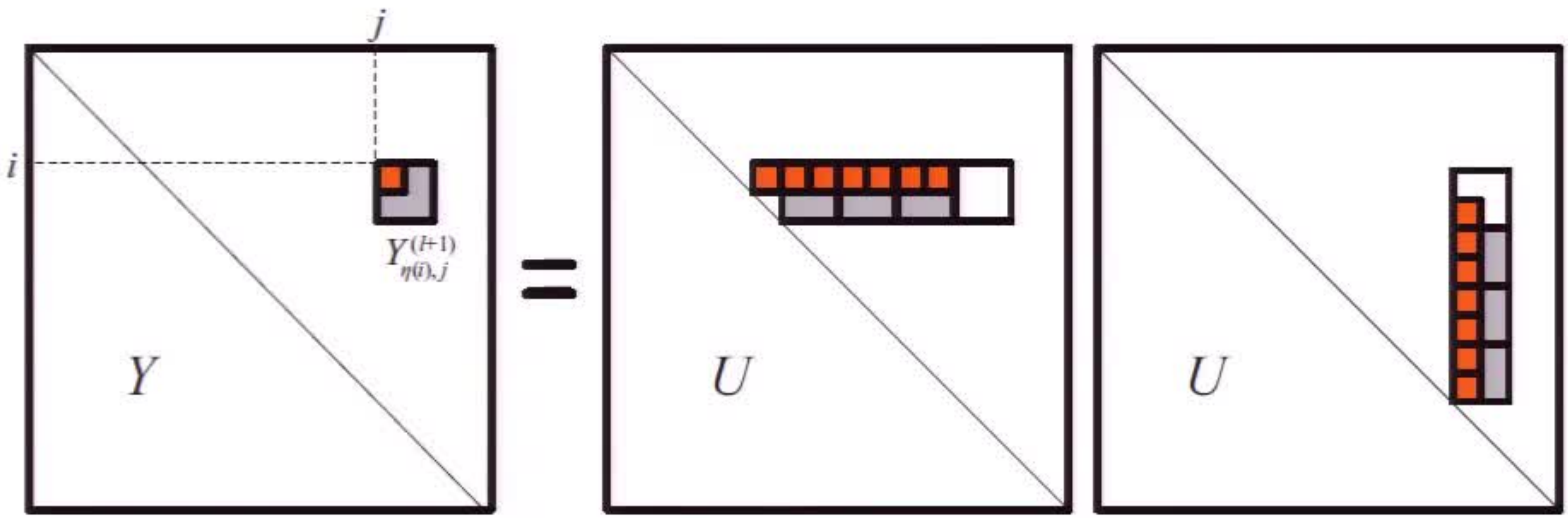
Define

$$X_{ij}^{(\ell)} = \sum_{k=\eta^{(\ell)}(i)}^{j-1} \left(U_{ik}^{(\ell)} T_{kj}^{(\ell)} - T_{ik}^{(\ell)} U_{kj}^{(\ell)} \right)$$

$$Y_{ij}^{(\ell)} = \sum_{k=\eta^{(\ell)}(i)}^{j-1} U_{ik}^{(\ell)} U_{kj}^{(\ell)}$$

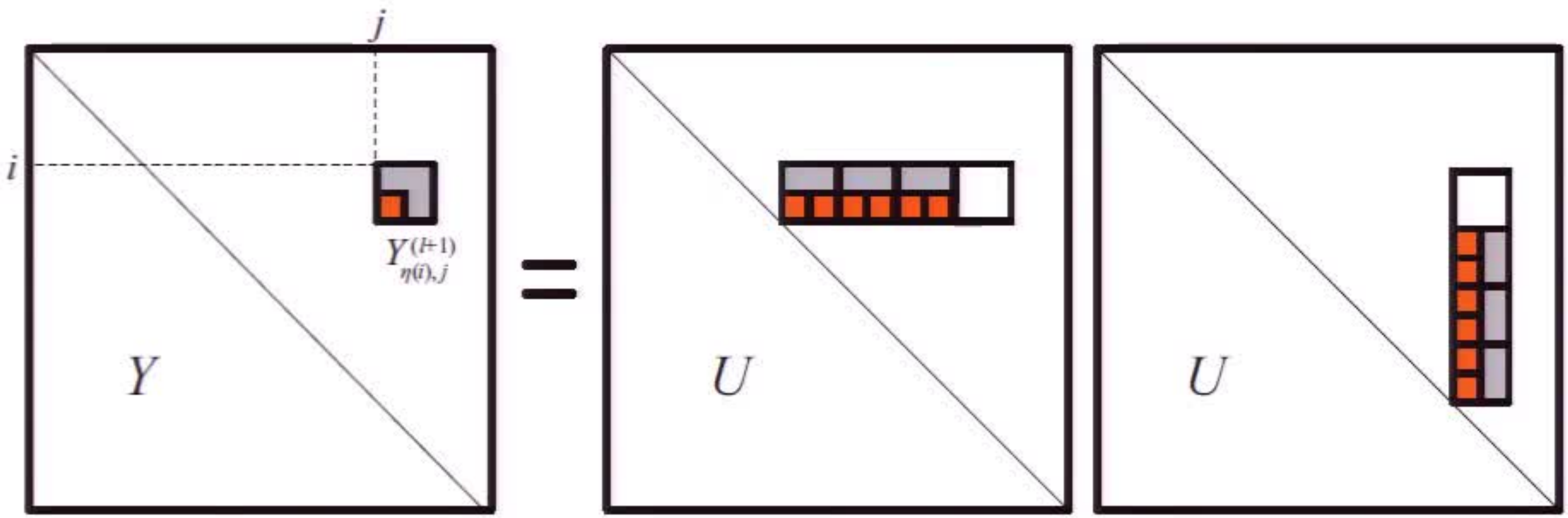
Algorithm computes blocks of X, Y, then the corresponding block of U

Computing the (1,1) Block



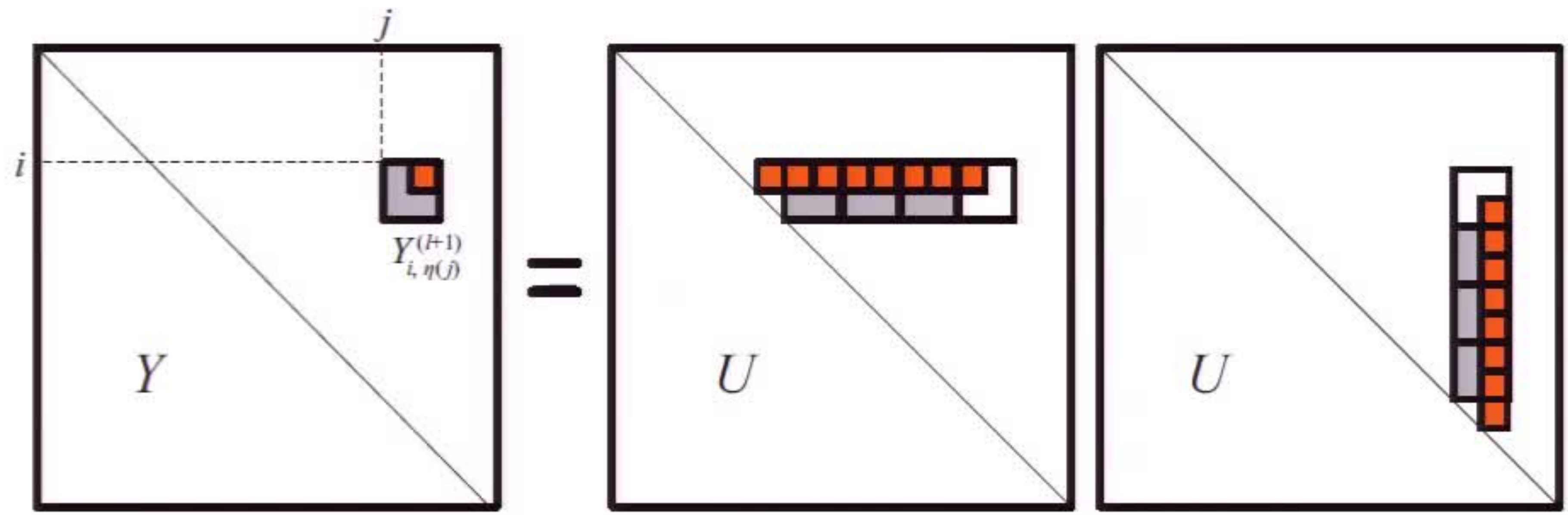
These blocks depend on the just-computed (2,1) block of U

Computing the (2,1) Blocks (Easy Ones)



$Y^{(\ell+1)}_{\eta^{(\ell+1)}(i),j}$ is simply part of the already-computed $Y^{(\ell)}_{i,j}$; same for $X^{(\ell+1)}_{\eta^{(\ell+1)}(i),j}$; can compute $Y^{(\ell+1)}_{\eta^{(\ell+1)}(i),j}$

Computing the (1,2) Block



Depend on both (1,1) and (2,2)

Reducing Arithmetic

The actual computations of u_{ij} are at the bottom of the recursion and they use *either* x_{ij} *or* y_{ij}

Which will be used is known a-priori; depends on sign of $u_{ii} + u_{jj}$

Can implement (and we have) modified matrix multiplication that only contributes to x_{ij} *or* y_{ij}

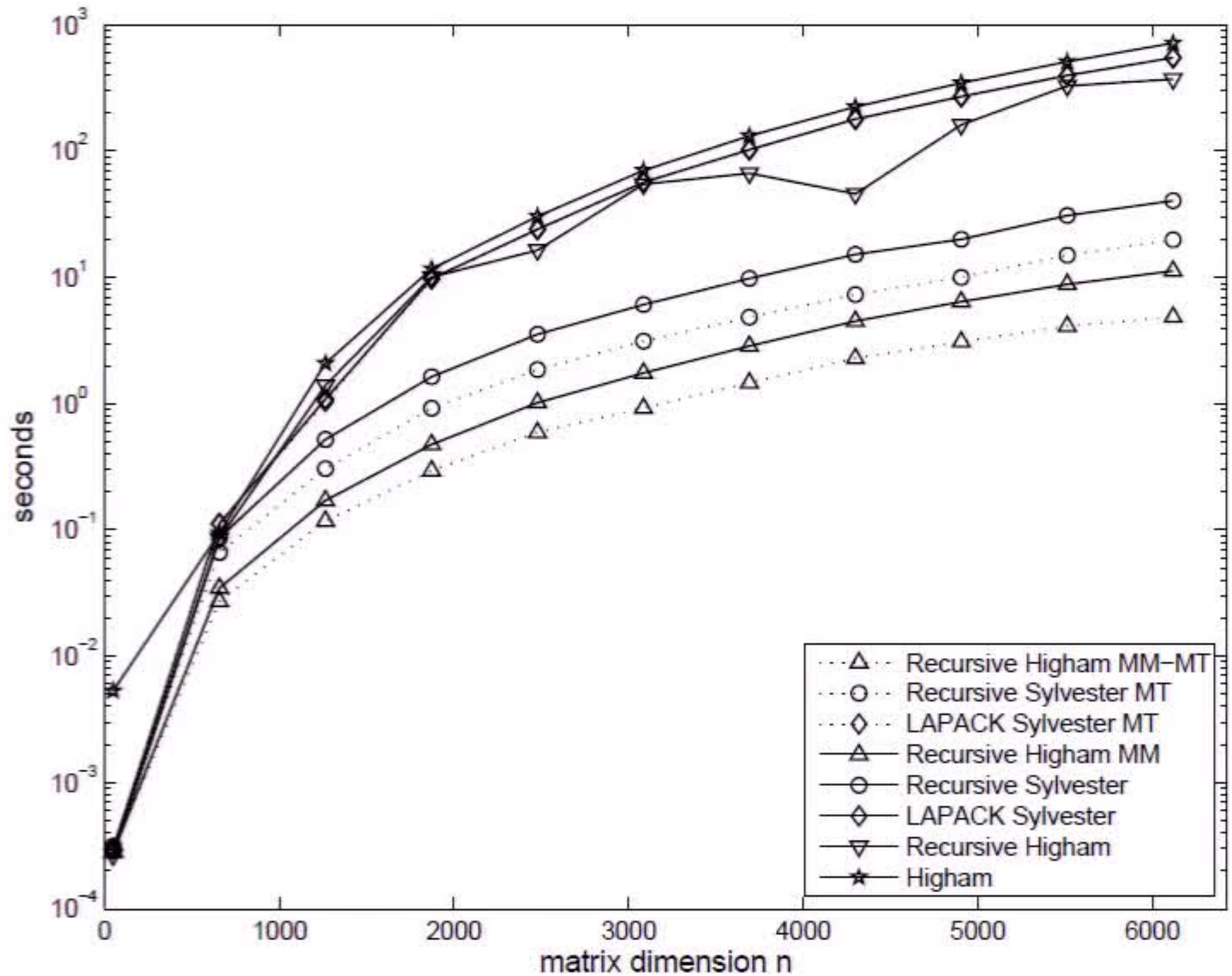
Likely to slow down in practice because we can't use xGEMM

Experimental Results

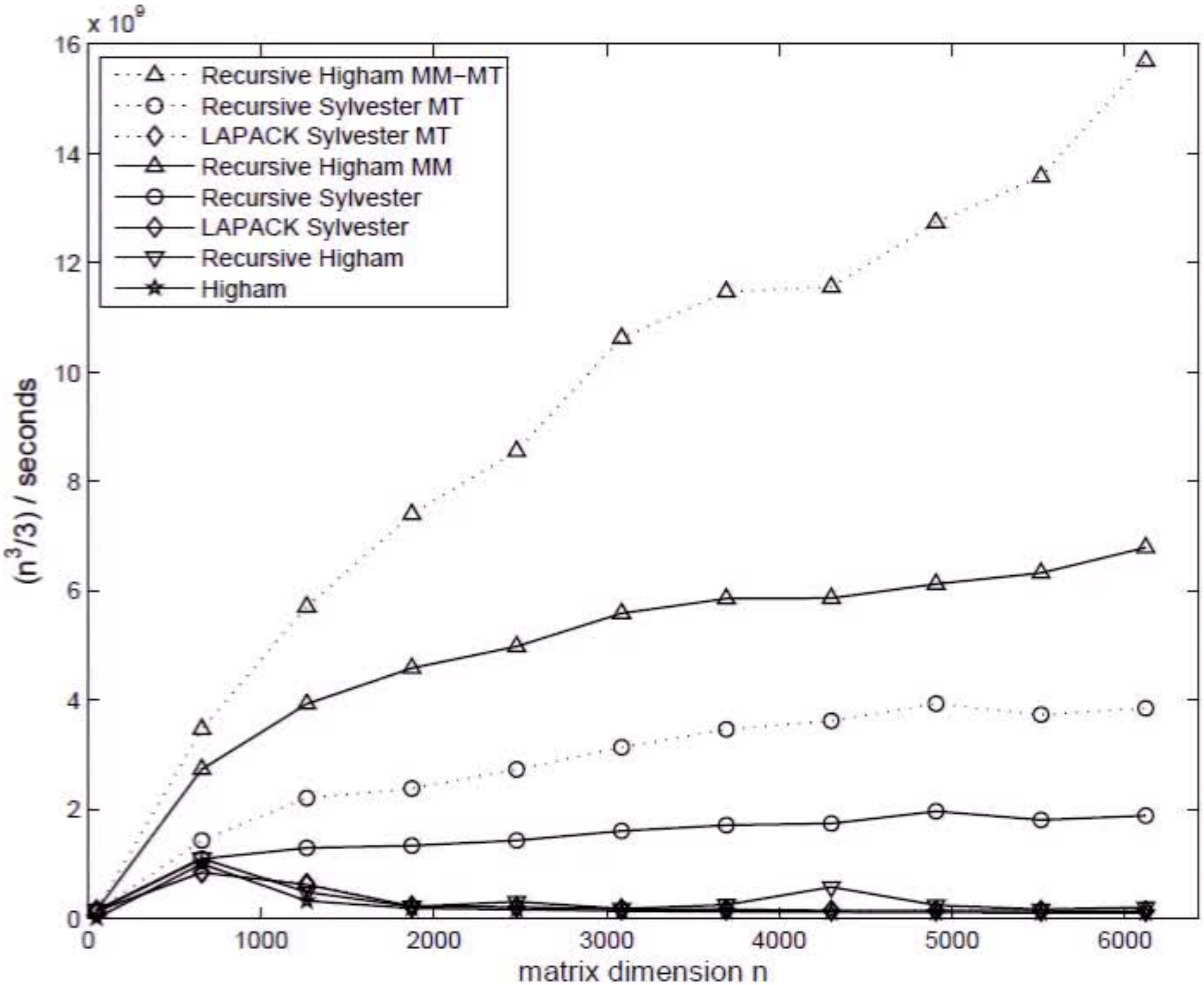
Test set: real part of Schur form of random matrices
$E[n_+] = E[n_-] = n/2$
Variance of the inertia is insignificant, so in all cases
$n_+ \approx n_- \approx n/2$

In later experiments we flip the sign of eigenvalues in these matrices so that exactly 3 are negative, in random positions.

Performance Comparison for $n_+ \approx n_- \approx n/2$



Performance Comparison for $n_+ \approx n_- \approx n/2$



$n_+ \approx n_- \approx n/2$: Observations

Recursive Higham-Parlett with xGEMM is fastest (Sylvester solver was in AN14)

Recursive Sylvester next best

Multithreaded xGEMM help both

Really slow (factor to 20 to 70):

- Trying to compute either x_{ij} or y_{ij}
- The original Higham-Parlett
- Sylvester using LAPACK routines

$n_- = 3$: Observations

The quadratic Schur-reordering+Sylvester methods are orders of magnitude faster