Robust Task-Parallel Solution of the Triangular Sylvester Equation

Angelika Schwarz, Carl Christian Kjelgaard Mikkelsen

PPAM 2019

UMEÅ UNIVERSITY

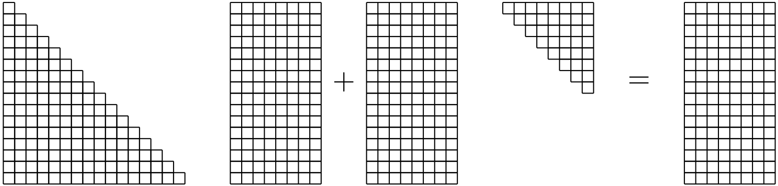
Triangular Sylvester Equation

Solve

$$\underset{(m \times m)}{\boldsymbol{A}} \quad \underset{(m \times n)}{\boldsymbol{X}} + \underset{(m \times n)}{\boldsymbol{X}} \quad \underset{(n \times n)}{\boldsymbol{B}} = \underset{(m \times n)}{\boldsymbol{C}}$$

where $\boldsymbol{A}$, $\boldsymbol{B}$ are triangular. The unique solution $\boldsymbol{X}$ exists iff

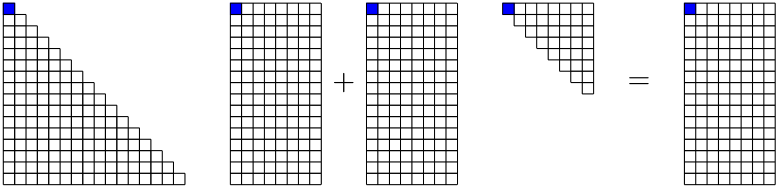
$$\sigma(\boldsymbol{A}) \cap \sigma(-\boldsymbol{B}) = \emptyset.$$



We store $\boldsymbol{X}$ on top of $\boldsymbol{C}$.

Scalar solution of the triangular Sylvester equation

$$a_{ii}x_{ij} + x_{ij}b_{jj} = c_{ij} - \sum_{k=1}^{i-1} a_{ik}x_{kj} - \sum_{k=1}^{j-1} x_{ik}b_{kj}$$

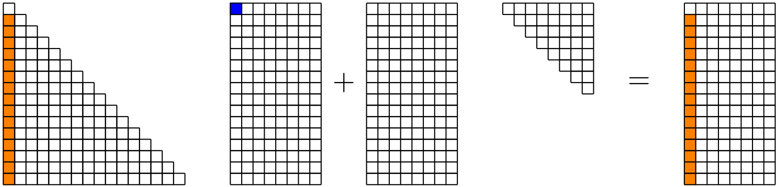


Solve

$a_{11}x_{11} + x_{11}b_{11} = c_{11}$

Scalar solution of the triangular Sylvester equation

$$a_{ii}x_{ij} + x_{ij}b_{jj} = c_{ij} - \sum_{k=1}^{i-1} a_{ik}x_{kj} - \sum_{k=1}^{j-1} x_{ik}b_{kj}$$



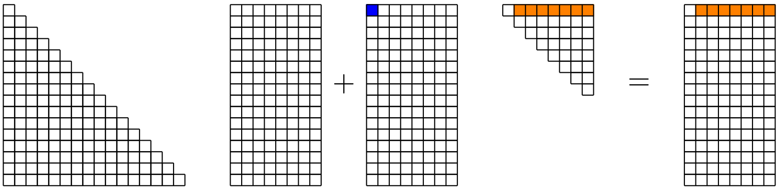
Update:

for $k \leftarrow 2 : m$

$\quad c_{k,1} \leftarrow c_{k,1} - a_{k,1}x_{11}$

Scalar solution of the triangular Sylvester equation

$$a_{ii}x_{ij} + x_{ij}b_{jj} = c_{ij} - \sum_{k=1}^{i-1} a_{ik}x_{kj} - \sum_{k=1}^{j-1} x_{ik}b_{kj}$$



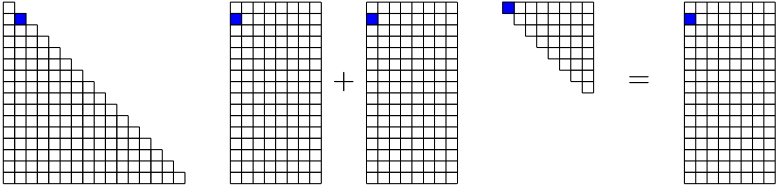
Update:

for $\ell \leftarrow 2 : n$

$\quad c_{1,\ell} \leftarrow c_{1,\ell} - x_{11}b_{1,\ell}$

Scalar solution of the triangular Sylvester equation

$$a_{ii}x_{ij} + x_{ij}b_{jj} = c_{ij} - \sum_{k=1}^{i-1} a_{ik}x_{kj} - \sum_{k=1}^{j-1} x_{ik}b_{kj}$$

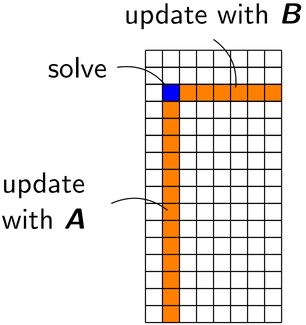


Solve

$a_{22}x_{21} + x_{21}b_{11} = c_{21}$

Scalar algorithm

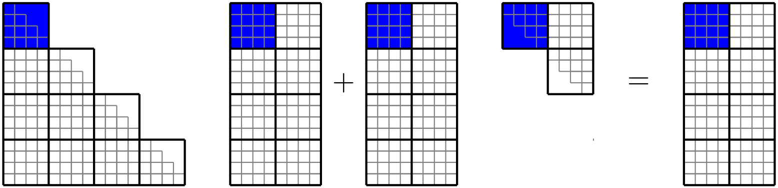
$$a_{ii}x_{ij} + x_{ij}b_{jj} = c_{ij} - \sum_{k=1}^{i-1} a_{ik}x_{kj} - \sum_{k=1}^{j-1} x_{ik}b_{kj}$$



for *row* $i \leftarrow 1, 2, \ldots, m$
    for *column* $j \leftarrow 1, 2, \ldots, n$
        Solve $a_{ii}x_{ij} + x_{ij}b_{jj} = c_{ij}$
        $c_{ij} \leftarrow x_{ij}$
        for $k \leftarrow i+1, i+2, \ldots, m$
            $c_{kj} \leftarrow c_{kj} - a_{ki}x_{ij}$
        for $\ell \leftarrow j+1, j+2, \ldots, n$
            $c_{i\ell} \leftarrow c_{i\ell} - x_{ij}b_{j\ell}$
return C

Tiled solution of the triangular Sylvester equation

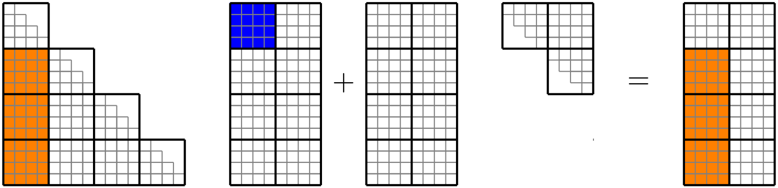
$$\boldsymbol{A}_{ii}\boldsymbol{X}_{ij} + \boldsymbol{X}_{ij}\boldsymbol{B}_{jj} = \boldsymbol{C}_{ij} - \sum_{k=1}^{i-1} \boldsymbol{A}_{ik}\boldsymbol{X}_{kj} - \sum_{k=1}^{j-1} \boldsymbol{X}_{ik}\boldsymbol{B}_{kj}$$



Solve with inner triangular Sylvester equation solver $(\textsc{Sylv})$
$\boldsymbol{A}_{11}\boldsymbol{X}_{11} + \boldsymbol{X}_{11}\boldsymbol{B}_{11} = \boldsymbol{C}_{11}$

Tiled solution of the triangular Sylvester equation

$$\boldsymbol{A}_{ii}\boldsymbol{X}_{ij} + \boldsymbol{X}_{ij}\boldsymbol{B}_{jj} = \boldsymbol{C}_{ij} - \sum_{k=1}^{i-1}\boldsymbol{A}_{ik}\boldsymbol{X}_{kj} - \sum_{k=1}^{j-1}\boldsymbol{X}_{ik}\boldsymbol{B}_{kj}$$



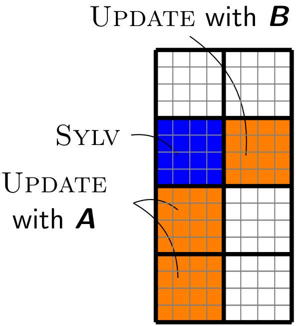
Tile updates with DGEMM

for $k \leftarrow 2 : m/b$ // b: tile size

$\quad \boldsymbol{C}_{k1} \leftarrow \boldsymbol{C}_{k1} - \boldsymbol{A}_{k1}\boldsymbol{X}_{11}$

Tiled algorithm

$$\boldsymbol{A}_{ii}\boldsymbol{X}_{ij} + \boldsymbol{X}_{ij}\boldsymbol{B}_{jj} = \boldsymbol{C}_{ij} - \sum_{k=1}^{i-1} \boldsymbol{A}_{ik}\boldsymbol{X}_{kj} - \sum_{k=1}^{j-1} \boldsymbol{X}_{ik}\boldsymbol{B}_{kj}$$



for *tile row* $i \leftarrow 1, 2, \ldots, m/b$
    for *tile column* $j \leftarrow 1, 2, \ldots, n/b$
        Solve $\boldsymbol{A}_{ii}\boldsymbol{X}_{ij} + \boldsymbol{X}_{ij}\boldsymbol{B}_{jj} = \boldsymbol{C}_{ij}$ (SYLV)
        $\boldsymbol{C}_{ij} \leftarrow \boldsymbol{X}_{ij}$
        for $k \leftarrow i+1, i+2, \ldots, m/b$
            $\boldsymbol{C}_{kj} \leftarrow \boldsymbol{C}_{kj} - \boldsymbol{A}_{ki}\boldsymbol{C}_{ij}$ (UPDATE)
        for $\ell \leftarrow j+1, j+2, \ldots, n/b$
            $\boldsymbol{C}_{i\ell} \leftarrow \boldsymbol{C}_{i\ell} - \boldsymbol{C}_{ij}\boldsymbol{B}_{j\ell}$ (UPDATE)
return $\boldsymbol{C}$

Equivalent Linear System

$$\underset{(m\times m)}{\boldsymbol{A}} \quad \underset{(m\times n)}{\boldsymbol{X}} \quad + \quad \underset{(m\times n)}{\boldsymbol{X}} \quad \underset{(n\times n)}{\boldsymbol{B}} \quad = \quad \underset{(m\times n)}{\boldsymbol{C}}$$

$$\left(\boldsymbol{I}_n \otimes \boldsymbol{A} + \boldsymbol{B}^T \otimes \boldsymbol{I}_m\right) \operatorname{vec} \boldsymbol{X} = \operatorname{vec} \boldsymbol{C}$$

Example. $m = n = 3$.

$$\left(\left(\begin{array}{c|c|c} \boldsymbol{A} & & \\ \hline & \boldsymbol{A} & \\ \hline & & \boldsymbol{A} \end{array} \right) + \left(\begin{array}{c|c|c} b_{11}\boldsymbol{I}_3 & & \\ \hline b_{12}\boldsymbol{I}_3 & b_{22}\boldsymbol{I}_3 & \\ \hline b_{13}\boldsymbol{I}_3 & b_{23}\boldsymbol{I}_3 & A\boldsymbol{I}_3 \end{array} \right) \right) \operatorname{vec} \boldsymbol{X} = \operatorname{vec} \boldsymbol{C}$$

Consider $\boldsymbol{A} = \boldsymbol{B}^T = \frac{1}{4} \begin{pmatrix} 1 & & \\ -4 & 1 & \\ -4 & -4 & 1 \end{pmatrix}, \boldsymbol{C} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$

Growth in the solution entries

$$\left(\begin{array}{ccc|ccc|ccc}
\frac{1}{2} & & & & & & & & \\
-1 & \frac{1}{2} & & & & & & & \\
-1 & -1 & \frac{1}{2} & & & & & & \\
\hline
-1 & & & \frac{1}{2} & & & & & \\
 & -1 & & -1 & \frac{1}{2} & & & & \\
 & & -1 & -1 & -1 & \frac{1}{2} & & & \\
\hline
-1 & & & -1 & & & \frac{1}{2} & & \\
 & -1 & & & -1 & & -1 & \frac{1}{2} & \\
 & & -1 & & & -1 & -1 & -1 & \frac{1}{2}
\end{array}\right)
\begin{pmatrix} 2 \\ 4 \\ 12 \\ 4 \\ 16 \\ 64 \\ 12 \\ 64 \\ 304 \end{pmatrix}
=
\begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Reassemble the matrix:

$$\boldsymbol{X} = \begin{pmatrix} 2 & 4 & 12 \\ 4 & 16 & 64 \\ 12 & 64 & 304 \end{pmatrix}$$

You ignore overflow at your own risk!

Scaled triangular Sylvester equation

$$\boldsymbol{A}(\alpha^{-1}\boldsymbol{X}) + (\alpha^{-1}\boldsymbol{X})\boldsymbol{B} = \boldsymbol{C}$$

- $\alpha \in (0, 1]$ effectively extends the floating-point range
- Avoid overflow by *dynamic* downscaling of solution
- Overflow threshold in double-precision $\Omega = 2^{1023}$

correct matrix	**representation in double-precision**	**scaled representation**
$\begin{pmatrix} 2^{1024} & 2^{1023} \\ 0 & 2^{-10} \end{pmatrix}$	$\begin{pmatrix} \infty & 2^{1013} \\ 0 & 2^{-10} \end{pmatrix}$	$\left(\frac{1}{2}\right)^{-1} \begin{pmatrix} 2^{1023} & 2^{1022} \\ 0 & 2^{-11} \end{pmatrix}$

Scaled tiled triangular Sylvester equation

- Associate every tile $\boldsymbol{X}_{ij}$ with one scaling factor $\alpha_{ij} \in (0, 1]$
- Represent tile $\boldsymbol{X}_{ij}$ as scaled tile $\alpha_{ij}^{-1}\boldsymbol{X}_{ij}$
- Solve the tile-locally scaled tiled triangular Sylvester equation

$$\boldsymbol{A}_{ii}(\alpha_{ij}^{-1}\boldsymbol{X}_{ij}) + (\alpha_{ij}^{-1}\boldsymbol{X}_{ij})\boldsymbol{B}_{jj} = \boldsymbol{C}_{ij} - \sum_{k=1}^{i-1}\boldsymbol{A}_{ik}(\alpha_{kj}^{-1}\boldsymbol{X}_{kj}) - \sum_{k=1}^{j-1}(\alpha_{ik}^{-1}\boldsymbol{X}_{ik})\boldsymbol{B}_{kj}$$

$$\begin{array}{|c|c|} \hline \alpha_{11}^{-1}\boldsymbol{X}_{11} & \alpha_{12}^{-1}\boldsymbol{X}_{12} \\ \hline \alpha_{21}^{-1}\boldsymbol{X}_{21} & \alpha_{22}^{-1}\boldsymbol{X}_{22} \\ \hline \alpha_{31}^{-1}\boldsymbol{X}_{31} & \alpha_{32}^{-1}\boldsymbol{X}_{32} \\ \hline \end{array}$$

$$\Downarrow$$

$$\alpha^{-1} \begin{array}{|c|c|} \hline \boldsymbol{X}_{11} & \boldsymbol{X}_{12} \\ \hline \boldsymbol{X}_{21} & \boldsymbol{X}_{22} \\ \hline \boldsymbol{X}_{31} & \boldsymbol{X}_{32} \\ \hline \end{array}$$

- The products $\alpha_{ij}^{-1}\boldsymbol{X}_{ij}$ are never explicitly formed
- Reduce α_{ij} to global scaling factor α

Robust tiled algorithm

$$\boldsymbol{A}_{ii}(\alpha_{ij}^{-1}\boldsymbol{X}_{ij}) + (\alpha_{ij}^{-1}\boldsymbol{X}_{ij})\boldsymbol{B}_{jj} = \boldsymbol{C}_{ij} - \sum_{k=1}^{i-1}\boldsymbol{A}_{ik}(\alpha_{kj}^{-1}\boldsymbol{X}_{kj}) - \sum_{k=1}^{j-1}(\alpha_{ik}^{-1}\boldsymbol{X}_{ik})\boldsymbol{B}_{kj}$$

Set all $\alpha_{ij} \leftarrow 1$
for *tile row* $i \leftarrow 1, 2, \ldots, m/b$
  for *tile column* $j \leftarrow 1, 2, \ldots, n/b$
    Solve $\boldsymbol{A}_{ii}(\beta^{-1}\boldsymbol{X}_{ij}) + (\beta^{-1}\boldsymbol{X})_{ij}\boldsymbol{B}_{jj} = \boldsymbol{C}_{ij}$ (RobustSylv)
    $[\alpha_{ij}, \boldsymbol{C}_{ij}] \leftarrow [\beta, \boldsymbol{X}_{ij}]$
    for $k \leftarrow i+1, i+2, \ldots, m/b$
      $\alpha_{kj}^{-1}\boldsymbol{C}_{kj} \leftarrow \alpha_{kj}^{-1}\boldsymbol{C}_{kj} - \boldsymbol{A}_{ki}(\alpha_{ij}^{-1}\boldsymbol{C}_{ij})$ (RobustUpdate)
    for $\ell \leftarrow j+1, j+2, \ldots, n/b$
      $\alpha_{i\ell}^{-1}\boldsymbol{C}_{i\ell} \leftarrow \alpha_{i\ell}^{-1}\boldsymbol{C}_{i\ell} - (\alpha_{ij}^{-1}\boldsymbol{C}_{ij})\boldsymbol{B}_{j\ell}$ (RobustUpdate)
Reduce $\alpha \leftarrow \min_{i,j}\{\alpha_{ij}\}$
Scale all tiles consistently $\boldsymbol{C}_{ij} \leftarrow (\alpha/\alpha_{ij})\boldsymbol{C}_{ij}$
return $[\alpha, \boldsymbol{C}]$

$\textsc{RobustUpdate}([\alpha, \boldsymbol{A}], [\beta, \boldsymbol{B}], [\gamma, \boldsymbol{C}])$

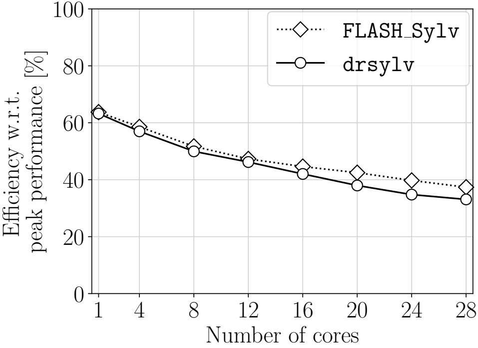
$\eta \leftarrow \min\{\alpha, \beta, \gamma\}$
$\zeta \leftarrow \textsc{ProtectUpdate}((\eta/\gamma)||\boldsymbol{C}||_\infty), (\eta/\alpha)||\boldsymbol{A}||_\infty, (\eta/\beta)||\boldsymbol{B}||_\infty)$
$\delta \leftarrow \eta\zeta$
$\boldsymbol{D} \leftarrow (\delta/\gamma)\boldsymbol{C} - [(\delta/\alpha)\boldsymbol{A}]\,[(\delta/\beta)\boldsymbol{B}]$
return $[\delta, \boldsymbol{D}]$

Used to compute

$$\delta^{-1}\boldsymbol{D} \leftarrow \gamma^{-1}\boldsymbol{C} - \boldsymbol{A}(\beta^{-1}\boldsymbol{B})$$
$$\delta^{-1}\boldsymbol{D} \leftarrow \gamma^{-1}\boldsymbol{C} - (\alpha^{-1}\boldsymbol{A})\boldsymbol{B}$$

Race on $m = n = 10000$

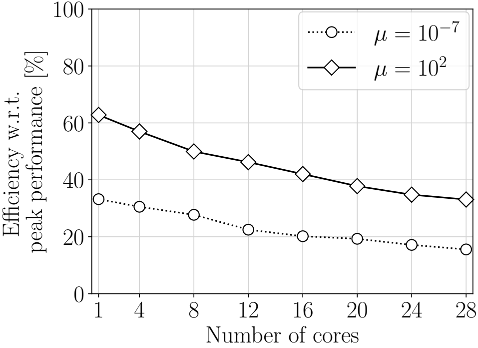
Comparison with non-robust task-parallel `FLASH_Sylv` (libflame)
No numerical scaling required



The overhead of overflow protection is negligible when numerical scaling is not necessary.

Race on $m = n = 10000$

$\mu = 10^2$: No numerical scaling required (best case)
$\mu = 10^{-7}$: Numerical scaling required in every step (worst case)



Numerical scaling is costly, but without it the problem cannot be solved.

Summary

- Already simple systems can have quickly growing solutions
- Dynamic scaling of the solution allows us to avoid overflow
- Tile-local scaling factors close the gap between libflame/`FLASH_Sylv` and LAPACK/`dtrsyl` without sacrificing parallel scalability
- Overhead from overflow protection is negligible when numerical scaling is not necessary
- Works for quasi-triangular $\boldsymbol{A}$, $\boldsymbol{B}$, too

There is no reason to not use a robust solver!

Task graph