A multilevel Schur complement preconditioner with ILU factorization for complex symmetric matrices

Rainer Schlundt

submitted: November 26, 2018

Weierstrass Institute
Mohrenstr. 39
10117 Berlin
Germany
E-Mail: rainer.schlundt@wias-berlin.de

No. 2556
Berlin 2018

2010 Mathematics Subject Classification. 65F08, 65F15, 65N22, 65Y05.

Key words and phrases. Complex symmetric sparse linear system, Schur complement, multilevel preconditioner, incomplete LU factorization, Bunch-Kaufman pivoting, domain decomposition, low rank approximation.
A multilevel Schur complement preconditioner with ILU factorization for complex symmetric matrices

Rainer Schlundt

Abstract

This paper describes a multilevel preconditioning technique for solving complex symmetric sparse linear systems. The coefficient matrix is first decoupled by domain decomposition and then an approximate inverse of the original matrix is computed level by level. This approximate inverse is based on low rank approximations of the local Schur complements. For this, a symmetric singular value decomposition of a complex symmetric matrix is used. The block-diagonal matrices are decomposed by an incomplete \( LDLT^T \) factorization with the Bunch-Kaufman pivoting method.

Using the example of Maxwell’s equations the generality of the approach is demonstrated.

1 Introduction

We consider iterative methods for solving large sparse systems

\[ Ax = b, \]

where \( A \in \mathbb{C}^{n \times n} \), \( A = A^T \), \( A \neq A^H \), \( b \in \mathbb{C}^n \), and \( x \in \mathbb{C}^n \). Krylov subspace methods combined with a preconditioner solve the above system \( Ax = b \). The preconditioner \( M \) is an approximation to \( A \). The solve of the preconditioned system is relatively inexpensive.

The domain decomposition (DD) approach decouples the original matrix \( A \). We do not form the global Schur complement system and do not solve it exactly. Let \( A \) be partitioned in \( 2 \times 2 \) block form as

\[ A = \begin{pmatrix} B & E \\ E^T & C \end{pmatrix}, \]

where \( B \in \mathbb{C}^{m \times m}, C \in \mathbb{C}^{s \times s}, E \in \mathbb{C}^{m \times s}, \) and \( n = m + s \). We will receive the following basic block factorization of \( A \)

\[ A = \begin{pmatrix} B & E \\ E^T & C \end{pmatrix} = \begin{pmatrix} I & 0 \\ E^T B^{-1} & I \end{pmatrix} \begin{pmatrix} B & 0 \\ 0 & S \end{pmatrix} \begin{pmatrix} I & B^{-1}E \\ 0 & I \end{pmatrix}, \]

where \( S \in \mathbb{C}^{s \times s}, S = C - E^T B^{-1} E, \) is the Schur complement. Using

\[ A^{-1} = \begin{pmatrix} I & -B^{-1}E \\ 0 & I \end{pmatrix} \begin{pmatrix} B^{-1} & 0 \\ 0 & S^{-1} \end{pmatrix} \begin{pmatrix} I & 0 \\ -E^T B^{-1} & I \end{pmatrix}, \]

the original system \( Ax = b \) can be easily solved if \( S^{-1} \) is available. The goal is to approximate \( S^{-1} \) such that \( S^{-1} \approx C^{-1} + LRA = S^{-1} \), where \( LRA \) stands for low rank approximation matrix. The preconditioner \( M \) then has the following form

\[ M = \begin{pmatrix} I & 0 \\ E^T B^{-1} & I \end{pmatrix} \begin{pmatrix} B & 0 \\ 0 & S \end{pmatrix} \begin{pmatrix} I & B^{-1}E \\ 0 & I \end{pmatrix}. \]
We can write
\[ S = C - E^T B^{-1} E = C^{1/2} (I - C^{-1/2} E^T B^{-1} E C^{-1/2}) C^{1/2} = C^{1/2} (I - G) C^{1/2} \] (6)
and
\[ S^{-1} = C^{-1/2} (I - G)^{-1} C^{-1/2} = C^{-1} + C^{-1/2} G (I - G)^{-1} C^{-1/2}. \] (7)
The symmetric matrix \( G \in \mathbb{C}^{s \times s} \) has a symmetric singular value decomposition (SSVD)
\[ G = C^{-1/2} E^T B^{-1} E C^{-1/2} = W \Sigma W^T, \] (8)
where \( W \) is a unitary matrix and \( \Sigma = \text{diag}(\sigma_1, \ldots, \sigma_s) \) with nonnegative \( \sigma_i \) (cf. [17]). Then \( LRA \)
is an approximation of \( C^{-1/2} G (I - G)^{-1} C^{-1/2} \). The remaining sections are organized as follows.
Section 2 gives an overview of the domain decomposition framework and the multilevel preconditioning
技术 proposed in [10]. The incomplete \( LDL^T \) factorization with the Bunch-Kaufman pivoting is
discussed in Section 3. Numerical experiments of a model problem are presented in Section 4.
Implementation details for a real symmetric matrix \( A \) are described in [7, 8].

2 Domain decomposition and multilevel preconditioning

An interesting class of domain decomposition methods is the hierarchical interface decomposition
(HID) ordering (cf. [2]). An HID ordering can be obtained from a standard graph partitioning (cf.
METIS [4]). The reordered matrix has the following multilevel recursive form:
\[ A_j = P_j C_{j-1} P_j^T = \begin{pmatrix} B_j & E_j \\ E_j^T & C_j \end{pmatrix} \quad \text{and} \quad C_0 \equiv A \quad \text{for} \quad j = 1, \ldots, \text{lev}. \] (9)

\( P_j \) is a permutation matrix and \( \text{lev} \) the number of levels. Each block \( B_j \) in \( A_j \) has a block-diagonal
structure resulting from this HID ordering. Analogous to (2), let \( A_j \) be partitioned at level \( j \) in block
form as
\[ A_j = \begin{pmatrix} B_j & E_j \\ E_j^T & C_j \end{pmatrix} = \begin{pmatrix} B_{j_1} & \ldots & E_{j_1} \\ \vdots & \ddots & \vdots \\ E_{j_p}^T & \ldots & C_j \end{pmatrix}, \] (10)
where \( B_{j_i} \in \mathbb{C}^{m_{j_i} \times m_{j_i}}, E_{j_i} \in \mathbb{C}^{m_{j_i} \times s_j}, E_j = (E_{j_1}, \ldots, E_{j_p}), B_{j_i} \in \mathbb{C}^{m_{j_i} \times m_{j_i}}, E_{j_i} \in \mathbb{C}^{m_{j_i} \times s_j}, 1 \leq i \leq p, n_j = m_j + s_j, \) and
\( m_j = m_{j_1} + \cdots + m_{j_p} \). Analogous to (3), at each level \( j \), the factorization of \( A_j \) is determined by
\[ A_j = \begin{pmatrix} B_j & E_j \\ E_j^T & C_j \end{pmatrix} = \begin{pmatrix} I & 0 \\ E_j^T B_j^{-1} \end{pmatrix} \begin{pmatrix} B_j & 0 \\ 0 & S_j \end{pmatrix} \begin{pmatrix} I & B_j^{-1} E_j \\ 0 & I \end{pmatrix}, \] (11)
where \( S_j = C_j - E_j^T B_j^{-1} E_j \) is the Schur complement at level \( j \). Thus
\[ A_j^{-1} = \begin{pmatrix} I & -B_j^{-1} E_j \\ 0 & I \end{pmatrix} \begin{pmatrix} B_j^{-1} & 0 \\ 0 & S_j^{-1} \end{pmatrix} \begin{pmatrix} I & 0 \\ -E_j^T B_j^{-1} \end{pmatrix}, \] (12)
is the inverse of \( A_j \). Analogous to (7), \( S_j^{-1} \) can be approximated by \( C_j^{-1} \) plus an approximation of
\( C_j^{-1/2} G_j (I - G_j)^{-1} C_j^{-1/2} \). The preconditioner \( M_j \) then has the following form
\[ M_j = \begin{pmatrix} I & 0 \\ E_j^T B_j^{-1} \end{pmatrix} \begin{pmatrix} B_j & 0 \\ 0 & S_j \end{pmatrix} \begin{pmatrix} I & B_j^{-1} E_j \\ 0 & I \end{pmatrix} \] (13)
and
\[ C_j^{-1} = P_j^T M_j^{-1} P_j + \hat{D}_j. \]  
\[ (14) \]
At each level \( j \), the symmetric matrix \( G_j \in \mathbb{C}^{n \times n} \), has a singular value decomposition (SVD)
\[ G_j = C_j^{-1/2} E_j^T B_j^{-1} E_j C_j^{-1/2} = U_j \Sigma_j V_j^H, \]  
\[ (15) \]
where \( U_j \) and \( V_j \) are unitary matrices and \( \Sigma_j = \text{diag}(\sigma_{j1}, \ldots, \sigma_{jn}) \) the singular values with real nonnegative \( \sigma_{ji} \). For the matrix \( G_j \) there exists a unitary matrix \( W_j \) such that
\[ G_j = C_j^{-1/2} E_j^T B_j^{-1} E_j C_j^{-1/2} = W_j \Sigma_j W_j^T \]  
\[ (16) \]
is an SSVD. An SSVD of a symmetric matrix can be determined from its SVD. Therefore we have to modify the singular vectors corresponding to nonzero singular values (cf. [1]). The matrix \( G_j (I - G_j)^{-1} \) results in
\[ G_j (I - G_j)^{-1} = W_j (I - \Sigma_j W_j^T W_j)^{-1} \Sigma_j W_j^T \]  
\[ (17) \]
and consequently
\[ C_j^{-1/2} G_j (I - G_j)^{-1} C_j^{-1/2} = Z_j (I - \Sigma_j Z_j^T C_j Z_j)^{-1} \Sigma_j Z_j^T, \quad Z_j = C_j^{-1/2} W_j. \]  
\[ (18) \]
Thus, the computation of a low rank approximation to \( S_j^{-1} - C_j^{-1} \) (cf. (7), (18)) can be obtained by the following SSVD problem
\[ C_j^{-1} E_j^T B_j^{-1} E_j C_j^{-1} = Z_j \Sigma_j Z_j^T. \]  
\[ (19) \]
Finally, the preconditioned system
\[ M^{-1} Ax = M^{-1} b \quad \text{with} \quad M^{-1} = C_0^{-1} = P_1^T M_1^{-1} P_1 \]  
\[ (20) \]
is to be solved.

3 Incomplete LDL$^T$ with Bunch-Kaufman pivoting

The matrices \( B_{ji} \), \( 1 \leq j \leq \text{lev}, 1 \leq i \leq p \), (cf. [10]) are factored using a form of incomplete Cholesky factorization, so we have \( B_{ji} \approx L_{ji} D_{ji} L_{ji}^T \). The core of the decomposition is a Crout variant of incomplete \( LU (ILU) \), introduced for symmetric matrices by [5], which itself extends works by [6] and [3]. The Crout-based decomposition is an attractive way for computing an incomplete \( LDL^T \) factorization for symmetric matrices, because it naturally preserves structural symmetry. This is especially true when applying dropping rules for the incomplete factorization. It is natural to store \( L_{ji} \) by columns and to have the lower triangular part of \( B_{ji} \) stored similarly.

Let the matrix \( A (A \leftarrow B_{ji}) \) be the sum of the matrices \( \hat{L} \) and \( \hat{D} \), that is, \( A = \hat{L} + \hat{D} + \hat{L}^T \). \( \hat{L} \) is the strict lower part of \( A \) and \( \hat{D} \) the diagonal. Only the lower triangular part of \( A \), \( (\hat{L} + \hat{D}) \), is in the Compressed Sparse Column (CSC) format stored. This is synonymous with the storage of \( (\hat{D} + \hat{L}^T) \) in CSR (Compressed Sparse Row) format. Algorithm [1] shows the Crout version of incomplete Cholesky factorization for a symmetric matrix \( A \), \( A \approx LDL^T \), using a delayed update strategy for the factors. The \( k \)-th column update procedure is described in Algorithm [2]. In the Bunch-Kaufman pivoting method, to find the next pivot only requires searching up to two columns in the reduced matrix. The two columns must be updated before proceeding with the search in the algorithm. Algorithm [3] describes the Bunch-Kaufman pivoting strategy. In Algorithm [1] the \( s \times s \) pivot is typically \( 1 \times 1 \) or \( 2 \times 2 \). The following dropping rule is used. Only the largest nonzero entries in every column are kept. The pre-specified maximum number of fill-ins per column is a multiple of the average number of nonzero elements per column in the original matrix. A bi-index data structure was used to address implementation difficulties in sparse matrix operations (cf. [3, 5, 6]).
Algorithm 1 Crout version of incomplete $LDL^T$ factorization

**Input:** Symmetric matrix $A$, matrix size $n$

**Output:** Matrices $P$, $L$, and $D$, such that $PA^TP \approx LDL^T$

1: **procedure** ILDLC
2: $k = 1$
3: while $k \leq n$ do
4: Find a $s \times s$ pivot in $A_{k:n,k:n}$, $s \in \{1, 2\}$ \>
   call BKPIVOT
5: Apply dropping rules to $w_{k+s:n,1:s}$ \>
   $w_{1:k+s-1,1:s} = 0$
6: $L_{k:k+s-1,k:k+s-1} = I$
7: for $i = k+s, \ldots, n$ do
8: if \{$L_{i,k} \neq 0 \lor L_{i,k+1} \neq 0$\} then
9: \quad $A_{i,i} = A_{i,i} - L_{i,k:k+s-1}D_{k:k+s-1,k:k+s-1}L_{k:k+s-1}^T$
10: end if
11: end for
12: $k = k + s$
13: end while
14: end procedure

Algorithm 2 $k$-th column update procedure

**Input:** Column vector $w$, partial factors $L$ and $D$, matrix size $n$, current column index $k$

**Output:** Updated column $w$

1: **procedure** UPDATE
2: $i = 1$
3: while $i < k$ do
4: $s \leftarrow$ size of the diagonal block with $D_{i,i}$ as it top left corner
5: if \{$L_{k,i} \neq 0 \lor L_{k,i+1} \neq 0$\} then
6: \quad $w_{k+1:n} = w_{k+1:n} - L_{k+1:n,i:i+s-1}D_{i:i+s-1,i:i+s-1}L_{k:i:i+s-1}^T$ \>
   $w_{1:k} = 0$
7: end if
8: $i = i + s$
9: end while
10: end procedure
Algorithm 3 Bunch-Kaufman pivoting method at step \( k \)

**Input:** Symmetric matrix \( A \) at step \( k \), partial factors \( L \) and \( D \), matrix size \( n \), current column index \( k \)

**Output:** Symmetric updated matrix \( A \), \( s \times s \) pivot, column vectors \( w_{1:s} \)

1: procedure BKPIVOT
2: \( \alpha = (1 + \sqrt{17})/8 \)
3: Load and update \( A_{k+1:n,k} \): \( w_{1:k,1} = 0, w_{k+1:n,1} = A_{k+1:n,k} \) \( \triangleright \) call UPDATE
4: Let \( \lambda = \|w_{k+1:n,1}\|_\infty = \max_{k+1 \leq j \leq n} |w_{j,1}| \) and \( w_{l,1} = \lambda \) \( \triangleright \) \( l \) smallest integer
5: if \( |A_{k,k}| \geq \alpha \lambda \) then
6: Use \( D_{k,k} = A_{k,k} \) as a 1 \( \times \) 1 pivot
7: \( s = 1 \)
8: else
9: Load and update \( A_{k+1:n,l} \): \( w_{1:k,2} = 0, w_{k+1:n,2} = (A_{l,k:l-1}, A_{l+1:n,l})^T \) \( \triangleright \) call UPDATE
10: Let \( \sigma = \max_{k+1 \leq j \leq n} |w_{j,2}| \)
11: if \( |A_{k,k}| \sigma \geq \alpha \lambda^2 \) then
12: Use \( D_{k,k} = A_{k,k} \) as a 1 \( \times \) 1 pivot
13: \( s = 1 \)
14: else if \( |A_{l,l}| \geq \alpha \sigma \) then
15: Use \( D_{k,k} = A_{l,l} \) as a 1 \( \times \) 1 pivot
16: \( w_{k+1:n,1} = w_{k+1:n,2} \)
17: \( s = 1 \) \( \triangleright \) interchange the \( k \)-th and the \( l \)-th rows and columns
18: else
19: Use \( \begin{pmatrix} D_{k,k} & D_{k,k+1} \\ D_{k+1,k} & D_{k+1,k+1} \end{pmatrix} = \begin{pmatrix} A_{k,k} & \lambda \\ \lambda & A_{l,l} \end{pmatrix} \) as a 2 \( \times \) 2 pivot
20: \( s = 2 \) \( \triangleright \) interchange the \( (k+1) \)-th and the \( l \)-th rows and columns
21: end if
22: end if
23: end procedure
4 Numerical experiments

Using the example of Maxwell’s equations we demonstrate the generality of the approach. We obtain in vector notation the following equations in integral form:

\[
\begin{align*}
\oint_{P} \vec{E} \cdot d\vec{l} &= -\frac{\partial}{\partial t} \oint_{A} \vec{B} \cdot d\vec{A} \\
\oint_{P} \vec{B} \cdot d\vec{A} &= \oint_{A} \vec{H} \cdot d\vec{A} + \oint_{A} \vec{J} \cdot d\vec{A} \\
\oint_{S} \vec{B} \cdot d\vec{S} &= 0 \\
\oint_{S} \vec{D} \cdot d\vec{S} &= \int_{V} q \, dV.
\end{align*}
\]  

(21)

The constitutive relations belonging to them are

\[
\vec{D} = \varepsilon \vec{E}, \quad \vec{B} = \mu \vec{H}, \quad \vec{J} = \kappa \vec{E}.
\]  

(22)

Here, \( A \) is a surface with boundary curve \( P \), \( V \) is a volume bounded by a surface \( S \), and \( q \) is the volume charge density. An orthogonal dual mesh is used to discretize the Maxwell’s equations using the Finite Integration Technique (FIT, [13, 14, 9]). The electric and magnetic voltages and fluxes over elementary objects are defined as state variables in the following way:

\[
\begin{align*}
\varepsilon_{i} &= \int_{L_{i}} \vec{E} \cdot d\vec{l} \\
\mu_{j} &= \int_{L_{j}} \vec{H} \cdot d\vec{l} \\
\nu_{i} &= \int_{A_{i}} \vec{D} \cdot \vec{n} \, d\vec{A} \\
\kappa_{j} &= \int_{A_{j}} \vec{B} \cdot \vec{n} \, d\vec{A} \\
\kappa_{i} &= \int_{A_{i}} \vec{J} \cdot \vec{n} \, d\vec{A} \\
\kappa_{k} &= \int_{V_{k}} q \, dV
\end{align*}
\]

where \( \vec{n} \) is the outward-pointing normal of the faces \( A_{j} \) and \( \tilde{A}_{i} \), respectively. If all field quantities vary sinusoidally with time, the coefficient matrices of the corresponding linear systems of equation are complex, symmetric, and indefinite. Using Krylov subspace methods, [20] can be solved iteratively (cf. [11, 12]).

We consider different dimensions of the coefficient matrices of the corresponding systems of linear equations, in fact \( n = 16632, n = 40824, n = 472416, \) and \( n = 4020192 \). At each level \( j, j = 1, 2, \ldots, \) the matrix \( A \) is partitioned into \( p, p \in \{0, 2, 3, 5, 10, 15\} \), non-overlapping subsets \( B_{ji}, i = 1, \ldots, p \). At each \( p, p \in \{2, 3, 5, 10, 15\} \), we compute the \( k, k \in \{5, 10, 15, 20\} \), largest singular values and the corresponding singular vectors to obtain a low rank approximation. For \( p = 0 \) the solution is computed by [11, 12]. The same applies on the one hand for the computation of the solution with the coefficient matrices \( B_{ji} \) and on the other hand we compute an incomplete \( LDL^T \) factorization of \( B_{ji} \), (cf. Section 3). The testing platform consists of Intel Xeon W3520 processors with 2.67 GHz.

The following notation is used throughout the section:

- icf: “T” indicates that the matrices \( B_{ji} \) were formed by an incomplete \( LDL^T \) factorization, on the other hand “F”
- its: number of iterations of preconditioned solver to reduce the initial residual by a factor of \( 10^{-8} \)
From Tables 1-4 we find the number of iterations and the wall clock times for the different dimensions. In Table 5 we find the corresponding dimensions of the matrices \(B_j\) and \(C_j\) for \(j = 1, 2, \ldots, p\) of the dimension \(n = 4020192\). For the other dimensions, i.e., \(n = 16632, n = 40824,\) and \(n = 472416\), the corresponding informations can be found in Tables 4-6 of [10]. The informations from the Tables 1-4 are shown graphically in the Figures 1-4. The red line indicates the values for \(p = 0\).

The following notation is used for the figures:

- (a): number of iterations (its, icf = “F”)
- (b): wall clock time for the iteration phase (s-t, icf = “F”)
- (c): the proportion of wall clock time s-t (coloured) in the total time p-t (icf = “F”)
- (d): number of iterations (its, icf = “T”)
- (e): wall clock time for the iteration phase (s-t, icf = “T”)
- (f): the proportion of wall clock time s-t (coloured) in the total time p-t (icf = “T”)

With the exception of \(n = 4020192\), it can be seen that the lowest iteration numbers in the iteration process have been achieved for \(\text{itc} = \text{“F”}, p = 2,\) and \(k \in \{5, 10, 15, 20\}\). For small dimensions, also \(p = 3\) is useful. This process is very time consuming. In general, the iteration numbers for \(\text{icf} = \text{“T”}\) are greater than those for \(\text{icf} = \text{“F”}\) and smaller than those for \(p = 0\). They are comparable for \(n = 16632\). For high-dimensional problems the computation of the incomplete \(LDLT^T\) factorization of the matrices \(B_j\) is also time consuming. Likewise, the computation with \(B_j\) for greater \(k \in \{5, 10, 15, 20\}\) becomes more time consuming. Experimental results indicate that this preconditioner based on Schur complement approach is robust in the iteration phase.

5 Conclusions

This paper presents a preconditioning method based on a Schur complement approach with low rank approximations for solving complex symmetric sparse linear systems. It tries to approximate the inverse of the Schur complement by exploiting low rank approximations. For this, a hierarchical graph decomposition reorders the matrix into a multilevel block form. On the negative side, building this preconditioner can be time consuming. A solve with the matrix \(B_j\) amounts to \(p\) local and independent solves with the matrices \(B_j, i = 1, \ldots, p\). These can be carried out by a preconditioned Krylov subspace iteration and by an incomplete \(LDLT^T\) factorization with Bunch-Kaufman pivoting, respectively. A big part of the computations to build a preconditioner based on Schur complement approach is attractive for massively parallel machines. This also applies to the application of the preconditioner in the iteration process.
Table 1: The number of iterations and the wall clock times for $n = 16\,632$.

<table>
<thead>
<tr>
<th>number of subsets</th>
<th>$k$: largest singular values</th>
<th>5</th>
<th>10</th>
<th>15</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>p icf</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>its</td>
<td>128</td>
<td>123</td>
<td>124</td>
<td>121</td>
</tr>
<tr>
<td></td>
<td></td>
<td>23.584</td>
<td>0.569</td>
<td>22.653</td>
<td>0.612</td>
</tr>
<tr>
<td></td>
<td></td>
<td>44.441</td>
<td>1.743</td>
<td>37.777</td>
<td>1.619</td>
</tr>
<tr>
<td></td>
<td></td>
<td>its = 157</td>
<td>p-t = 0.238</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>its</td>
<td>142</td>
<td>126</td>
<td>147</td>
<td>126</td>
</tr>
<tr>
<td></td>
<td></td>
<td>22.424</td>
<td>0.616</td>
<td>23.032</td>
<td>0.748</td>
</tr>
<tr>
<td></td>
<td></td>
<td>45.940</td>
<td>1.583</td>
<td>37.425</td>
<td>1.692</td>
</tr>
<tr>
<td>3</td>
<td>its</td>
<td>146</td>
<td>142</td>
<td>143</td>
<td>141</td>
</tr>
<tr>
<td></td>
<td></td>
<td>19.490</td>
<td>0.735</td>
<td>19.209</td>
<td>0.920</td>
</tr>
<tr>
<td></td>
<td></td>
<td>23.290</td>
<td>1.134</td>
<td>27.267</td>
<td>1.539</td>
</tr>
<tr>
<td>5</td>
<td>its</td>
<td>182</td>
<td>177</td>
<td>187</td>
<td>176</td>
</tr>
<tr>
<td></td>
<td></td>
<td>20.314</td>
<td>1.282</td>
<td>21.171</td>
<td>1.776</td>
</tr>
<tr>
<td></td>
<td></td>
<td>29.963</td>
<td>2.215</td>
<td>33.603</td>
<td>2.954</td>
</tr>
<tr>
<td>10</td>
<td>its</td>
<td>211</td>
<td>208</td>
<td>208</td>
<td>204</td>
</tr>
<tr>
<td></td>
<td></td>
<td>21.885</td>
<td>1.701</td>
<td>21.806</td>
<td>2.545</td>
</tr>
<tr>
<td></td>
<td></td>
<td>34.714</td>
<td>3.053</td>
<td>34.976</td>
<td>4.241</td>
</tr>
<tr>
<td>15</td>
<td>its</td>
<td>211</td>
<td>208</td>
<td>208</td>
<td>204</td>
</tr>
<tr>
<td></td>
<td></td>
<td>21.885</td>
<td>1.701</td>
<td>21.806</td>
<td>2.545</td>
</tr>
<tr>
<td></td>
<td></td>
<td>34.714</td>
<td>3.053</td>
<td>34.976</td>
<td>4.241</td>
</tr>
</tbody>
</table>

Table 2: The number of iterations and the wall clock times for $n = 40\,824$.

<table>
<thead>
<tr>
<th>number of subsets</th>
<th>$k$: largest singular values</th>
<th>5</th>
<th>10</th>
<th>15</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>p icf</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>its</td>
<td>172</td>
<td>401</td>
<td>168</td>
<td>410</td>
</tr>
<tr>
<td></td>
<td></td>
<td>200.5</td>
<td>4.892</td>
<td>196.0</td>
<td>5.479</td>
</tr>
<tr>
<td></td>
<td></td>
<td>238.5</td>
<td>9.180</td>
<td>252.8</td>
<td>9.972</td>
</tr>
<tr>
<td></td>
<td></td>
<td>its = 510</td>
<td>p-t = 1.609</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>its</td>
<td>182</td>
<td>389</td>
<td>173</td>
<td>385</td>
</tr>
<tr>
<td></td>
<td></td>
<td>157.6</td>
<td>5.111</td>
<td>151.2</td>
<td>5.777</td>
</tr>
<tr>
<td></td>
<td></td>
<td>193.9</td>
<td>7.960</td>
<td>191.5</td>
<td>8.760</td>
</tr>
<tr>
<td>3</td>
<td>its</td>
<td>294</td>
<td>398</td>
<td>283</td>
<td>380</td>
</tr>
<tr>
<td></td>
<td></td>
<td>144.2</td>
<td>5.661</td>
<td>139.3</td>
<td>6.823</td>
</tr>
<tr>
<td></td>
<td></td>
<td>166.0</td>
<td>7.608</td>
<td>168.0</td>
<td>8.981</td>
</tr>
<tr>
<td>5</td>
<td>its</td>
<td>347</td>
<td>430</td>
<td>341</td>
<td>417</td>
</tr>
<tr>
<td></td>
<td></td>
<td>133.5</td>
<td>6.752</td>
<td>132.5</td>
<td>8.906</td>
</tr>
<tr>
<td></td>
<td></td>
<td>153.9</td>
<td>8.254</td>
<td>165.1</td>
<td>11.085</td>
</tr>
<tr>
<td>10</td>
<td>its</td>
<td>382</td>
<td>465</td>
<td>366</td>
<td>431</td>
</tr>
<tr>
<td></td>
<td></td>
<td>127.6</td>
<td>8.015</td>
<td>124.6</td>
<td>10.498</td>
</tr>
<tr>
<td></td>
<td></td>
<td>151.0</td>
<td>9.505</td>
<td>161.6</td>
<td>13.157</td>
</tr>
<tr>
<td>15</td>
<td>its</td>
<td>382</td>
<td>465</td>
<td>366</td>
<td>431</td>
</tr>
<tr>
<td></td>
<td></td>
<td>127.6</td>
<td>8.015</td>
<td>124.6</td>
<td>10.498</td>
</tr>
<tr>
<td></td>
<td></td>
<td>151.0</td>
<td>9.505</td>
<td>161.6</td>
<td>13.157</td>
</tr>
</tbody>
</table>
Table 3: The number of iterations and the wall clock times for $n = 472\,416$.

<table>
<thead>
<tr>
<th>number of subsets</th>
<th>$k$: largest singular values</th>
</tr>
</thead>
<tbody>
<tr>
<td>p icf</td>
<td>5</td>
</tr>
<tr>
<td>its</td>
<td>344 943</td>
</tr>
<tr>
<td>s-t</td>
<td>6930.6 157.3</td>
</tr>
<tr>
<td>p-t</td>
<td>8203.2 1054.6</td>
</tr>
</tbody>
</table>

Table 4: The number of iterations and the wall clock times for $n = 4\,020\,192$.

<table>
<thead>
<tr>
<th>number of subsets</th>
<th>$k$: largest singular values</th>
</tr>
</thead>
<tbody>
<tr>
<td>p icf</td>
<td>5</td>
</tr>
<tr>
<td>its</td>
<td>14 659 5295</td>
</tr>
<tr>
<td>s-t</td>
<td>15 159 - 15 602 - 16 033</td>
</tr>
<tr>
<td>p-t</td>
<td>79 548 - 80 733 - 80 785</td>
</tr>
</tbody>
</table>

DOI 10.20347/WIAS.PREPRINT.2556 Berlin 2018
Table 5: The dimensions of the matrices $B_{ji}$ and $C_j$ for $n = 4020192$.

<table>
<thead>
<tr>
<th>$p$</th>
<th>level $j$</th>
<th>dim($B_{ji}$)</th>
<th>dim($C_j$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1</td>
<td>1995946</td>
<td>28355</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>13976</td>
<td>401</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>193</td>
<td>14</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1326396</td>
<td>51348</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>17021</td>
<td>534</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>172</td>
<td>13</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>783419</td>
<td>971301</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>19355</td>
<td>691</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>138</td>
<td>19</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>390394</td>
<td>186772</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>18041</td>
<td>5426</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>537</td>
<td>56</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>251999</td>
<td>230150</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>14788</td>
<td>7249</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>482</td>
<td>74</td>
</tr>
</tbody>
</table>

Figure 1: The number of iterations and wall clock times for $n = 16632$
Figure 2: The number of iterations and wall clock times for $n = 40\,824$

Figure 3: The number of iterations and wall clock times for $n = 472\,416$
Figure 4: The number of iterations and wall clock times for $n = 4\ 020\ 192$

References


