Algebraic Preconditioning Approaches and Their Applications

Matthias Bollhöfer
TU Braunschweig, Institute for Computational Mathematics
D-38106 Braunschweig, Germany
m.bollhoefer@tu-bs.de

February 27, 2015

Abstract

We will review approaches to numerically treat large-scale systems of equations including preconditioning, in particular those methods which are suitable for solving linear systems in parallel. We will also demonstrate how completion techniques can serve as a useful tool to prevent ill-conditioned systems. Beside parallel aspects for preconditioning, multilevel factorization methods will be investigated and finally we will demonstrate how these methods can be combined for approximate matrix inversion methods.

1 Introduction

Solving linear systems of the form $Ax = b$, where $A \in \mathbb{R}^{n\times n}$ is nonsingular, $x, b \in \mathbb{R}^n$ efficiently is an ubiquitous problem in many scientific applications such as solving partial differential equations, inverting matrices or parts of matrices or computing eigenstates in computational physics and many other application areas. For specific application problems, methods that are tailored to the underlying problem often serve best as problem-dependent solver, e.g. multigrid methods [34, 67, 71] are among the best methods for solving large classes of partial differential equations efficiently. However, when the underlying application problems do not posses enough problem-dependent structure information to allow for specific solution methods, more general methods are needed. Often enough, sparse direct solution methods (e.g. [22, 23, 62]) are very efficient and even if their efficiency with respect to computation time and memory is not quite satisfactory, their robustness is a strong argument to prefer these kind of methods, in particular, because only a small number of parameters needs to be adapted, if any. In contrast to that, preconditioned Krylov subspace

solvers [32, 59, 31] are a frequently used alternative whenever an efficient preconditioner is available to solve the system in a reasonable amount of time. Nowadays as multicore and manycore architectures become standard even for desktop computers, parallel approaches to raise efficiency have gained attraction and are not anymore restricted to supercomputers. Many parallelization strategies are based on divide & conquer principles which decompose the whole problem into a sequence of smaller problems to be treated independently plus an additional coupling system to reveal the original problem [11, 15, 49, 33, 3, 62]. Among many parallelization approaches to solve linear systems, general black-box approaches are based on splitting the system or, partitioning the system appropriately into one part that is easily treated in parallel and a remaining part. Due to the rapidly increasing number of cores available for parallel solution techniques, direct solvers are often replaced by hybrid solvers in order to solve some part of the system directly while the additional coupling system is solved iteratively (see e.g. [47, 30]). With respect to their core part, these methods are based on a similar parallelization principle. To describe the breadth of parallel preconditioning approaches for efficiently solving linear systems would be too much to be covered by this article. Here we will focus on selected aspects which can also be used for efficient multilevel incomplete factorization techniques and for inverting parts of a matrix.

We will start in Section 2 reviewing splitting and partitioning methods for solving block-tridiagonal systems in parallel, in particular parallel direct and hybrid methods are often based on this kind of approach. After that we will display in Section 3, how similar methods can be set up even when the system is not block-tridiagonal. Section 4 will state how splitting-type methods can be improved to avoid ill-conditioned systems. Next we will demonstrate in Section 5 how algebraic multilevel preconditioners can be easily analyzed and improved and finally Section 6 demonstrates how multilevel methods on the one hand and parallel partitioning methods on the other hand can be employed for approximate matrix inversion.

2 Hybrid solution methods

With ever increasing size, large-scale systems are getting harder to be solved by direct methods and often enough, out-of-core techniques are required in order to solve systems, even in a parallel environment, since the memory consumption may exceed the available main memory. As a compromise between direct methods and preconditioned Krylov subspace methods, hybrid solvers that mix both ideas can be used. We briefly describe the two most common approaches that allow for efficient parallel treatment as well as for hybrid solution methods. Suppose that

$$A = C - EF^T, \tag{1}$$

where $C \in \mathbb{R}^{n\times n}$ is nonsingular and $E, F^T \in \mathbb{R}^{n\times q}$ are of lower rank $q \ll n$. The Sherman-Morrison-Woodbury formula

$$A^{-1} = C^{-1} + C^{-1}E(I - F^TC^{-1}E)^{-1}F^TC^{-1} \tag{2}$$

yields that solving $Ax = b$ is equivalent to

$$\text{solve } Cy = b, \text{ set } r := F^T y, \text{ solve } Rz = r, \text{ set } c = b + Ez, \text{ solve } Cx = c.$$

Here one has to solve two systems $Cy = b$, $Cx = c$ with C directly and a further small system $Rz = r$ with

$$R = I - F^T C^{-1} E \in \mathbb{R}^{q \times q}. \tag{3}$$

One can easily verify that R is nonsingular as well. The bottleneck of this splitting approach consists of computing the small system R explicitly which is most time-consuming. Usually having a small rank q, solving $CU = E$ can be performed efficiently using direct methods. The matrix U is sometimes [11] also called "spike matrix", since it refers to the non-trivial block columns of $C^{-1}A$. If it pays off, one could avoid solving the system $Cx = c$ by using the relation $x = Uz + y$ instead. However, when the rank is increasing, significantly more time is consumed. Thus, alternatively to solving $Rz = r$ directly, iterative solution methods that only require matrix-vector products are a favorable alternative and this finally yields a hybrid solution method [11, 49, 47]. A natural way of obtaining a splitting (1) for large-scale sparse matrices consists of partitioning the matrix A into two diagonal blocks plus a few nonzero off-diagonal entries outside the block-diagonal pattern which are then obviously of lower rank, i.e.,

$$A = \begin{pmatrix} C_1 & 0 \\ 0 & C_2 \end{pmatrix} - \begin{pmatrix} 0 & E_1 F_2^T \\ E_2 F_1^T & 0 \end{pmatrix} = \begin{pmatrix} C_1 & 0 \\ 0 & C_2 \end{pmatrix} - \begin{pmatrix} E_1 & 0 \\ 0 & E_2 \end{pmatrix} \begin{pmatrix} 0 & F_1 \\ F_2 & 0 \end{pmatrix}^T \equiv C - EF^T. \tag{4}$$

This procedure can be recursively applied to C_1 and C_2 to obtain a nested sequence of splittings and solving the systems via the Sherman–Morrison–Woodbury formula (2) can be performed recursively as well [65]. Although being elegant, splitting (4) has the drawback that the recursive application of splittings may also lead to higher complexity [48, 49]. More efficiently, an immediate parallelization approach with p processors prefers to substitute (4) by

$$A = \begin{pmatrix} C_1 & & & 0 \\ & C_2 & & \\ & & \ddots & \\ 0 & & & C_p \end{pmatrix} - EF^T \tag{5}$$

for suitably chosen EF^T. For block-tridiagonal systems having $m \geqslant p$ or significantly more diagonal blocks, EF^T is easily constructed. Suppose for simplicity that $m = l \cdot p$ for some $l \in \{1, 2, 3, \ldots, \}$. Then we have

$$A = \begin{pmatrix} A_{11} & A_{12} & & & & 0 \\ A_{21} & A_{22} & A_{23} & & & \\ & \ddots & \ddots & \ddots & & \\ & & A_{m-1,m-2} & A_{m-1,m-1} & A_{m-1,m} & \\ 0 & & & A_{m,m-1} & A_{mm} & \end{pmatrix}, \; C_i = (A_{rs})_{r,s=(i-1)l+1,\ldots,il} \tag{6}$$

and

$$EF^T = \begin{pmatrix} 0 & E_{12}F_{12}^T & & & 0 \\ E_{21}F_{21}^T & 0 & E_{23}F_{23}^T & & \\ & \ddots & \ddots & \ddots & \\ & & E_{p-1,p-2}F_{p-1,p-2}^T & 0 & E_{p-1,p}F_{p-1,p}^T \\ 0 & & & E_{p,p-1}F_{p,p-1}^T & 0 \end{pmatrix},$$

where

$$E_{i,i+1}F_{i,i+1}^T = \begin{pmatrix} 0 & 0 \\ -A_{il,il+1} & 0 \end{pmatrix},\ E_{i+1,i}F_{i+1,i}^T = \begin{pmatrix} 0 & -A_{il+1,il} \\ 0 & 0 \end{pmatrix}$$

and one could even employ a low rank factorization of $A_{il,il+1}$ and $A_{il+1,il}$ to decrease the rank further. We can take advantage of instantaneously splitting the initial system into p parts since we only obtain a single coupling system R, which is usually small but hard to solve in parallel. Besides, computing R now only requires solving $C_iU_i = (E_{i,i-1},\ E_{i,i+1})$, $i = 1, \ldots, p-1$ simultaneously without further recursion. Here $E_{0,1}$ and $E_{p,p+1}$ are void. Because of its ease, this variant may be preferred to the recursive approach.

Another approach for solving systems $Ax = b$ in parallel consists of partitioning the initial system A into subsystems rather than splitting the matrix A. This approach is favorable in particular in cases where the diagonal blocks of A can be assumed to be safely nonsingular (i.e., the case of positive definite matrices or diagonal dominant matrices). In this case we partition A as

$$A = \left(\begin{array}{ccc|c} C_1 & & 0 & E_{1,p+1} \\ & \ddots & & \vdots \\ 0 & & C_p & E_{p,p+1} \\ \hline F_{1,p+1}^T & \cdots & F_{p,p+1}^T & C_{p+1} \end{array}\right) \equiv \left(\begin{array}{c|c} C & E \\ \hline F^T & C_{p+1} \end{array}\right) \tag{7}$$

and solving $Ax = b$ is easily obtained from the block LU decomposition of the system. I.e., partition

$$x^T = \left(\begin{array}{ccc|c} x_1^T & \cdots & x_p^T & x_{p+1}^T \end{array}\right) \equiv \left(\begin{array}{c|c} \hat{x}^T & x_{p+1}^T \end{array}\right),$$
$$b^T = \left(\begin{array}{ccc|c} b_1^T & \cdots & b_p^T & b_{p+1}^T \end{array}\right) \equiv \left(\begin{array}{c|c} \hat{b}^T & b_{p+1}^T \end{array}\right).$$

Then x is obtained as follows.

$$\text{solve } Cy = \hat{b},\ \text{set } r := b_{p+1} - F^T y,\ \text{solve } Sx_{p+1} = r,\ \text{set } c = b - Ex_{p+1},\ \text{solve } C\hat{x} = c.$$

Here we set $S := C_{p+1} - F^T C^{-1} E$ as the Schur complement. Similar to the case of splitting A as in (5) the major amount of work here is spent in computing S, i.e., computing $C_iU_i = E_{i,p+1}$, $i = 1, \ldots, p$. A natural alternative would also be in this case to solve $Sx_{p+1} = r$ using iterative solution methods which again leads to a hybrid solution method [49, 30, 1]. We like to point out that within the context of solving partial differential equations, these kind of methods are usually called domain decomposition methods, see e.g. [63, 74], which will definitely be beyond the scope of this paper. Instead we will focus on several algebraic aspects.

Example 1 *We demonstrate the difference of the splitting approach (5) and the partitioning approach (7) as direct and hybrid solvers when the block diagonal system is factored using LU decomposition. The alternatives are either generating and solving the coupling systems R and S directly or to avoid explicit computation and use an iterative solver instead. For simplicity we choose the problem $-\Delta u = f$ in $\Omega = [0,1]^2$, with Dirichlet boundary conditions and 5-point-star discretization. We display a simplified parallel model, where we measure only the maximum amount of computation time over all blocks p whenever a system with C_i is treated. In Figure 1 we compare the direct method versus the hybrid method for (5) and (7) based on the initial block-tridiagonal structure of the underlying system with natural ordering. We use MATLAB for these experiments. As*

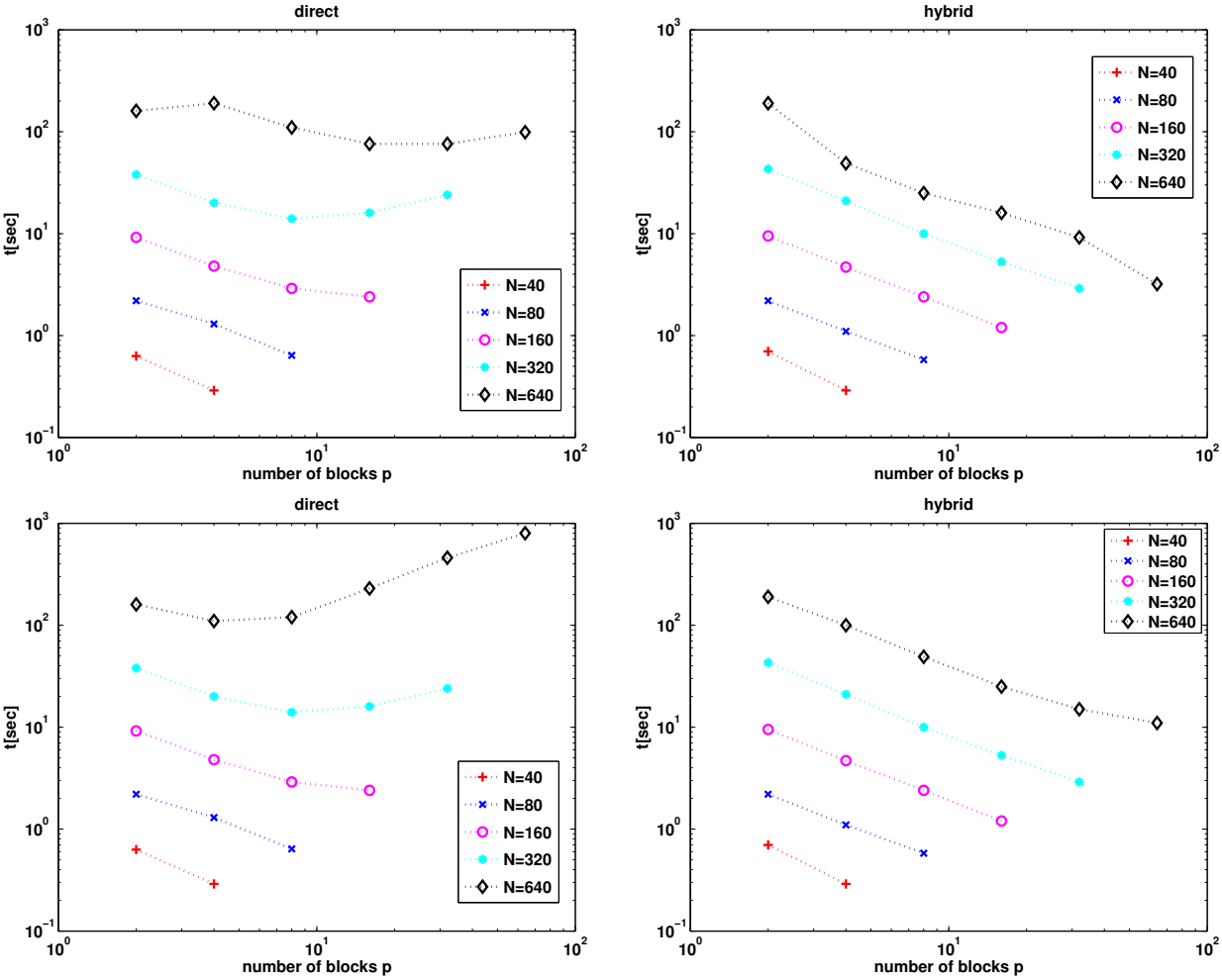


Figure 1: direct methods (left), hybrid methods (right), splitting approaches (top), partitioning approaches (bottom)

we can see from Figure 1, the direct approach is only feasible for small p, since otherwise R and S become too big as confirmed by theoretical estimates in [49]. Moreover, the

computation of the "spike-matrix" U requires solving $2N$ systems with each diagonal block C_i. We can also see that there is no great difference between the splitting approach and the partitioning approach, although in the splitting approach the system is roughly twice as big and nonsymmetric which is the reason for using Bi-CGSTAB [69] as iterative solver. For the partitioning approach CG [36] can be used. Both iterative solvers use a relative residual of 10^{-8} for termination.We also remark at this point that the number of iteration steps significantly increases as the number of blocks p increases (as expected by the domain decomposition theory).

Both approaches based on splittings as in (5) or based on partitionings (7) are relatively similar with respect to parallelization and computational amount of work. The splitting-based approach allows to modify the blocks if necessary, the partitioning-based approach is simpler since it does not rely on especially constructed splittings which is advantageous when the diagonal blocks are safely nonsingular. In Section 3 we will compare both approaches and further generalize them in particular for systems that are not necessarily block-tridiagonal.

3 Reordering and partitioning the system

We will now generalize how to split A as in (1) or to partition A as in (7). First of all we discuss the situation when the (block-)diagonal part of A is far away from having large entries, e.g. in the sense of some diagonal dominance measure [60] such as

$$r_i = \frac{|a_{ii}|}{\sum_{j=1}^{n} |a_{ij}|} \in [0,1],\ i = 1, \ldots, n. \tag{8}$$

Note that a value of r_i larger than $\frac{1}{2}$ refers to a diagonal dominant row. The use of maximum weight matchings [24, 8, 25] is often very helpful to improve the diagonal dominance and to hopefully obtain diagonal blocks that are better conditioned. Maximum weight matchings replace A by

$$A^{(1)} = D_l A D_r \Sigma \tag{9}$$

where $D_l, D_r \in \mathbb{R}^{n\times n}$ are nonsingular, nonnegative diagonal matrices and $\Sigma \in \mathbb{R}^{n\times n}$ is a permutation matrix such that

$$|a_{ij}^{(1)}| \leqslant 1,\ |a_{ii}^{(1)}| = 1, \quad \text{for all } i,j = 1, \ldots, n.$$

Algorithms for computing maximum weight matchings for sparse matrices [24, 25] are experimentally often very fast of complexity $\mathcal{O}(n + nz)$, where nz refers to the number of nonzero elements of A. Note that theoretical bounds are much worse and also that maximum weight matchings are known to be strongly sequential. We illustrate the effect of maximum weight matchings in the following example. For details we refer to [24].

Example 2 *We consider the sample matrix "west0479" (available from the University of Florida collection) of size* $n = 479$ *and number of nonzeros* $nz = 1887$*. In Figure 2 we illustrate the effect of maximum weight matching for this particular matrix. The diagonal*

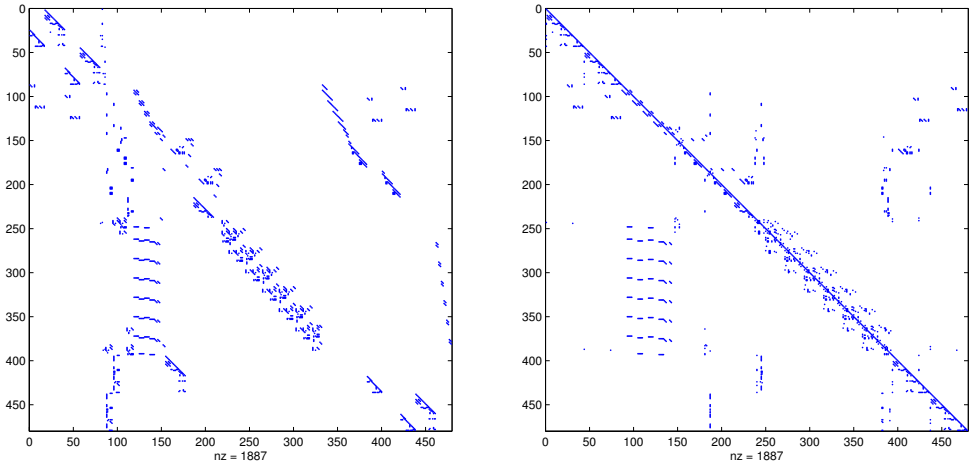


Figure 2: Sample matrix before reordering and rescaling (left) and afterwards (right)

dominance measure r_i *from (8) changes on the average from* $\frac{1}{n}\sum_i r_i^{(old)} \approx 5.7 \cdot 10^{-3}$ *initially to* $\frac{1}{n}\sum_i r_i^{(new)} \approx 0.49$ *after maximum weight matching is applied.*

Even if the system is well-suited with respect to its diagonal blocks, partitioning the matrix into p blocks remains to be done prior to solving the system in a hybrid fashion or to invert parts of the system. To do so, multilevel nested dissection [38, 39] can be used. Formally $A^{(1)}$ is replaced by

$$A^{(2)} = \Pi^T A^{(1)} \Pi$$

for some permutation matrix $\Pi \in \mathbb{R}^{n \times n}$. When targeting a splitting of A such as in (5), nested dissection by edges is the natural partitioning of the system whereas reordering the system matrix A as in (7) requires nested dissection by nodes. We illustrate the difference between both permutation strategies using the following simple undirected graph of a matrix in Example 3. Note that $G(A)$ is called (undirected) graph of A, if it consists of nodes $\mathcal{V} = \{1, \ldots, n\}$ and edges $\mathcal{E} = \{\{i, j\} : \ a_{ij} \neq 0 \text{ or } a_{ji} \neq 0, \forall i \neq j\}$.

Example 3 *We consider an example that frequently applies in solving partial differential equations for a model problem. The graph we use is simply a grid (see Figure 3).*

To reorder the system with respect to the nested dissection approach there exist fast reordering tools, e.g., the MeTis software package [37].

Up to now rescaling and reordering the system matrix can be considered as relatively cheap compared to solving $Ax = b$ or inverting parts of A [24, 8, 25].

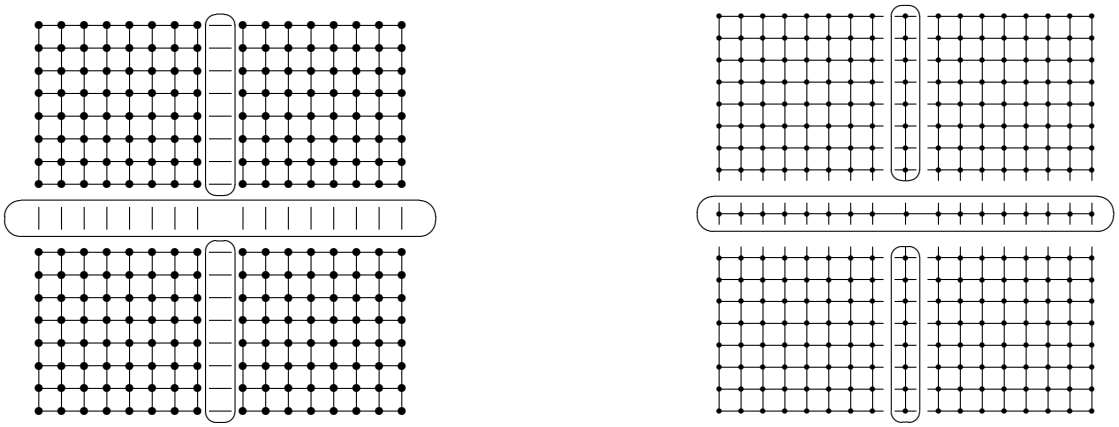


Figure 3: Nested dissection by edges (left) and nested dissection by nodes (right)

3.1 Reordering the system for a splitting-type approach

Now we describe how the preprocessing step can in particular advance splitting or partitioning the system compared with only using a block-tridiagonal structure as in (6). Here we may assume that the underlying matrix is not just block-tridiagonal but sparse. We will start with partitioning the graph with respect to the edges.

Definition 1 *Suppose that $A \in \mathbb{R}^{n\times n}$, $\mathcal{V} = \{1,\dots,n\}$. Let $\mathcal{C}_1 \dot\cup \cdots \dot\cup \mathcal{C}_p = \mathcal{V}$ be a disjoint union of $\mathcal{V}$, partitioning $\mathcal{V}$ into p disjoint subsets. We define $G_M(A) := (\mathcal{V}_M, \mathcal{E}_M)$, where $\mathcal{V}_M = \{1,\dots,p\}$,*

$$\mathcal{E}_M = \{\{r,s\} \subset \mathcal{V}_M \times \mathcal{V}_M : r \neq s, \quad \text{there exist } i \in \mathcal{C}_r, j \in \mathcal{C}_s, \text{ such that } a_{ij} \neq 0\}.$$

We call $G_M(A)$ block or modified graph of A with respect to $\mathcal{C}_1,\dots,\mathcal{C}_p$.

$G_M(A)$ can be regarded as block graph of A after reordering A such that the entries of $\mathcal{C}_1,\dots,\mathcal{C}_p$ are taken in order of appearance and using the associated block matrix shape, i.e., given a suitable permutation matrix $\Pi \in \mathbb{R}^{n\times n}$ we obtain

$$\Pi^T A \Pi = \begin{pmatrix} A_{11} & \cdots & A_{1p} \\ \vdots & & \vdots \\ A_{p1} & \cdots & A_{pp} \end{pmatrix}$$

and many blocks A_{ij} are expected to be zero or of low rank.

Let $e_1,\dots,e_n$ be the standard unit vector basis of $\mathbb{R}^n$. We denote by I_r the matrix of column unit vectors from $\mathcal{C}_r$, i.e.,

$$I_r = (e_j)_{j\in\mathcal{C}_r}, r = 1,\dots,p.$$

Then after reordering A with respect to $\mathcal{C}_1, \ldots, \mathcal{C}_p$ we obtain

$$P^T A P = C - EF^T$$

where

$$C = \begin{pmatrix} A_{11} & & 0 \\ & \ddots & \\ 0 & & A_{pp} \end{pmatrix}, \; EF^T = \sum_{\{r,s\} \in \mathcal{E}_M} (I_r, I_s) \begin{pmatrix} 0 & -A_{rs} \\ -A_{sr} & 0 \end{pmatrix} (I_r, I_s)^T .$$

If we compute some low rank factorization $-A_{rs} = E_{rs} F_{rs}^T$, $-A_{sr} = E_{sr} F_{sr}^T$, then we obtain E and F in a similar way compared with the block tridiagonal case. Suppose that $m = \#\mathcal{E}_M$ and the edges $\{r, s\}$ of $\mathcal{E}_M$ are taken in a suitable order $\{r_1, s_1\}, \ldots, \{r_m, s_m\}$. Then we define E, F via

$$E = (E_1, \ldots, E_m), \; F = (F_1, \ldots, F_m), \tag{10}$$

where

$$E_i = (I_{r_i}, I_{s_i}) \cdot \begin{pmatrix} E_{r_i,s_i} & 0 \\ 0 & E_{s_i,r_i} \end{pmatrix}, \quad F_i = (I_{r_i}, I_{s_i}) \cdot \begin{pmatrix} 0 & F_{r_i,s_i} \\ F_{s_i,r_i} & 0 \end{pmatrix}. \tag{11}$$

We note that if A_{r_i,s_i} and A_{s_i,r_i} have rank q_{r_i,s_i}, q_{s_i,r_i}, then the total rank of E, F is

$$q = \sum_{\{r,s\} \in \mathcal{E}_M} (q_{rs} + q_{sr}) . \tag{12}$$

For general sparse matrices this might lead to a significantly smaller q compared with the case where A is reordered into a block-tridiagonal shape as in Section 2. We will illustrate this effect in the following example.

Example 4 *Consider a matrix A such that its graph is a grid with $M \times M$ grid points, i.e., $n = M^2$. Suppose further that the number of processors p can be written $p = P^2$ and that M is a multiple of P.*

For $p = 4$ we illustrate in Figure 4 two different canonical ways of partitioning the underlying graph. A graph like in Figure 4 may serve as a toy problem for some class of partial differential equations. In the simplest case for the elliptic boundary value problem $-\Delta u = f$ in $[0, 1]^2$ with Dirichlet boundary conditions and 5-point-star difference stencil a graph similar to Figure 4 is obtained. The edges would refer to numerical values -1, the cross points would refer to diagonal entries with value 4. In this case the left partitioning

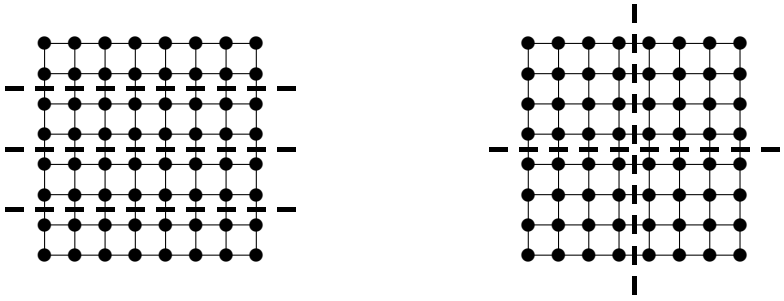


Figure 4: Partitioning the grid into 4 sub-grids horizontally (left) and in checker board fashion (right)

of the domain in Figure 4 would lead to

$$E = \left(\begin{array}{cc|cc|cc} 0 & & & & & \\ I & 0 & & & & \\ \hline 0 & I & & & & \\ & 0 & 0 & & & \\ & & I & 0 & & \\ \hline & & 0 & I & & \\ & & & 0 & 0 & \\ & & & & I & 0 \\ \hline & & & & 0 & I \\ & & & & & 0 \end{array}\right), \; F = \left(\begin{array}{cc|cc|cc} 0 & & & & & \\ 0 & -I & & & & \\ \hline -I & 0 & & & & \\ & 0 & 0 & & & \\ & & 0 & -I & & \\ \hline & & -I & 0 & & \\ & & & 0 & 0 & \\ & & & & 0 & -I \\ \hline & & & & -I & 0 \\ & & & & & 0 \end{array}\right).$$

Each of the identity matrices has size M. The generalization to p sub-blocks is straightforward and would lead to E, F of size $n \times (2(p-1)M)$.

In contrast to this, the checker board partitioning in Figure 4 would lead to E and F which look almost as follows

$$E = \left(\begin{array}{ccc|ccc|ccc} & & & & & & 0 & I & \\ 0 & & & & & & & 0 & \\ I & 0 & & & & & & & \\ \hline 0 & I & & & & & & & \\ & 0 & 0 & & & & & & \\ & & I & 0 & & & & & \\ \hline & & 0 & I & & & & & \\ & & & 0 & 0 & & & & \\ & & & & I & 0 & & & \\ \hline & & & & 0 & I & & & \\ & & & & & 0 & 0 & & \\ & & & & & & I & 0 & \end{array}\right), \; F = \left(\begin{array}{ccc|ccc|ccc} & & & & & & & -I & 0 \\ 0 & & & & & & & & 0 \\ 0 & -I & & & & & & & \\ \hline -I & 0 & & & & & & & \\ & 0 & 0 & & & & & & \\ & & 0 & -I & & & & & \\ \hline & & -I & 0 & & & & & \\ & & & 0 & 0 & & & & \\ & & & & 0 & -I & & & \\ \hline & & & & -I & 0 & & & \\ & & & & & 0 & 0 & & \\ & & & & & & 0 & -I & \end{array}\right).$$

Here, the identity matrices are only of size $M/2$. Strictly speaking, the identity matrices overlap at the center of the grid. We skip this detail for ease of description. For the checker

board partitioning the generalization to $p = P^2$ blocks would lead to a rank proportional to $\sqrt{p}M$ which is significantly less compared with the first case as it grows slower with respect to p.

3.2 Reordering the system for a partitioning-type approach

In contrast to splitting the initial system we now partition it, which means that rather than using nested dissection by edges, we now require nested dissection by nodes as illustrated in Example 3. In this case partitioning the system with respect to the underlying graph can also be advantageous compared to the strategy where A is simply permuted to block-tridiagonal form. We will illustrate this in Example 5.

Example 5 *We consider again a graph of a matrix A that can be represented as a grid in two spatial dimensions. Suppose that the number of processors p can be written $p = P^2$. We assume that the number n of grid points can be written as $n = (M+P-1)^2$ and that M is a multiple of P. For $p = 4$ we illustrate in Figure 5 two obvious ways of partitioning the graph. If we again consider the 5-point-star difference stencil for discretizing the problem*

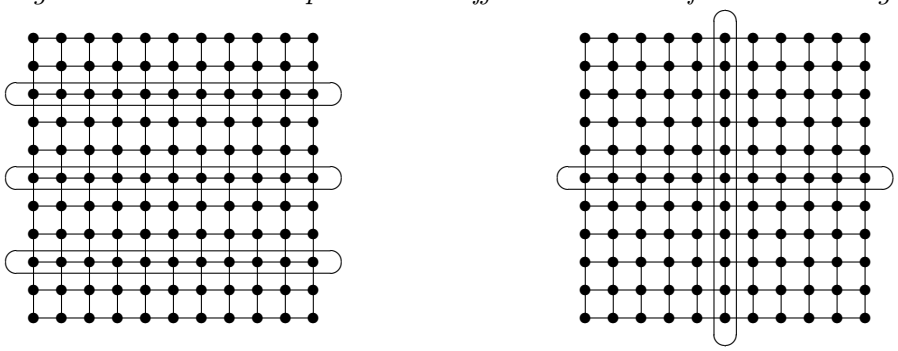


Figure 5: Partitioning the grid into 4 sub-grids horizontally (left) and in checker board fashion (right)

$-\Delta u = f$ in $[0,1]^2$ we still end up with a matrix partitioning

$$A = \left(\begin{array}{c|c} C & E \\ \hline F^T & C_{p+1} \end{array}\right).$$

For the horizontal partitioning approach in general the identity matrices have size $M+p-1$. The size of the Schur-complement $S = C_{p+1} - F^T C^{-1} E$ is identical to the number of nodes we removed, i.e., its size is $(p-1)(M+p-1)$.

In the checker board partitioning case the Schur-complement will have size $2(P-1)(M+P-1)-(P-1)^2$ which is roughly of order $2\sqrt{p}M$ for $p \ll M$. Therefore the checker board partitioning leads to a significantly smaller Schur-complement with respect to p compared with the horizontal approach.

3.3 Splitting-type approach versus partitioning-type approach

After we have illustrated how to preprocess a system $A \to A^{(2)}$ such that the system is either suitable for a splitting-type approach (5) or a partitioning-type approach (7), we will now highlight the common properties, the major differences and which approach should be preferred depending on the situation.

First of all, with respect to parallelization, one has to distinguish whether reordering the system is a suitable option. This is important since depending on the application, the original system matrix A may not be available in total, but it could be distributed over different machines. This is in particular the case for distributed memory machines, where the problem is already generated in parallel. In this case partitioning the system by permutation refers to re-distributing the system in order to obtain a better load balance. This in turn can become quite expensive. When using finite element application for partial differential equations, domain decomposition methods partition the physical domain and the nodes on the interfaces between the domains share the neighbouring subdomains. Algebraically this refers to the partitioning-type approach (7). Otherwise, if there is no natural background why a specific node should share two or more different parts of the system, a more natural distribution in practical applications would be the splitting-type approach (5). For shared memory systems like modern multicore or upcoming manycore architectures we would usually have the whole system matrix A available and we are free to decide which approach should be our method of choice.

A major difference between the splitting-type method (5) and the partitioning-type approach is the size of the systems R and S in similar circumstances like Examples 4 and 5, where the size of R is approximately twice as big as that of S for the block tridiagonal case and for the checker board case the difference is even larger. This is because in the partitioning–type approach the size of S is exactly the number of nodes to be taken out by nested dissection (by nodes), while in the splitting case the size of R is bounded by twice the number of edges (or the number of off-diagonal entries) taken out from graph using nested dissection by edges. The number of edges is usually bigger than the number of nodes and one even obtains a factor 2. On the other hand the rank $q_{rs} + q_{sr}$ of the matrices A_{rs} and A_{sr} that are taken out is the local contribution to the size of R and certainly the rank could be also less than the number of edges. However, there is one improvement that can be obtained for free in the splitting case, which is referred to as minimum rank decoupling [49, 66]. Suppose for simplicity that $q_{rs} = q_{sr}$. If these numbers differ, we could enlarge the factorization $E_{rs}F_{rs}^T$ or $E_{sr}F_{sr}^T$ of smaller size by zeros. Alternatively to (5) we could use the splitting

$$A = \begin{pmatrix} C_1(X) & & & 0 \\ & C_2(X) & & \\ & & \ddots & \\ 0 & & & C_p(X) \end{pmatrix} - E(X)X^{-1}F(X)^T, \tag{13}$$

where we replace locally for any $r < s$

$$\begin{pmatrix} 0 & E_{rs}F_{rs}^T \\ E_{sr}F_{sr}^T & 0 \end{pmatrix}$$

by

$$\begin{pmatrix} E_{rs}X_{rs}F_{sr}^T & E_{rs}F_{rs}^T \\ E_{sr}F_{sr}^T & E_{sr}X_{rs}^{-1}F_{rs}^T \end{pmatrix}$$

for some nonsingular $X_{rs} \in \mathbb{R}^{q_{rs}\times q_{rs}}$ and modify the diagonal blocks $C_1, \ldots, C_p$ appropriately to compensate the changes in the block diagonal position. The advantage of this modification consists of reducing the local rank by a factor 2 since

$$\begin{pmatrix} E_{rs}X_{rs}F_{sr}^T & E_{rs}F_{rs}^T \\ E_{sr}F_{sr}^T & E_{sr}X_{rs}^{-1}F_{rs}^T \end{pmatrix} = \begin{pmatrix} E_{rs}X_{rs} \\ E_{sr} \end{pmatrix} X_{rs}^{-1} \begin{pmatrix} X_{rs}F_{sr}^T & F_{rs}^T \end{pmatrix}. \tag{14}$$

In the simplest case we could choose $X_{rs} = I$. The associated diagonal matrices C_r are changed to

$$C_r(X) := C_r + \sum_{\substack{s:s>r \\ \{r,s\}\in\mathcal{E}_M}} E_{rs}X_{rs}F_{rs}^T + \sum_{\substack{s:s<r \\ \{r,s\}\in\mathcal{E}_M}} E_{rs}X_{rs}^{-1}F_{rs}^T$$

adding only low-rank contributions to C_r. For sparse matrices these modifications only change entries of C_r that are connected to neighbouring blocks. Thus, if $p \ll n$, only a lower-rank part of small size is changed in C_r. For partial differential equations one could read this modification as imposing some kind of inner boundary condition and a natural question will be how to suitably choose

$$X = \operatorname{diag}\left(X_{rs}\right)_{\{r,s\}\in\mathcal{E}_M}. \tag{15}$$

This will be subject of the next section.

To end this section we will demonstrate the benefits of minimum rank decoupling ($X = I$) and using graph partitioning rather than working with a block-tridiagonal shape.

Example 6 *We continue with the problem $-\Delta u = f$ on the unit square in two spatial dimensions and N grid points in each spatial dimension. Here we obtain that $F = E$ and we also have that R is symmetric positive definite. This allows to fully exploit symmetry not only for each C_i, but also R using the Cholesky decomposition, resp. the conjugate gradient method. We use the same settings as in Example 1, except that we perform the numerical experiments for the splitting-type approach (5) only.*

In contrast to Example 1, the size of the "spike-matrix" U now only requires solving $4 \cdot N/p$ systems in parallel rather than $2N$ systems. With increasing size of processors this reduces the overhead for computing the "spike-matrix" significantly. Moreover, as illustrated in Example 5, the size of R also grows much slower than in the block-tridiagonal case and the number of CG steps also increases more slowly. In total this makes the direct approach much more competitive for larger p and explains the remarkable improvement in Figure 6 compared to Figure 1 with respect to the computation time and the scalability.

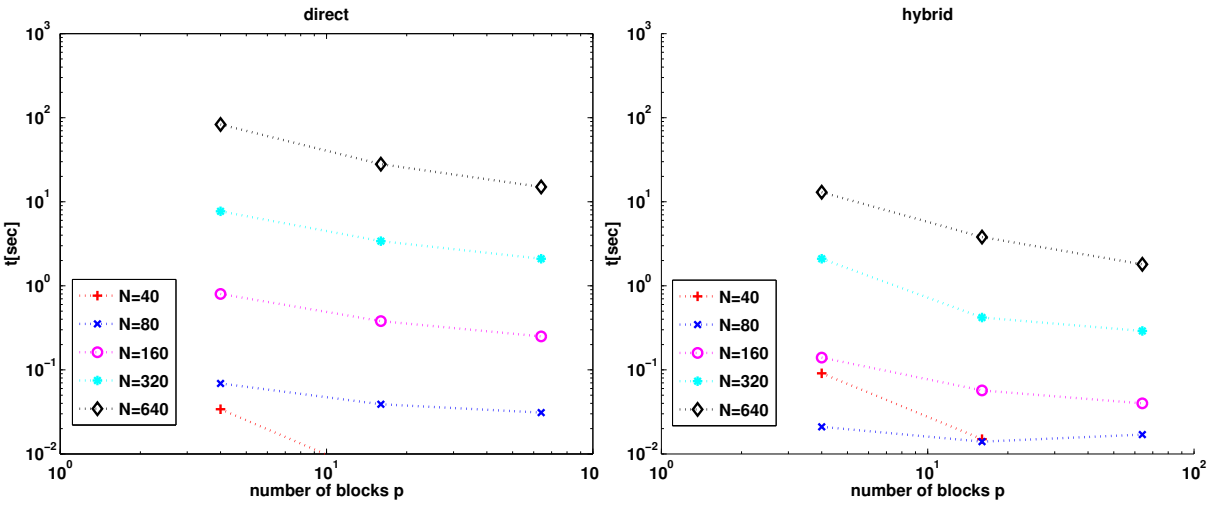


Figure 6: splitting-based direct method (left), splitting-based hybrid method (right)

4 Minimum rank decoupling and completion

We will now discuss the problem of choosing X in (15) in the minimum rank decoupling case. This problem is connected to the problem of matrix completion [26, 27]. For the problem of completion one is interested in determining a suitable X such that $W(X) = \begin{pmatrix} W_{11} & W_{12} \\ W_{21} & X \end{pmatrix}$ has certain desired properties, e.g. a small norm, an inverse with small norm or a small condition number. For details we refer to [26, 27]. Here the completion problem comes along with the choice of X from (15). We will follow the arguments in [26, 27]. Suppose that $\mathcal{E}_M = \{\{r_1, s_1\}, \ldots, \{r_m, s_m\}\}$ with the convention that we use $r_i < s_i$. Given the splitting (13) depending on X we set

$$E^{(1)} = \left(E_1^{(1)}, \ldots, E_m^{(1)}\right), \qquad F^{(1)} = \left(F_1^{(1)}, \ldots, F_m^{(1)}\right),$$
$$E^{(2)} = \left(E_1^{(2)}, \ldots, E_m^{(2)}\right), \qquad F^{(2)} = \left(F_1^{(2)}, \ldots, F_m^{(2)}\right),$$

where for any $\{r_i, s_i\} \in \mathcal{E}_M$ such that $r_i < s_i$ we define

$$E_i^{(1)} = I_{r_i} E_{r_i,s_i},\ E_i^{(2)} = I_{s_i} E_{s_i,r_i}, \quad F_i^{(1)} = I_{r_i} F_{s_i,r_i},\ F_i^{(2)} = I_{s_i} F_{r_i,s_i}.$$

Then the minimum rank decoupling (14) can be written as

$$E(X)X^{-1}F(X)^T = \underbrace{(E^{(1)}X + E^{(2)})}_{E(X)} X^{-1} \underbrace{(F^{(1)}X^T + F^{(2)})^T}_{F(X)^T}$$

and the block diagonal part $C(X)$ is analogously characterized by

$$C(X) = C + E^{(1)}X(F^{(1)})^T + E^{(2)}X^{-1}(F^{(2)})^T.$$

Here, as before, C refers to the unmodified block diagonal part and $A = C(X) - E(X)X^{-1}F(X)^T$. If our matrix A is block-tridiagonal, then $E^{(1)}, F^{(1)}$ refer to modifications in the lower right

block of a diagonal block C_i, whereas $E^{(2)}$, $F^{(2)}$ refer to the upper left corners. Using the Sherman-Morrison-Woodbury formula we find that

$$A^{-1} = C(X)^{-1} + C(X)^{-1}E(X) \underbrace{\left(X - F(X)^T C(X)^{-1} E(X)\right)^{-1}}_{\equiv R(X)^{-1}} F(X)^T C(X)^{-1}$$

and a natural objective is to improve the properties of $C(X)$ or of the coupling system

$$R(X) = X - F(X)^T C(X)^{-1} E(X).$$

Rewriting $R(X)^{-1}$ (again using (2)) we can see that

$$R(X)^{-1} = X^{-1} - X^{-1}(F^{(1)}X^T + F^{(2)})^T A^{-1}(E^{(1)}X + E^{(2)})X^{-1}.$$

Taking into account that usually we only have two factors EF^T instead of three factors $E(X)X^{-1}F(X)^T$, we would factorize $X = X_L X_U$ and replace $E(X)X^{-1}F(X)^T$ by $(E^{(1)}X_L + E^{(2)}X_U^{-1}) \cdot (F^{(1)}X_U^T + F^{(2)}X_L^{-T})^T$. This in turn means that $R(X)$ should approximate X rather than I and similarly, $R(X)^{-1}$ has to approximate X^{-1}. If we wish approximate a multiple αX^{-1} of X^{-1} we conclude that using $Y = X^{-1}$ we obtain in the optimal case

$$0 = \alpha Y - R(X)^{-1} = (\alpha - 1)Y - (F^{(1)} + F^{(2)}Y^T)^T A^{-1}(E^{(1)} + E^{(2)}Y). \tag{16}$$

Note that (16) is called algebraic Riccati equation with respect to Y. For the application of numerical methods for solving Riccati equations we refer to [20, 19, 51, 41, 4, 42]. Here we mention a simple criterion when this quadratic equation simplifies. Since we will not follow this direction in detail we leave the proof to the reader.

Proposition 1 *Suppose that* $(F^{(1)})^T C^{-1} E^{(2)} = 0$ *and* $(F^{(2)})^T C^{-1} E^{(1)} = 0$. *Then (16) is equivalent to*

$$0 = \alpha Y - (D + Y) - (D + Y)B(D + Y), \tag{17}$$

where $B = (F^{(2)})^T A^{-1} E^{(2)}$ *and* $D = (F^{(1)})^T C^{-1} E^{(1)}$.

Example 7 *We continue Examples 3, 4 for the case of a block-tridiagonal partitioning. Note that in the case of minimum rank decoupling we will obviously have* $E = F$ *and*

$$E = \left(\begin{array}{c|c|c} \begin{matrix}0\\I\end{matrix} & & \\ \hline \begin{matrix}I\\0\end{matrix} & \begin{matrix}0\\I\end{matrix} & \\ \hline & \begin{matrix}I\\0\end{matrix} & \begin{matrix}0\\I\end{matrix} \\ \hline & & \begin{matrix}I\\0\end{matrix} \end{array}\right),$$

since the trivial choice $X = I$ modifies the original off-diagonal blocks of type

$$\begin{pmatrix} 0 & I \\ I & 0 \end{pmatrix}$$

in the minimum rank case to blocks of type

$$\begin{pmatrix} I & I \\ I & I \end{pmatrix} = \begin{pmatrix} I \\ I \end{pmatrix} \begin{pmatrix} I & I \end{pmatrix}.$$

In this case we will have

$$E = F = E^{(1)} + E^{(2)} = F^{(1)} + F^{(2)} \equiv \left(\begin{array}{c|c|c} 0 & & \\ I & & \\ \hline 0 & & \\ & 0 & \\ & I & \\ \hline & 0 & \\ & & 0 \\ & & I \\ \hline & & 0 \\ & & \end{array}\right) + \left(\begin{array}{c|c|c} & & \\ 0 & & \\ \hline I & & \\ 0 & & \\ & 0 & \\ \hline & I & \\ & 0 & \\ & & 0 \\ \hline & & I \\ & & 0 \end{array}\right).$$

For modifying $C(X)$ we aim to reduce $\|C(X)\|$ or $\|C(X)^{-1}\|$. Moreover, with respect to the sparsity of C, we cannot afford much more than a diagonal matrix X. As long as $\|X\|, \|X^{-1}\| \leqslant \kappa$ for some constant $\kappa > 0$, the norm of $C(X)$ is suitably bounded. In contrast to that, $\|C(X)^{-1}\|$ might still be large. Completion can be directly used to bound the norm of the inverse of

$$W(X) = \left(\begin{array}{c|cc} C & E^{(1)} & E^{(2)} \\ \hline (F^{(1)})^T & -X^{-1} & 0 \\ (F^{(2)})^T & 0 & -X \end{array}\right)$$

since the associated Schur complement in the top left corner satisfies

$$C(X) = C - \begin{pmatrix} E^{(1)} & E^{(2)} \end{pmatrix} \begin{pmatrix} -X^{-1} & 0 \\ 0 & -X \end{pmatrix}^{-1} \begin{pmatrix} F^{(1)} & F^{(2)} \end{pmatrix}^T.$$

Since $C(X)^{-1}$ is the leading top left block of $W(X)^{-1}$, a bound for the norm of $W(X)$ also leads to a bound for $\|C(X)^{-1}\|$. Following [27] we define α_0 via

$$\alpha_0 = \min\left\{\sigma_{\min}\left[C, E^{(1)}, E^{(2)}\right], \sigma_{\min}\left[C^T, F^{(1)}, F^{(2)}\right]\right\}, \tag{18}$$

where $\sigma_{\min}$ denotes the associated smallest singular value.

Lemma 1 *We define for any $r = 1, \ldots, p$,*

$$C_{E,r} := C_r C_r^T + \sum_{s:s\neq r} E_{rs}E_{rs}^T,\ C_{F,r} := C_r^T C_r + \sum_{s:s\neq r} F_{rs}^T F_{rs}.$$

Then we have that

$$\alpha_0^2 = \min_r\{\lambda_{\min}(C_{E,r}), \lambda_{\min}(C_{F,r})\}.$$

Proof.
It is clear that α_0^2 can be obtained from the smallest eigenvalue of $CC^T + E^{(1)}(E^{(1)})^T + E^{(2)}(E^{(2)})^T$ and $C^TC + (F^{(1)})^TF^{(1)} + (F^{(2)})^TF^{(2)}$.

By definition we have

$$CC^T + E^{(1)}(E^{(1)})^T + E^{(2)}(E^{(2)})^T \;\; = \;\; CC^T + \sum_{i=1}^m E_i^{(1)}(E_i^{(1)})^T + \sum_{i=1}^m E_i^{(2)}(E_i^{(2)})^T,$$

which is block-diagonal by construction and precisely reduces to the block-diagonal matrix $\mathrm{diag}(C_{E,1}, \ldots, C_{E,p})$. Note that since we always assume that $r_i < s_i$, the local sum over all $s : s \neq r$ covers both sums with $E_i^{(1)}$ and $E_i^{(2)}$. Similar arguments apply to $C^TC + (F^{(1)})^TF^{(1)} + (F^{(2)})^TF^{(2)}$. □

As consequence of Lemma 1 we can compute α_0 for each diagonal block separately in parallel. This simplifies the overall complexity.

We note that given $\alpha < \alpha_0$, the general unconstrained solution X rather than $\begin{pmatrix} -X^{-1} & 0 \\ 0 & -X \end{pmatrix}$ of $\|W(X)^{-1}\|_2 \leqslant \frac{1}{\alpha}$ is is stated explicitly in [27]. In addition we would like to point out that the singular values $\sigma_1, \ldots, \sigma_n$ [31] of any matrix C can be determined by

$$\sigma_l \equiv \sigma_l(C) = \max_{\substack{\dim U = l \\ \dim V = l}} \min_{\substack{u\in U\setminus\{0\} \\ v\in V\setminus\{0\}}} \frac{v^TCu}{\|v\|_2\ \|u\|_2}.$$

Furthermore since C and $C(X)$ are block-diagonal, we can compute the singular values of each $C_r(X)$ independently. Having

$$C_r(X) = C_r + \sum_{s:s>r} E_{rs}X_{rs}F_{sr}^T + \sum_{s:s<r} E_{rs}X_{rs}^{-1}F_{sr}^T,$$

for some neighbouring diagonal blocks $s \in \{s_1, \ldots, s_t\}$ of C_r, we can locally choose $U_r^* \perp (F_{s_1,r}, \ldots, F_{s_t,r})$ and $V_r^* \perp (E_{r,s_1}, \ldots, E_{r,s_t})$. We define $q_r = \sum_{s:s\neq r} q_{rs}$, where q_{rs} refers to the number of columns (i.e., the rank) of E_{rs} and F_{rs} and define

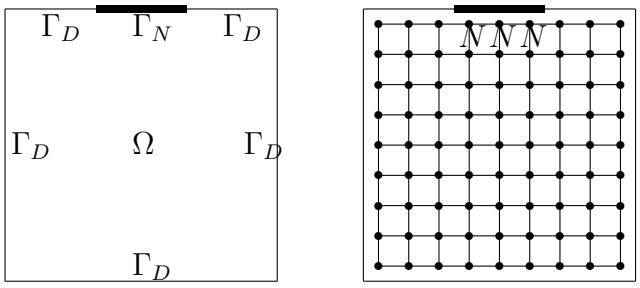
$$\sigma_{n_r-q_r}^* := \min_{\substack{u\in U_r^*\setminus\{0\} \\ v\in V_r^*\setminus\{0\}}} \frac{v^TC_ru}{\|v\|_2\ \|u\|_2}. \tag{19}$$

Then we immediately obtain

$$\sigma_l(C_r) \geqslant \sigma^*_{n_r-q_r} \geqslant \sigma_{n_r}(C_r),\ \sigma_l(C_r(X)) \geqslant \sigma^*_{n_r-q_r} \geqslant \sigma_{n_r}(C_r(X)) \text{ and } \sigma^*_{n_r-q_r} \geqslant \alpha_0$$

for any $l \leqslant n_r - q_r$, $r = 1, \ldots, p$. This also shows that $\sigma^*_{n_r-q_r}$ gives an upper bound for α_0 which cannot be improved. This is also reasonable since the remaining rows and columns of $C_r(X)$ coincide with those of A up to some zeros. Due to the sparsity of the off-diagonal blocks A_{rs}, A_{sr}, our matrices U^*_r and V^*_r would cover many unit vectors associated with unknowns that are not connected with neighbouring blocks in the sense of the underlying graph $G_M(A)$.

Example 8 *We will discuss again the equation $-\Delta u = f$ on the unit square $\Omega = [0,1]^2$ in two spatial dimensions. We give a simplified model of different boundary conditions, namely Dirichlet boundary conditions $u = g$ on Γ_D and some kind of Neumann-type boundary conditions $\partial u/\partial \nu = 0$ on Γ_N.*



To simplify the discussion we use the 5-point-star difference stencil which leads to a matrix with 4 on the main diagonal and -1 in the off-diagonal positions as described in Example 4. At the positions associated with Γ_N we reduce the diagonal entry from 4 to 3 which refers to first order Neumann boundary conditions. We divide the domain into a checker board of 9 subdomains which corresponds to a block-diagonal splitting with 9 diagonal blocks. For each of the diagonal block we sketch the associated relevant singular values. We will choose a grid size of total size 150×150. This means if we have $p = 9 = 3 \times 3$ diagonal blocks, then each diagonal block is of size $n_r = 2500$. The rank q_r is between 100 and 200 depending on the diagonal block. We will compare each local $\sigma^*_{n_r-q_r}$ with

1. *$\sigma_{n_r-q_r}(C_r)$ and $\sigma_{n_r}(C_r)$ of the original block-diagonal matrix and with*
2. *$\sigma_{n_r-q_r}(C_r(I))$ and $\sigma_{n_r}(C_r(I))$ for minimum-rank decoupling using $X = I$.*

$\sigma^*_{n_r-q_r}$

$2.5 \cdot 10^{-3}$	$8.1 \cdot 10^{-3}$	$2.5 \cdot 10^{-3}$
$5.1 \cdot 10^{-3}$	$8.2 \cdot 10^{-3}$	$5.1 \cdot 10^{-3}$
$2.5 \cdot 10^{-3}$	$5.1 \cdot 10^{-3}$	$2.5 \cdot 10^{-3}$

$\sigma_{n_r-q_r}(C_r)$
$\sigma_{n_r}(C_r)$

$4.9 \cdot 10^{-1}$	$7.4 \cdot 10^{-1}$	$4.9 \cdot 10^{-1}$
$2.4 \cdot 10^{-3}$	$7.6 \cdot 10^{-3}$	$2.4 \cdot 10^{-3}$
$7.2 \cdot 10^{-1}$	$9.6 \cdot 10^{-1}$	$7.2 \cdot 10^{-1}$
$4.8 \cdot 10^{-3}$	$7.6 \cdot 10^{-3}$	$4.8 \cdot 10^{-3}$
$4.9 \cdot 10^{-1}$	$7.2 \cdot 10^{-1}$	$4.9 \cdot 10^{-1}$
$2.4 \cdot 10^{-3}$	$4.8 \cdot 10^{-3}$	$2.4 \cdot 10^{-3}$

$\sigma_{n_r-q_r}(C_r(I))$
$\sigma_{n_r}(C_r(I))$

$5.0 \cdot 10^{-1}$	$7.6 \cdot 10^{-1}$	$5.0 \cdot 10^{-1}$
$2.5 \cdot 10^{-3}$	$7.8 \cdot 10^{-3}$	$2.5 \cdot 10^{-3}$
$7.4 \cdot 10^{-1}$	$1.0 \cdot 10^{0}$	$7.4 \cdot 10^{-1}$
$4.9 \cdot 10^{-3}$	$7.9 \cdot 10^{-3}$	$4.9 \cdot 10^{-3}$
$5.0 \cdot 10^{-1}$	$7.4 \cdot 10^{-1}$	$5.0 \cdot 10^{-1}$
$2.5 \cdot 10^{-3}$	$4.9 \cdot 10^{-3}$	$2.5 \cdot 10^{-3}$

We can see in this specific example that $\sigma^*_{n_r-q_r}$ serves as a fairly well upper bound for $\sigma_{n_r}(C)$ and $\sigma_{n_r}(C(I))$. This is easily explained by the nature of the partial differential equation, since $\sigma^*_{n_r-q_r}$ refers to the smallest singular value of the subsystem which leaves out the nodes at the interfaces. This system is in general only slightly smaller but with similar properties as each C_r and $C_r(I)$, except that one can read omitting the nodes near the interfaces as choosing Dirichlet boundary conditions everywhere.

We can now easily apply the analytic solution of the completion problem from [27] but we like to note that the constraint with X and X^{-1} is in general not satisfied. We will focus on each local problem involving the blocks $(r,r),(r,s),(s,r),(s,s)$. This simplifies the completion problem dramatically and also allows to treat it for each pair of neighbouring diagonal blocks separately.

Lemma 2 *Let $\{r,s\} \in \mathcal{E}_M$ such that $r < s$. Let*

$$\alpha_0 = \min\left\{\sigma_{\min}\left(\begin{array}{cc|cc} C_r & 0 & E_{rs} & 0 \\ 0 & C_s & 0 & E_{sr} \end{array}\right),\ \sigma_{\min}\left(\begin{array}{cc|cc} C_r^T & 0 & F_{rs} & 0 \\ 0 & C_s^T & 0 & F_{sr} \end{array}\right)\right\}.$$

Given $\alpha < \alpha_0$ such that α is not a singular value of C_r or C_s, then the general solution X of

$$\left\|\left(\begin{array}{cc|cc} C_r & 0 & E_{rs} & 0 \\ 0 & C_s & 0 & E_{sr} \\ \hline F_{sr}^T & 0 & X_{rr} & X_{rs} \\ 0 & F_{rs}^T & X_{sr} & X_{ss} \end{array}\right)^{-1}\right\|_2 \leqslant \frac{1}{\alpha}$$

satisfies $X_{rs} = 0 = X_{sr}^T$ and

$$\begin{aligned} X_{rr} &= F_{sr}^T C_r^T \left(C_r C_r^T - \alpha^2 I\right)^{-1} E_{rs} + \alpha Y_{rr}, \\ X_{ss} &= F_{rs}^T C_s^T \left(C_s C_s^T - \alpha^2 I\right)^{-1} E_{sr} + \alpha Y_{ss}, \end{aligned} \tag{20}$$

where Y_{rr}, Y_{ss} may be any matrices such that

$$\begin{aligned} Y_{rr}\left(I - E_{rs}^T(C_rC_r^T + E_{rs}E_{rs}^T - \alpha^2 I)^{-1}E_{rs}\right)Y_{rr}^T &\geqslant I + F_{sr}^T(C_r^TC_r - \alpha^2 I)^{-1}F_{sr} \\ Y_{ss}\left(I - E_{sr}^T(C_sC_s^T + E_{sr}E_{sr}^T - \alpha^2 I)^{-1}E_{sr}\right)Y_{ss}^T &\geqslant I + F_{rs}^T(C_s^TC_s - \alpha^2 I)^{-1}F_{rs} \end{aligned}$$

in the sense of quadratic forms.

Proof.
We set

$$\hat{C} = \mathrm{diag}(C_r, C_s), \hat{E} = \mathrm{diag}(E_{rs}, E_{sr}), \hat{F} = \mathrm{diag}(F_{sr}, F_{rs}).$$

Except for the block-diagonal structure of X this lemma exactly reveals Theorem 3.1 from [27], which states that there exists X such that

$$X = \hat{F}^T\hat{C}^T\left(\hat{C}\hat{C}^T - \alpha^2 I\right)^{-1}\hat{E} + \alpha Y, \text{ where}$$

$$Y\left(I - \hat{E}^T(\hat{C}\hat{C}^H + \hat{E}\hat{E}^T - \alpha^2 I)^{-1}\hat{E}\right)Y^T \geqslant I + \hat{F}^T(\hat{C}^T\hat{C} - \alpha^2 I)^{-1}\hat{F}.$$

The underlying block structure of $\hat{C}$, $\hat{E}$ and $\hat{F}$ obviously induce the block structure of X. □

We like to mention that often enough (say in applications arising from partial differential equations), the diagonal part of a matrix is well-conditioned enough to be used, i.e., rather than using the complete inverses in Lemma 2, we could work with the diagonal parts before inverting the matrices. In this case, simplified versions of X_{rr}, X_{ss} from Lemma 2 could be used to define $-X_{rs}^{-1}, -X_{rs}$.

We like to mention that the Hermitian case can be treated more easily as stated in Theorem 2.1 in [27]. Even if A and C are symmetric and positive definite, $E = F$ and if X is chosen positive definite as well, the constraint minimization problem

$$\left\| \left(\begin{array}{c|cc} C & E^{(1)} & E^{(2)} \\ \hline (E^{(1)})^T & -X^{-1} & 0 \\ (E^{(2)})^T & 0 & -X \end{array}\right)^{-1} \right\|_2 \leqslant \frac{1}{\alpha}$$

refers to a Hermitian but indefinite problem. In this case we always have

$$\lambda_l(C_r(X)) \equiv \sigma_l(C_r(X)) \geqslant \lambda_l(C_r)$$

since in the sense of quadratic forms we have

$$C_r(X) = C_r + \sum_{s:s>r} E_{rs}X_{rs}E_{rs}^T + \sum_{s:s<r} E_{rs}X_{rs}^{-1}E_{rs}^T \geqslant C_r.$$

This can be observed in Example 8. Thus $\|C_r(X)^{-1}\|$ can only become better than $\|C_r^{-1}\|$ and the same applies to the condition number as long as $\|C_r(X)\| \approx \|C_r\|$.

Example 9 *We will continue Example 8, except that the elliptic operator $-u_{xx} - u_{yy}$ is now replaced by $-\varepsilon u_{xx} - \varepsilon u_{yy}$ with varying coefficient ε as illustrated below.*

24	4	24
4	1955	4
24	4	24

For simplicity we assume that the larger value is taken on the interfaces. We like to stress that each local interface between two neighbouring diagonal blocks is essentially of the following type for a suitable $\alpha \geqslant \beta$ (e.g. $\alpha = 3 \cdot 1955$, $\beta = 4$)

$$\begin{pmatrix} A_{rr} & A_{rs} \\ A_{sr} & A_{ss} \end{pmatrix} = \left(\begin{array}{ccc|ccc} \alpha+\beta & -\beta & & & & \\ -\beta & \ddots & \ddots & & -\beta I & \\ & \ddots & \ddots & & & \\ \hline & & & 4\beta & -\beta & \\ & -\beta I & & -\beta & \ddots & \ddots \\ & & & & \ddots & \ddots \end{array}\right)$$

Since minimum rank decoupling adds positive semidefinite matrices to the diagonal blocks, we propose to move the interface nodes (which reflect the jumps of ε), to the diagonal blocks with larger ε. In this case the diagonal entries of the blocks with larger ε have relatively small diagonal entries at the nodes connected to the neighbouring blocks, e.g., for the $(2,2)$ block, the diagonal entries are $4\varepsilon = 7820$ for the inner nodes,whereas the diagonal entries of the $(2,2)$ system in the extremal four corners are only half as big. Since we work with splittings rather than with removing nodes to a remaining Schur complement system, this effect cannot be avoided. We illustrate this effect by stating the largest and smallest singular value $\sigma_{\max}(C_r)$, $\sigma_{\min}(C_r)$ for each diagonal block C_r of the unmodified block diagonal matrix C (we will use $N = 40$).

$\sigma_1(C_r)$
$\sigma_{n_r}(C_r)$

$1.9 \cdot 10^2$ $5.7 \cdot 10^{-2}$	$3.2 \cdot 10^1$ $4.7 \cdot 10^{-2}$	$1.9 \cdot 10^2$ $5.7 \cdot 10^{-2}$
$3.2 \cdot 10^1$ $2.9 \cdot 10^{-2}$	$1.6 \cdot 10^4$ $3.9 \cdot 10^{-1}$	$3.2 \cdot 10^1$ $2.9 \cdot 10^{-2}$
$1.9 \cdot 10^2$ $5.7 \cdot 10^{-2}$	$3.2 \cdot 10^1$ $2.9 \cdot 10^{-2}$	$1.9 \cdot 10^2$ $5.7 \cdot 10^{-2}$

Knowing that for large ε the diagonal entries close to the interfaces are less than in the interior of the diagonal block, one can use this information to increase the diagonal entries, e.g., the $(2,2)$ block. Choosing $X = 40 \cdot I$ for all X_{rs} in the $(2,2)$ block and X_{rs}^{-1} outside improves the condition number dramatically. Similarly, for the four blocks in the corner of

the domain we could increase the diagonal entries further using $X = 4 \cdot I$. This improves the condition number of several diagonal blocks significantly while other diagonal blocks are hardly affected.

$$\sigma_1(C_r(X))$$
$$\sigma_{n_r}(C_r(X))$$

$1.9 \cdot 10^2$ $7.4 \cdot 10^{-2}$	$3.2 \cdot 10^1$ $4.7 \cdot 10^{-2}$	$1.9 \cdot 10^2$ $7.4 \cdot 10^{-2}$
$3.2 \cdot 10^1$ $3.0 \cdot 10^{-2}$	$1.6 \cdot 10^4$ $2.4 \cdot 10^1$	$3.2 \cdot 10^1$ $3.0 \cdot 10^{-2}$
$1.9 \cdot 10^2$ $7.4 \cdot 10^{-2}$	$3.2 \cdot 10^1$ $3.0 \cdot 10^{-2}$	$1.9 \cdot 10^2$ $7.4 \cdot 10^{-2}$

Finally, with respect to ε, the best condition is obtained in the order 1955/04/24. This example demonstrates that completion is able to improve the condition number up to two orders of magnitude in this example and leading to a lower rank between A and $C(X)$ at the same time. We also reiterate that part of this success is moving the interface nodes to the diagonal blocks with larger ε.

5 Algebraic Multilevel Preconditioning

So far we have discussed how to improve the diagonal blocks in block-diagonal splitting and for both approaches, the splitting-type approach and the partitioning-type approach we have assumed that the systems are solved directly. Often enough, in practice we prefer to solve these systems iteratively using preconditioned Krylov subspace solvers. Since many application problems arise from the discretization of partial differential equations, preconditioning methods based on (algebraic) multilevel methods are preferred. Therefore this section will discuss algebraic multilevel methods and we will also give some ideas how splitting or partitioning the original system as in Section 3 may be used to parallelize the approach. Multilevel methods [34, 67] in general are popular for solving systems arising from partial differential equations. However, when information about some kind of grid hierarchy is not available, one often has to use algebraic approaches to construct multilevel methods which mimic the behaviour of multigrid methods using analogous terminology such as smoothing and coarse grid correction. As long as the system arises from partial differential equations, say using finite element discretization, one has additional information about the underlying physical problem and in this case one may use agglomeration techniques in order to glue together clusters of element matrices to successively build an algebraic coarsening hierarchy (cf. e.g. [70, 17, 35, 21]). Somehow in the opposite direction of this development, recent approaches to finite element aggregation are based on a relatively simple aggregation approach but instead they are supplemented with flexible Krylov subspace solvers at every level (also referred to as K-cycle), see e.g. [52, 55, 54]. In a similar direction, algebraic multilevel Krylov methods use K-cycles as well but shift the

coarse grid operator additionally [28, 29]. Further approaches such as [58, 64, 5, 6] strongly focus on the underlying matrix and construct the multilevel hierarchy algebraically. This eventually justifies to employ multilevel incomplete factorization as basis of the coarsening process, either when there is a strong link to an underlying partial differential equation [5, 6, 56, 72, 73] or using purely algebraic methodology such as diagonal dominance, independent sets or related ideas, see e.g., [68, 16, 61, 14]. We will link earlier work on algebraic multilevel methods [12, 13] to illustrate the theoretical and practical performance of the multilevel incomplete factorization method, therefore we will restrict the description to these class of methods. Following [14], we rescale and reorder the initial system $A \in \mathbb{R}^{n,n}$ to obtain

$$\hat{A} = P^T DADP,$$

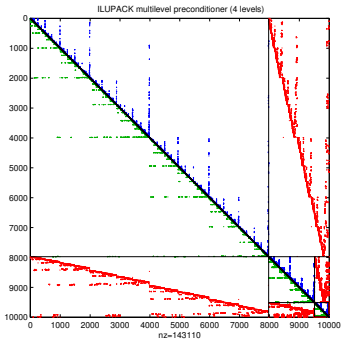
where $D \in \mathbb{R}^{n,n}$ is a nonsingular diagonal matrix and $P \in \mathbb{R}^{n,n}$ is a permutation matrix. Here D is chosen such that DAD has all diagonal entries equal to 1. Fill-reducing algorithms such as (approximate) minimum degree [2] or multilevel nested dissection [38, 39] can be used afterwards to prevent (incomplete) factorization methods from producing too much fill. Then we perform a partial approximate LDL^T factorization of type

$$\Pi^T \hat{A} \Pi = \begin{pmatrix} B & E^T \\ E & C \end{pmatrix} = \begin{pmatrix} L_B & 0 \\ L_E & I \end{pmatrix} \begin{pmatrix} D_B & 0 \\ 0 & S_C \end{pmatrix} \begin{pmatrix} L_B^T & E_F^T \\ 0 & I \end{pmatrix} + \mathcal{E} \tag{21}$$

where we allow further symmetric permutations $\Pi \in \mathbb{R}^{n,n}$ for stability of the factorization. Here D_B refers to a nonsingular diagonal matrix, L_B is lower triangular with unit diagonal part and $\mathcal{E}$ refers to some appropriate perturbation. Furthermore we have $B \approx L_B D_B L_B^T$, $L_E D_B L_B^T \approx E$. Eventually we obtain a remaining approximate Schur complement $S_C \approx C - EB^{-1}F$ that consists of all delayed pivots which were not suitable to serve as pivots during the approximate factorization. Applying the whole procedure to S_C then leads to a multilevel incomplete factorization, where level-by-level, the size of the remaining Schur complement is reduced until it reaches a size such that it can be easily factorized, say, by a dense Cholesky factorization method. Multilevel incomplete factorization methods as described here are well-established, see e.g. [5, 68, 16, 61, 60]. For ease of notation we collect the permutation matrices Π and P in a single permutation matrix and call it again P. Often enough, L_B is not stored explicitly, but implicitly defined via $L_E := EL_B^{-T} D_B^{-1}$ saving some memory at the cost of solving an additional system. We like to point out that in this case $\mathcal{E}$ from (21) has an empty $(1,2)$ block and $(2,1)$ block. The same applies if $S_C := C - L_E D_B L_E^T$ is chosen. Having only one block $\mathcal{E}_B$ different from zero does not necessarily mean that the approximate factorization is more accurate, since this $\mathcal{E}_B$ propagates through the factorization and using the approximate factorization for preconditioning requires to apply the inverses in equation (21) which may lift the influence of $\mathcal{E}$.

Example 10 *We illustrate for the model problem $-\Delta u = f$ in two spatial dimensions and $N = 100$ grid points in each direction the skeleton of a multilevel factorization.*

<table>
<tr><td>$L_{B,1}+L_{B,1}^T$</td><td colspan="3">E_1^T</td></tr>
<tr><td rowspan="3">E_1</td><td>$L_{B,2}+L_{B,2}^T$</td><td colspan="2">E_2^T</td></tr>
<tr><td rowspan="2">E_2</td><td>$L_{B,3}+L_{B,3}^T$</td><td>E_3^T</td></tr>
<tr><td>E_3</td><td>$\ddots$</td></tr>
</table>



We like to emphasize that at least a single level factorization yields an approximate inverse of type

$$\hat{A}^{-1} \approx \begin{pmatrix} I \\ 0 \end{pmatrix} (L_B D_B L_B^T)^{-1} \begin{pmatrix} I \\ 0 \end{pmatrix}^T + Q S_C^{-1} Q^T \text{ where } Q = \begin{pmatrix} -L_B^{-T} L_E^T \\ I \end{pmatrix}. \tag{22}$$

Approximate inverse preconditioners of this type are well-studied in literature and we like to study how a preconditioner of type

$$M^{(1)} = LL^T + Q S_C^{-1} Q^T \tag{23}$$

will approximate $\hat{A}$ for some nonsingular $L \in \mathbb{R}^{n,n}$. In the limit, when

$$LL^T \to \begin{pmatrix} I \\ 0 \end{pmatrix} (L_B D_B L_B^T)^{-1} \begin{pmatrix} I \\ 0 \end{pmatrix}^T$$

we will also obtain some information about the (multilevel) incomplete factorization as preconditioner. Here we can imagine that for some positive σ, τ, we could have, e.g.,

$$L = \begin{pmatrix} \sigma L_B^{-T} D_B^{-1/2} & 0 \\ 0 & \tau I \end{pmatrix}.$$

We denote by m the remaining block size of $C \in \mathbb{R}^{m,m}$. It was shown in [12], that the optimal preconditioner for $\hat{A}$ of type $LL^T + QZ^{-1}Q^T$, $Q \in \mathbb{R}^{n,m}$, $Z \in \mathbb{R}^{m,m}$, with respect to the condition number of the preconditioned system is given by choosing Q as the matrix of eigenvectors $Q_{opt} = [q_1, \ldots, q_m]$ of $L^T \hat{A} L$ with respect to its m smallest eigenvalues $\lambda_1, \ldots, \lambda_m$ and $Z = Q^T A Q$ is almost optimal. It is obvious that for any nonsingular X, $Q_{opt} \to Q_{opt} X$, $Z \to (X^{-T} Z X^{-1})$ is optimal as well, i.e., Q has to approximate the invariant subspace of $L^T \hat{A} L$ associated with its smallest eigenvalues. Taking into account the optimality of Q_{opt} the natural question arises for the preconditioner $M^{(1)}$ from (23) how close the specific choice Q matches the optimal Q_{opt}. We like to mention that $L^T \hat{A} L$ must have eigenvalues less than or equal to 1 which is satisfied for sufficiently small σ and τ. Note also that since we have scaled the original system A initially and since $D_B^{-1/2} L_B^{-1} B L_B^{-T} D_B^{-1/2} \approx I$, σ and τ need not be chosen too small. Indeed we may expect

that $\sigma, \tau = \mathcal{O}(1)$ is already sufficient. One can also verify easily that in the limit case as $\tau \to 0$, we have

$$(L^T \hat{A} L)^{-1} = \mathcal{O}(1) + L^{-1} \begin{pmatrix} -B^{-1}E^T \\ I \end{pmatrix} (C - EB^{-1}E^T)^{-1} \begin{pmatrix} -B^{-1}E^T \\ I \end{pmatrix}^T L^{-T}.$$

This illustrates that asymptotically as $\tau \to 0$ the largest m eigenvalues of $(L^T \hat{A} L)^{-1}$ and their associated invariant subspace is fairly well approximated by

$$L^{-1} \begin{pmatrix} -B^{-1}E^T \\ I \end{pmatrix}$$

which is therefore close to the optimal rank m choice. This in turn justifies choosing

$$Q = \begin{pmatrix} -L_B^{-T} L_E^T \\ I \end{pmatrix}$$

for the preconditioner $M^{(1)}$ and $S_C \approx Q^T A Q$ as almost optimal choice. We will next illustrate how Theorem 4 from [12] describes the quality of the preconditioner $M^{(1)}$ and we will further use this Theorem in order to improve the multilevel incomplete factorization preconditioner (21).

There are two key properties that need to be fulfilled. First of all, we need W such that

$$W^T \hat{A} Q = 0 \text{ and } \Delta W^T \hat{A} W - W^T L^{-T} L^{-1} W \text{ positive semidefinite} \tag{24}$$

for some $\Delta > 0$. Second, the approximate Schur complement S_C has to satisfy

$$\gamma Q^T \hat{A} Q \leqslant S_C \leqslant \Gamma Q^T \hat{A} Q$$

in the sense of quadratic forms for some $0 < \gamma \leqslant \Gamma$. Then

$$\operatorname{cond}(\left(M^{(1)}\right)^{-1/2} \hat{A} \left(M^{(1)}\right)^{-1/2}) \leqslant \frac{\gamma}{(\gamma + 1) \max\{\Gamma, \Delta\}}.$$

While γ and Γ are quite natural bounds, the delicate question is the size of Δ. One can easily verify that using $E = L_E D_B L_B^T$ and $Z = C - L_E D_B L_E^T$ we have

$$\hat{A} Q = \begin{pmatrix} -\mathcal{E}_B \tilde{B}^{-1} E^T \\ Z \end{pmatrix}, \text{ where } \tilde{B} = L_B D_B L_B^T. \tag{25}$$

This allows to define

$$W^T := \begin{pmatrix} I & -\mathcal{E}_B \tilde{B}^{-1} E^T Z^{-1} \end{pmatrix}$$

and to bound Δ. We will not follow this approach in detail and use a different way to examine the multilevel ILU as preconditioner, but even here we can immediately see that $\mathcal{E}_B \tilde{B}^{-1} E^T S_C^{-1}$ plays a central role. Usually the multilevel factorization on each level is set

up such that B can be easily approximated by $\tilde{B}$ using some criterion such as diagonal dominance or diagonal blocks of small size whereas S_C is usually more critical. This in turn means that even a small error $\mathcal{E}_B$ may be amplified by S_C^{-1} significantly. This is in line with algebraic multigrid theory (see e.g. [53]), that for approximate inverses (here $\tilde{B}^{-1}$) in multigrid methods it is not enough to approximate the original matrix B^{-1} sufficiently.

We will now give a simple theorem to state the approximation quality of (21) directly.

Theorem 1 *Consider the approximate factorization from (21) and assume that* $E = L_E D_B L_B^T$. *Furthermore, suppose that* $\tilde{B} = L_B D_B L_B^T$ *satisfies*

$$\lambda B \leqslant \tilde{B} \leqslant \Lambda B$$

for some $0 < \lambda \leqslant \Lambda$ *and that there exist* $0 < \gamma \leqslant \Gamma$ *such that*

$$\gamma\ Z \leqslant S_C \leqslant \Gamma Z, \text{ where } Z = C - E\tilde{B}^{-1}E^T$$

and we assume that Z *is positive definite. Define the preconditioned system* T *by*

$$T = \begin{pmatrix} D_B & 0 \\ 0 & S_C \end{pmatrix}^{-1/2} \begin{pmatrix} L_B & 0 \\ L_E & I \end{pmatrix}^{-1} \hat{A} \begin{pmatrix} L_B & 0 \\ L_E & I \end{pmatrix}^{-T} \begin{pmatrix} D_B & 0 \\ 0 & S_C \end{pmatrix}^{-1/2}.$$

Then

$$\operatorname{cond}(T) \leqslant \frac{\max\{\Lambda, \Gamma\}(\frac{\Lambda}{\lambda} + \sqrt{\Lambda}\|H\|_2)}{\min\{\Lambda, \gamma\}(1 - \sqrt{\Lambda}\|H\|_2)},$$

where

$$H = D_B^{-1/2} L_B^{-1} \mathcal{E}_B \tilde{B}^{-1} E^T Z^{-1/2},$$

provided that $\sqrt{\Lambda}\|H\|_2 < 1$.

Proof.
We have that

$$C - L_E D_B L_E^T = C - (EL_B^{-T} D_B^{-1}) D_B (EL_B^{-T} D_B^{-1})^T = C - E\tilde{B}^{-1}E^T = Z.$$

From (25) we immediately obtain

$$\hat{A} \begin{pmatrix} -L_B^{-T} L_E^T \\ I \end{pmatrix} = \hat{A}Q = \begin{pmatrix} -\mathcal{E}_B \tilde{B}^{-1} E^T \\ Z \end{pmatrix}.$$

We define T_Λ via

$$T_\Lambda := \begin{pmatrix} \frac{1}{\Lambda} D_B & 0 \\ 0 & Z \end{pmatrix}^{-1/2} \begin{pmatrix} L_B & 0 \\ L_E & I \end{pmatrix}^{-1} \hat{A} \begin{pmatrix} L_B & 0 \\ L_E & I \end{pmatrix}^{-T} \begin{pmatrix} \frac{1}{\Lambda} D_B & 0 \\ 0 & Z \end{pmatrix}^{-1/2}.$$

Since we know that Q is the second block column of $\begin{pmatrix} I & 0 \\ L_E L_B^{-1} & I \end{pmatrix}^{-T}$ it follows that

$$\begin{aligned} T_\Lambda &= \begin{pmatrix} \sqrt{\Lambda} D_B^{-1/2} L_B^{-1} & 0 \\ 0 & Z^{-1/2} \end{pmatrix} \begin{pmatrix} B & -\mathcal{E}_B \tilde{B}^{-1} E^T \\ -\mathcal{E}_B \tilde{B}^{-1} E^T & Z \end{pmatrix} \begin{pmatrix} \sqrt{\Lambda} L_B^{-T} D_B^{-1/2} & 0 \\ 0 & Z^{-1/2} \end{pmatrix} \\ &= \begin{pmatrix} \Lambda D_B^{-1/2} L_B^{-1} B L_B^{-T} D_B^{-1/2} & -\sqrt{\Lambda} D_B^{-1/2} L_B^{-1} \mathcal{E}_B \tilde{B}^{-1} E^T Z^{-1/2} \\ -\sqrt{\Lambda} Z^{-1/2} \mathcal{E}_B \tilde{B}^{-1} E^T L_B^{-T} D_B^{-1/2} & I \end{pmatrix}. \end{aligned}$$

We can see that the $(1,2)$ block exactly refers to $-\sqrt{\Lambda}H$. Since $\tilde{B} \leqslant \Lambda B$ in quadratic forms it follows that $\Lambda D_B^{-1/2} L_B^{-1} B L_B^{-T} D_B^{-1/2} \geqslant I$. This in turn implies that

$$T_\Lambda - \begin{pmatrix} I & -\sqrt{\Lambda}H \\ -\sqrt{\Lambda}H^T & I \end{pmatrix} = \begin{pmatrix} \Lambda D_B^{-1/2} L_B^{-1} B L_B^{-T} D_B^{-1/2} - I & 0 \\ 0 & 0 \end{pmatrix}$$

is positive semidefinite. Thus on one hand we have

$$\lambda_{\min}(T_\Lambda) > 1 - \sqrt{\Lambda}\|H\|_2,$$

provided that $\Lambda\|H\|_2 < 1$. On the other hand we have

$$\lambda_{\max}(T_\Lambda) = \|T_\Lambda\|_2 \leqslant \|\Lambda D_B^{-1/2} L_B^{-1} B L_B^{-T} D_B^{-1/2}\|_2 + \sqrt{\Lambda}\|H\|_2 \leqslant \frac{\Lambda}{\lambda} + \sqrt{\Lambda}\|H\|_2.$$

To conclude the proof, we point out that the preconditioned system refers to $T \equiv T_1$ and we obviously have that

$$\min\{\frac{1}{\Lambda}, \frac{1}{\Gamma}\} \begin{pmatrix} \Lambda \tilde{B}^{-1} & 0 \\ 0 & Z^{-1} \end{pmatrix} \leqslant \begin{pmatrix} \tilde{B}^{-1} & 0 \\ 0 & S_C^{-1} \end{pmatrix} \leqslant \max\{\frac{1}{\Lambda}, \frac{1}{\gamma}\} \begin{pmatrix} \Lambda \tilde{B}^{-1} & 0 \\ 0 & Z^{-1} \end{pmatrix}$$

which directly implies

$$\lambda_{\min}(T) \geqslant \min\{\frac{1}{\Lambda}, \frac{1}{\Gamma}\}(1 - \sqrt{\Lambda}\|H\|_2),\ \lambda_{\max}(T) \leqslant \max\{\frac{1}{\Lambda}, \frac{1}{\gamma}\}(\frac{\Lambda}{\lambda} + \sqrt{\Lambda}\|H\|_2).$$

□

We give an interpretation of the bound obtained by Theorem 1. In practice, λ and Λ are expected to be close to 1, so this effect can be ignored, i.e. we essentially have

$$\mathrm{cond}(T) \lessapprox \frac{\Gamma(1 + \|H\|_2)}{\gamma(1 - \|H\|_2)}.$$

Furthermore we note that

1. $\hat{A}$ is diagonally scaled, thus $\|E\|$ is moderately bounded,

2. $\|\mathcal{E}_B\|$ is considerably small when B is well-suited (say diagonally dominant or close to it).

Thus the main effects are how well S_C approximates Z and how $Z^{-1/2}$ amplifies the remaining small terms in $\|H\|_2$ which is not known in advance.

There are some techniques to keep the influence of $Z^{-1/2}$ smaller and to improve approximating Z by S_C, which we will discuss in the sequel. First of all, we like to point out that similar influence of Z^{-1} or S_C^{-1} is also illustrated by (24). We can improve Δ (cf. [13]) by considering

$$\Delta W^T \hat{A}^2 W - W^T \hat{A} W$$

instead. Besides, considering the preconditioner $M^{(2)}$ from [12]

$$M^{(2)} = 2LL^T - LL^T \hat{A} LL^T + (I - LL^T A) \begin{pmatrix} -\tilde{B}^{-1}E^T \\ I \end{pmatrix} S_C^{-1} \begin{pmatrix} -\tilde{B}^{-1}E^T \\ I \end{pmatrix}^T (I - ALL^T)$$

will yield improved bounds since in this case, essentially only

$$\Delta W^T (2\hat{A}^2 - \hat{A}^3) W, W^T \hat{A} W$$

are taken into account for the estimates (cf. Theorem 4 in [12]). We will not go into the details of deriving bounds for this case. but mention that this preconditioner one can read as replacing $\tilde{B}$ by more accurate approximations and thus reducing the error $\mathcal{E}_B$. Indeed, $M^{(2)}$ is obtained from the simple 2-level multilevel scheme

$$\begin{aligned} & I - M^{(2)}\hat{A} \\ \equiv \; & (I - \begin{pmatrix} \tilde{B}^{-1} & 0 \\ 0 & 0 \end{pmatrix} \hat{A})(I - \begin{pmatrix} -\tilde{B}^{-1}E^T \\ I \end{pmatrix} S_C^{-1} \begin{pmatrix} -\tilde{B}^{-1}E^T \\ I \end{pmatrix}^T \hat{A})(I - \begin{pmatrix} \tilde{B}^{-1} & 0 \\ 0 & 0 \end{pmatrix} \hat{A}) \end{aligned} \tag{26}$$

which demonstrates how a multilevel incomplete factorization can be easily upgraded to serve as algebraic multigrid method, see e.g. [57]. In the sense of multigrid methods, the first and the third factor are usually considered as smoothing while the factor in the middle reveals the coarse grid correction. We will demonstrate the difference between the simple multilevel incomplete factorization and its induced algebraic multigrid.

Example 11 *Again we will consider the well-known problem $-\Delta u = f$ on a unit square in two spatial dimensions with Dirichlet boundary conditions and N grid points in every direction. We compare*

1. *the multilevel incomplete factorization from [14] with its default options (in particular a drop tolerance of 10^{-2} and preconditioned conjugate gradient method that stops when the relative error in the energy norm drops below 10^{-6}). This gives a multilevel incomplete Cholesky factorization (MLIC),*

2. the associated algebraic multigrid method associated with $M^{(2)}$.

Both operators will serve as preconditioners for the conjugate gradient method. Beside the computation time and the number of CG steps we will state the relative fill $\frac{nz(LDL^T)}{nz(A)}$ for the number of nonzero entries.

N	*MLIC computation*		*MLIC-CG*		$M^{(2)}$*-CG*	
	[sec]	*fill*	*[sec]*	*steps*	*[sec]*	*steps*
100	$6.6 \cdot 10^{-2}$	2.7	$5.1 \cdot 10^{-2}$	26	$9.1 \cdot 10^{-2}$	23
200	$2.5 \cdot 10^{-1}$	2.8	$3.4 \cdot 10^{-1}$	43	$6.7 \cdot 10^{-1}$	38
400	$1.3 \cdot 10^{0}$	2.8	$3.3 \cdot 10^{0}$	77	$6.7 \cdot 10^{0}$	66
800	$5.8 \cdot 10^{0}$	2.8	$3.0 \cdot 10^{1}$	135	$6.6 \cdot 10^{1}$	119
1600	$2.6 \cdot 10^{1}$	2.9	$2.4 \cdot 10^{2}$	237	$5.7 \cdot 10^{2}$	221

As we can see, although the number of iteration steps is slightly reduced the total computational amount of work even increases. Besides, none of the methods scales linearly.

We can see from Example 11, simply improving the quality of $\tilde{B}$ is not enough which is well-known (see e.g. [53]). Here the source is two-fold. On the one hand one has to ensure that the perturbation is sensitive with respect to H. On the other hand S_C needs to approximate Z sufficiently. Here we attempt to approach these requirement by computing a modified multilevel incomplete factorization that is exact for the vector e with all ones. Besides, when upgrading the multilevel ILU to an algebraic multigrid method, more natural improvements can be achieved borrowing the smoothing and coarse grid methodology from AMG. In the sense of AMG, in (26) we replace

$$\begin{pmatrix} \tilde{B}^{-1} & 0 \\ 0 & 0 \end{pmatrix} \longrightarrow \begin{cases} G^{-1} \\ G^{-T} \end{cases}$$

by more general approximations such as the inverse of the lower triangular part of $\hat{A}$ (Gauss-Seidel) or a damped inverse of the diagonal part (Jacobi). To preserve symmetry, one uses G^{-T} in the first factor of (26) and G^{-1} in the third factor. Additionally, since the middle factor in (26) solves the coarse grid only approximately in the multilevel case, one recursive call refers to the traditional V-cycle while two recursive calls are refered to as W-cycle. We note that since the factorization in (21) is exact for e, we have $\mathcal{E}_B e = 0$, $Be = \tilde{B}e$ and $S_C e = Ze$ and e can be regarded as sample vector for the low frequencies. Several algebraic multigrid methods and incomplete factorization methods make use of this improvement, see e.g. [64, 72, 73].

Example 12 *We will continue Example 11 and consider the following preconditioners in two spatial dimensions, except that now the multilevel incomplete factorization (21) is exact for e. We will compare the following preconditioners.*

1. *Modified multilevel incomplete Cholesky (MLIC)*
2. *V-cycle AMG with one Gauss-Seidel forward and one Gauss-Seidel backward smoothing step (AMGV),*
3. *W-cycle AMG with one Gauss-Seidel forward and one Gauss-Seidel backward smoothing step (AMGW).*

N	*MLIC comput.*		*MLIC-CG*		*AMGV-CG*		*AMGW-CG*	
	[sec]	*fill*	*[sec]*	*steps*	*[sec]*	*steps*	*[sec]*	*steps*
100	$6.5 \cdot 10^{-2}$	2.9	$3.2 \cdot 10^{-2}$	15	$6.9 \cdot 10^{-2}$	14	$8.7 \cdot 10^{-2}$	8
200	$3.1 \cdot 10^{-1}$	3.0	$1.5 \cdot 10^{-1}$	18	$3.9 \cdot 10^{-1}$	16	$4.4 \cdot 10^{-1}$	8
400	$1.6 \cdot 10^{0}$	3.1	$9.5 \cdot 10^{-1}$	20	$2.6 \cdot 10^{0}$	19	$2.6 \cdot 10^{0}$	8
800	$7.4 \cdot 10^{0}$	3.1	$5.4 \cdot 10^{0}$	23	$1.6 \cdot 10^{1}$	21	$1.5 \cdot 10^{1}$	8
1600	$3.4 \cdot 10^{1}$	3.2	$2.6 \cdot 10^{1}$	25	$8.1 \cdot 10^{1}$	24	$8.3 \cdot 10^{1}$	9

Although the number of CG steps, in particular for W-cycle, is better, the overall complexity is best for the multilevel ILU, because the approach is simpler and the intermediate coarse grid systems are not required. The latter are known to fill-up during the coarsening process.

Example 13 *We will conclude this section with another example AF_SHELL3 from sheet metal forming, available at the University of Florida sparse matrix collection, to demonstrate the flexibility of algebraic multilevel ILU preconditioning. The symmetric positive definite system has a size of* $n = 504'855$ *with approximately* 35 *nonzero entries per row. We will compare the methods without test vector* e *and with* e*.*

without test vector e

MLIC comput.		*MLIC-CG*		*AMGV-CG*		*AMGW-CG*	
[sec]	*fill*	*[sec]*	*steps*	*[sec]*	*steps*	*[sec]*	*steps*
$5.1 \cdot 10^{1}$	3.9	$9.7 \cdot 10^{1}$	79	$2.5 \cdot 10^{2}$	82	$3.2 \cdot 10^{2}$	42

with test vector e

MLIC comput.		*MLIC-CG*		*AMGV-CG*		*AMGW-CG*	
[sec]	*fill*	*[sec]*	*steps*	*[sec]*	*steps*	*[sec]*	*steps*
$6.0 \cdot 10^{1}$	4.2	$5.0 \cdot 10^{1}$	40	$1.2 \cdot 10^{2}$	38	$1.9 \cdot 10^{2}$	20

Similar to Example 12, the ILU performs best, although not in terms of iteration steps. Again, using e *to improve the method is beneficial.*

We finally like to mention that the partitioning approach as indicated in Section 3 for nested dissection by nodes may also serve as parallelization approach prior to the incomplete factorization.

Example 14 *We consider again the model problem* $-\Delta u = f$ *and sketch in Figure 7 the parallel multilevel incomplete factorization in the cases* $p = 2, 4$ *and* $N = 100$ *grid points in each direction.*

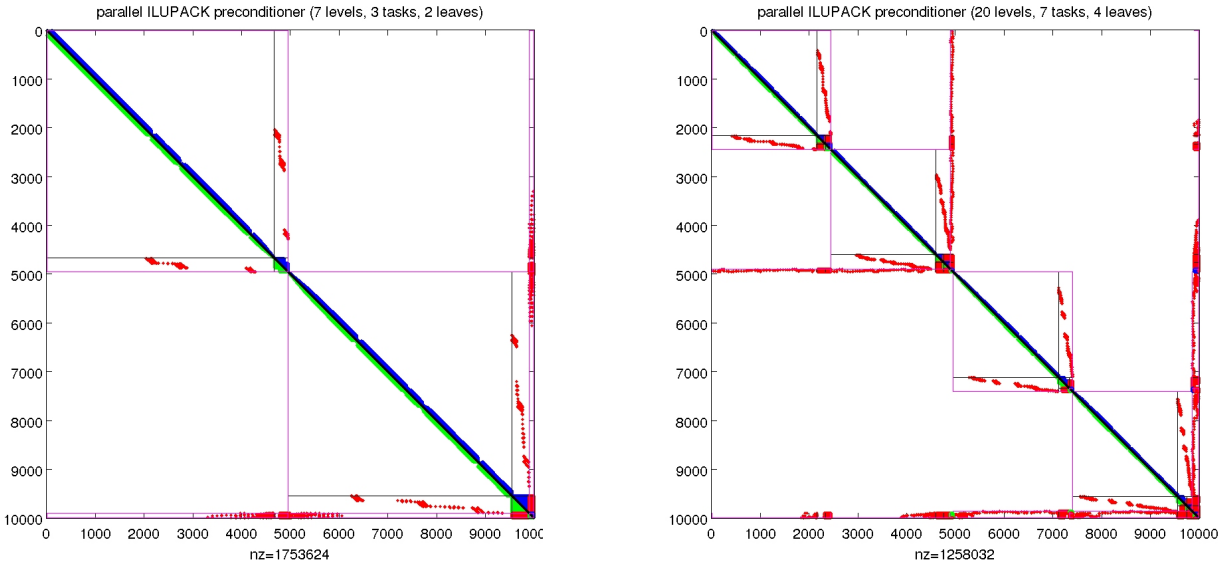


Figure 7: Parallel multilevel incomplete factorization, $p = 2$(left), $p = 4$(right)

6 Approximate Inversion Using Multilevel Approximation

In this final section we will illustrate how most of the aspects discussed in the previous sections can be usefully united for the approximate inversion of matrices. Functions of entries of inverses of matrices like all diagonal entries of a sparse matrix inverse or its trace arise in several important computational applications such as density functional theory [40], covariance matrix analysis in uncertainty quantification [7], simulation of quantum field theories [43], vehicle acoustics optimization [50], or when evaluating Green's functions in computational nanolelectronics [46]. Often enough, modern computational methods for matrix inversion are based on reordering or splitting the system into independent parts [11, 49], since in this case the (approximate) inverse triangular factors tend to be relatively sparse which simplifies their computation [44, 45, 65, 66]. Here we will use the following ingredients.

1. We will use the partitioning approach (7) from Section 2 for partitioning the systems. If some of the diagonal blocks were ill-conditioned, one could alternatively fall back to the splitting approach (5) and use a completion approach,

2. the multilevel incomplete factorization from Section 5 will be used as approximate factorization.

For the multilevel incomplete factorization we scale and reorder at each level the system using nested dissection. In principle, an approximate factor

$$L = \left(\begin{array}{ccc|c} L_{11} & & 0 & \\ & \ddots & & 0 \\ 0 & & L_{p-1,p-1} & \\ \hline L_{p1} & \cdots & L_{p,p-1} & L_{pp} \end{array} \right)$$

is easily inverted. This structure keeps the inverse factor sparse and can be applied recursively and is used in approximate inverse methods [18] and is part of several methods for exact inversion [45, 44, 65].

Example 15 *Consider the problem $-\Delta u = f$ on the unit square in two spatial dimensions with 5-point-star-stencil. The system will be reordered with nested dissection [38]. Figure 8 illustrates the incomplete Cholesky factor and its (approximate) inverse. Although its approximate inverse uses about ten times more memory it still approximately sparse.*

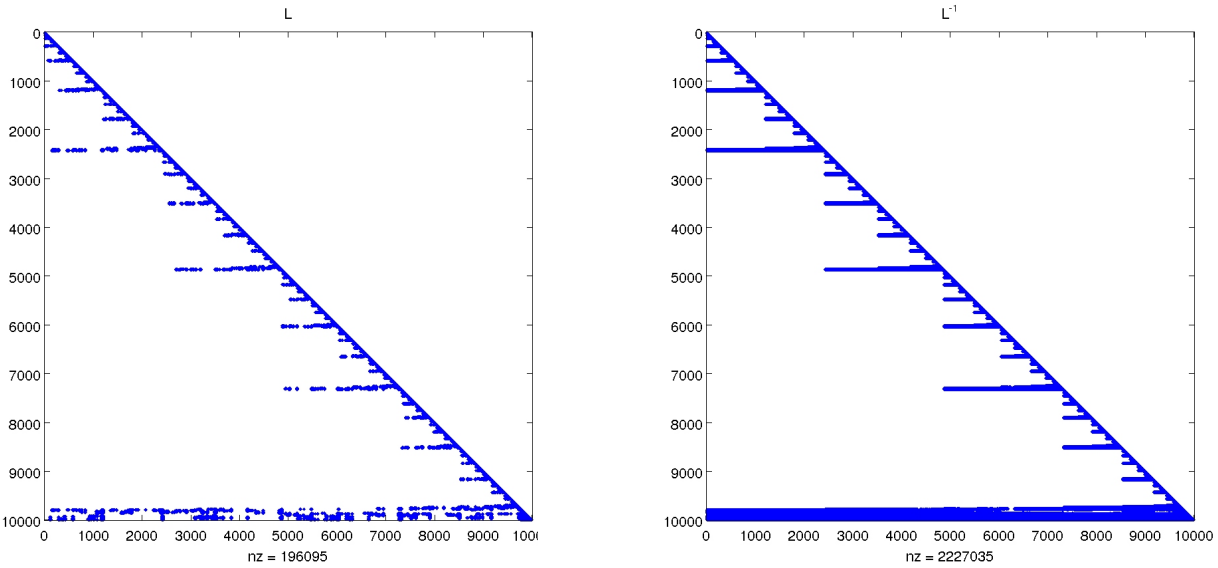


Figure 8: triangular factor and its (approximate) inverse after nested dissection reordering

Next we like to mention that multilevel incomplete factorizations can be rewritten as a single-level factorization. Consider the incomplete factorization (21) and suppose that $P_2^T D_2 S_C D_2 P_2 = L_C D_C L_C^T + \mathcal{E}_C$. One can easily verify that substitution into (21) leads to a factorization of the form $\hat{P}^T \hat{D} \hat{A} \hat{D} \hat{P} = \hat{L}\hat{L}^T + \hat{\mathcal{E}}$ with modified permutation matrix

$\hat{P}$, new diagonal matrix $\hat{D}$, lower triangular matrix $\hat{L}$ and some perturbation $\hat{\mathcal{E}}$. The triangular factors from Example 15 already refer to a multilevel factorization that was formally rewritten as a single-level factorization.

When inverting the triangular factorization (21) we already know that

$$A^{-1} \approx DP \left[\begin{pmatrix} I \\ 0 \end{pmatrix} \tilde{B}^{-1} \begin{pmatrix} 0 & I \end{pmatrix} + \begin{pmatrix} -L_B^{-T} L_E^T \\ I \end{pmatrix} S_C^{-1} \begin{pmatrix} -L_E L_B^{-1} & I \end{pmatrix} \right] P^T D,$$

where equality holds if $\mathcal{E} = 0$ and in particular selected entries such as the diagonal entries of A^{-1} are dominated by S_C^{-1} when $\|\tilde{B}^{-1}\|$ is well-bounded. Here again, as before, we have set $\tilde{B} = L_B D_B L_B^T$. One can compute the diagonal entries of the inverses separately from the sum [65]. Computing the diagonal entries $\mathcal{D}(\tilde{B}^{-1})$ of $\tilde{B}^{-1}$ is easily achieved because of the nested dissection partition and the multilevel approach. It is harder to compute the remaining Schur complement S_C^{-1} in general. But again in a multilevel setting, S_C is substituted until eventually only a system of small size is left over. If we construct the multilevel factorization such that $L_E L_B^{-1}$ is bounded [14], then the influence of the diagonal entries $\mathcal{D}(L_B^{-T} L_E^T S_C^{-1} L_E L_B^{-1})$ in the inversion of $\hat{A}$ remains on the same order as $\|S_C^{-1}\|$. To construct $\tilde{B}$ that is easy to invert and to keep $\|L_E L_B^{-1}\|$ bounded justifies to use a multilevel approach instead of a single level incomplete factorization.

Example 16 *We consider the linear operator A that is obtained from $-\Delta u = f$ on the unit square in two spatial dimensions using as before 5-point-difference stencil, Dirichlet boundary conditions and N grid points in each spatial direction. Here $\mathcal{D}(A^{-1})$ is explicitly known which simplifies numerical comparisons. We will use a multilevel incomplete factorization from [14] using different drop tolerances τ. Pivoting is introduced such that successively $\|L_B^{-1}\|$, $\|L_E L_B^{-1}\|$ are approximately kept below a given threshold κ; here we will choose $\kappa = 100$. For details of this strategy we refer to [14].*

N	τ	$\frac{\|\hat{A} - \hat{L} D \hat{L}^T\|}{\|\hat{A}\|}$	$\frac{\|\mathcal{D}(A^{-1} - \hat{P}\hat{D}\ L^{-T} D^{-1} L^{-1} \hat{D} \hat{P}^T)\|}{\|\mathcal{D}(A^{-1})\|}$	$\frac{\|\operatorname{trace}(A^{-1} - \hat{P}\hat{D}\ L^{-T} D^{-1} L^{-1} \hat{D} \hat{P}^T)\|}{\|\operatorname{trace}(A^{-1})\|}$
50	10^{-4}	$4.2 \cdot 10^{-5}$	$1.1 \cdot 10^{-4}$	$5.2 \cdot 10^{-6}$
100	10^{-4}	$1.8 \cdot 10^{-5}$	$1.9 \cdot 10^{-5}$	$2.6 \cdot 10^{-6}$
200	10^{-4}	$1.4 \cdot 10^{-5}$	$3.8 \cdot 10^{-5}$	$2.3 \cdot 10^{-6}$
50	10^{-5}	$2.6 \cdot 10^{-6}$	$5.5 \cdot 10^{-6}$	$1.2 \cdot 10^{-7}$
100	10^{-5}	$2.2 \cdot 10^{-6}$	$1.2 \cdot 10^{-6}$	$6.2 \cdot 10^{-8}$
200	10^{-5}	$3.2 \cdot 10^{-4}$	$1.6 \cdot 10^{-3}$	$1.8 \cdot 10^{-4}$
50	10^{-6}	$8.5 \cdot 10^{-16}$	$8.1 \cdot 10^{-15}$	$5.4 \cdot 10^{-16}$
100	10^{-6}	$2.1 \cdot 10^{-8}$	$1.8 \cdot 10^{-8}$	$3.1 \cdot 10^{-10}$
200	10^{-6}	$1.1 \cdot 10^{-5}$	$6.1 \cdot 10^{-5}$	$1.0 \cdot 10^{-5}$

The displayed norm here is always $\| \bullet \|_\infty$. We point out that the multilevel incomplete factorization is not yet fit for approximate inversion. For this reason we do not display the

computation time. We can see that the error with respect to the inverse is of the same order as the drop tolerance or at most one order greater which demonstrates the effectiveness of this approach.

Finally we mention that to turn the multilevel approach into an efficient method for approximate inversion, the approach would have to be modified to

$$\begin{pmatrix} W_B & 0 \\ W_E & I \end{pmatrix} \hat{A} \begin{pmatrix} W_B^T & W_E^T \\ 0 & I \end{pmatrix} = \begin{pmatrix} D_B & 0 \\ 0 & S_C \end{pmatrix} + \mathcal{E}$$

which refers to a multilevel approximate inverse-type approach generalizing the AINV method [9, 10]. This will be subject of future research and algorithms.

Conclusions

In this paper we have demonstrated that several Numerical Linear Algebra methods can be efficiently used in many recent preconditioning techniques and matrix inversion methods. They give deep information about the underlying approximation and help to improve these methods.

References

[1] E. Agullo, L. Giraud, A. Guermouche, A. Haidar, and J. Roman. Parallel algebraic domain decomposition solver for the solution of augmented systems. *Advances in Engineering Software*, 60-61:23–30, 2012.

[2] P. Amestoy, T. A. Davis, and I. S. Duff. An approximate minimum degree ordering algorithm. *SIAM J. Matrix Anal. Appl.*, 17(4):886–905, 1996.

[3] P. R. Amestoy, I. S. Duff, and J.-Y. L'Excellent. Multifrontal parallel distributed symmetric and unsymmetric solvers. *Comput. Methods in Appl. Mech. Engrg.*, 184:501–520, 2000.

[4] G. Ammar, P. Benner, and V. Mehrmann. A multishift algorithm for the numerical solution of algebraic Riccati equations. *Electr. Trans. Num. Anal.*, 1:33–48, 1993.

[5] O. Axelsson and P. Vassilevski. Algebraic multilevel preconditioning methods I. *Numer. Math.*, 56:157–177, 1989.

[6] O. Axelsson and P. Vassilevski. Algebraic multilevel preconditioning methods II. *SIAM J. Numer. Anal.*, 27:1569–1590, 1990.

[7] C. Bekas, A. Curioni, and I. Fedulova. Low-cost high performance uncertainty quantification. *Concurrency and Computation: Practice and Experience*, 24:908–920, 2011.

[8] M. Benzi, J. C. Haws, and M. Tůma. Preconditioning highly indefinite and nonsymmetric matrices. *SIAM J. Sci. Comput.*, 22(4):1333–1353, 2000.

[9] M. Benzi, C. D. Meyer, and M. Tůma. A sparse approximate inverse preconditioner for the conjugate gradient method. *SIAM J. Sci. Comput.*, 17:1135–1149, 1996.

[10] M. Benzi and M. Tůma. A sparse approximate inverse preconditioner for nonsymmetric linear systems. *SIAM J. Sci. Comput.*, 19(3):968–994, 1998.

[11] M. W. Berry and A. Sameh. Multiprocessor schemes for solving block tridiagonal linear systems. *The International Journal of Supercomputer Applications*, 1(3):37–57, 1988.

[12] M. Bollhöfer and V. Mehrmann. Algebraic multilevel methods and sparse approximate inverses. *SIAM J. Matrix Anal. Appl.*, 24(1):191–218, 2002.

[13] M. Bollhöfer and V. Mehrmann. Some convergence estimates for algebraic multilevel preconditioners. In V. Olshevsky, editor, *Fast Algorithms for Structured Matrices: Theory and Applications*, volume 323, pages 293–312. AMS, SIAM, 2003.

[14] M. Bollhöfer and Y. Saad. Multilevel preconditioners constructed from inverse–based ILUs. *SIAM J. Sci. Comput.*, 27(5):1627–1650, 2006.

[15] S. Bondeli. *Parallele Algorithmen zur Lösung tridiagonaler Gleichungssysteme*. Dissertationsschrift, ETH Zürich, Department Informatik, Institut f. Wissenschaftliches Rechnen, 1991.

[16] E. Botta and F. Wubs. Matrix renumbering ILU: an effective algebraic multilevel ILU-decomposition. *SIAM J. Matrix Anal. Appl.*, 20:1007–1026, 1999.

[17] M. Brezina, A. J. Cleary, R. D. Falgout, V. E. Henson, J. E. Jones, T. A. Manteuffel, S. F. McCormick, and J. W. Ruge. Algebraic multigrid based on element interpolation (AMGe). *SIAM J. Sci. Comput.*, 22:1570–1592, 2000.

[18] R. Bridson and W.-P. Tang. Ordering, anisotropy and factored sparse approximate inverses. *SIAM J. Sci. Comput.*, 21(3):867–882, 1999.

[19] A. Bunse-Gerstner and V. Mehrmann. A symplectic QR-like algorithm for the solution of the real algebraic Riccati equation. *IEEE Transactions on Automatic Control*, AC-31:1104–1113, 1986.

[20] R. Byers and V. Mehrmann. Symmetric updating of the solution of the algebraic Riccati equation. *Methods of Operations Research*, 54:117–125, 1986.

[21] T. Chartier, R. D. Falgout, V. E. Henson, J. Jones, T. Manteuffel, S. McCormick, J. Ruge, and P. S. Vassilevski. Spectral AMGe (ρAMGe). *SIAM J. Sci. Comput.*, 25:1–26, 2004.

[22] T. A. Davis and I. S. Duff. A combined unifrontal/multifrontal method for unsymmetric sparse matrices. *ACM Trans. Math. Software*, 25(1):1–19, 1999.

[23] J. W. Demmel, S. C. Eisenstat, J. R. Gilbert, X. S. Li, and J. W. H. Liu. A supernodal approach to sparse partial pivoting. *SIAM J. Matrix Anal. Appl.*, 20(3):720–755, 1999.

[24] I. S. Duff and J. Koster. The design and use of algorithms for permuting large entries to the diagonal of sparse matrices. *SIAM J. Matrix Anal. Appl.*, 20(4):889–901, 1999.

[25] I. S. Duff and J. Koster. On algorithms for permuting large entries to the diagonal of a sparse matrix. *SIAM J. Matrix Anal. Appl.*, 22(4):973–996, 2001.

[26] L. Elsner, C. He, and V. Mehrmann. Minimizing the condition number of a positive definite matrix by completion. *Numer. Math.*, 69:17–24, 1994.

[27] L. Elsner, C. He, and V. Mehrmann. Minimization of the norm, the norm of the inverse and the condition number of a matrix by completion. *Numer. Lin. Alg. w. Appl.*, 2(2):155–171, 1995.

[28] Y. A. Erlangga and R. Nabben. Multilevel projection-based nested Krylov iteration for boundary value problems. *SIAM J. Sci. Comput.*, 30(3):1572–1595, 2008.

[29] Y. A. Erlangga and R. Nabben. Algebraic multilevel Krylov methods. *SIAM J. Sci. Comput.*, 31(5):3417–3437, 2009.

[30] L. Giraud, A. Haidar, and Y. Saad. Sparse approximations of the Schur complement for parallel algebraic hybrid linear solvers in 3d. *Numerical Mathematics: Theory, Methods and Applications*, 3(3):276–294, 2010.

[31] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, fourth edition, 2012.

[32] A. Greenbaum. *Iterative Methods for Solving Linear Systems*. Frontiers in Applied Mathematics. SIAM Publications, 1997.

[33] A. Gupta, M. Joshi, and V. Kumar. WSSMP: A high-performance serial and parallel symmetric sparse linear solver. PARA'98 Workshop on Applied Parallel Computing in Large Scale Scientific and Industrial Problems, Umea, Sweden, June 1998.

[34] W. Hackbusch. *Multigrid Methods and Applications*. Springer-Verlag, 1985.

[35] V. E. Henson and P. S. Vassilevski. Element-free AMGe: General algorithms for computing interpolation weights in AMG. *SIAM J. Sci. Comput.*, 23(2):629–650, 2001.

[36] M. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Research Nat. Bur. Standards*, 49:409–436, 1952.

[37] G. Karypis and V. Kumar. *MeTis: Unstrctured Graph Partitioning and Sparse Matrix Ordering System, Version 2.0*, August 1995.

[38] G. Karypis and V. Kumar. A coarse-grain parallel formulation of multilevel k-way graph-partitioning algorithm. In *Proc. 8th SIAM Conference on Parallel Processing for Scientific Computing*. SIAM Publications, 1997.

[39] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20(1):359–392, 1998.

[40] W. Kohn, L. Sham, et al. Self-consistent equations including exchange and correlation effects. *Phys. Rev*, 140(4A):A1133–A1138, 1965.

[41] P. Kunkel and V. Mehrmann. Numerical solution of Riccati differential algebraic equations. *Linear Algebra Appl.*, 137/138:39–66, 1990.

[42] P. Lancaster and L. Rodman. *The Algebraic Riccati Equation*. Oxford University Press, 1995.

[43] D. Lee and I. Ipsen. Zone determinant expansions for nuclear lattice simulations. *Physical review C*, 68:064003–1–064003–8, 2003.

[44] S. Li, S. Ahmed, G. Klimeck, and E. Darve. Computing entries of the inverse of a sparse matrix using the FIND algorithm. *J. Comput. Phys.*, 227(22):9408–9427, 2008.

[45] L. Lin, C. Yang, J. C. Meza, J. Lu, L. Ying, and W. E. SelInv – an algorithm for selected inversion of a sparse symmetric matrix. *ACM Transactions on Mathematical Software*, 37(4):40:1–40:19, 2011.

[46] M. Luisier, T. Boykin, G. Klimeck, and W. Fichtner. Atomistic nanoelectronic device engineering with sustained performances up to 1.44 pflop/s. In *High Performance Computing, Networking, Storage and Analysis (SC), 2011 International Conference for*, pages 1–11. IEEE, 2011.

[47] M. Manguoglu, A. H. Sameh, and O. Schenk. PSPIKE: A parallel hybrid sparse linear system solver. In *Euro-Par*, pages 797–808, 2009.

[48] V. Mehrmann. Divide and conquer algorithms for tridiagonal linear systems. In W. Hackbusch, editor, *Parallel Algorithms for PDEs, Proceedings of the 6th GAMM-Seminar, Kiel*, Notes on Numerical Fluid Mechanics, pages 188–199. Vieweg, Braunschweig, Germany, 1990.

[49] V. Mehrmann. Divide & conquer methods for block tridiagonal systems. *Parallel Comput.*, 19:257–279, 1993.

[50] V. Mehrmann and C. Schröder. Nonlinear eigenvalue and frequency response problems in industrial practice. *J. Math. in Industry*, 1:7, 2011.

[51] V. Mehrmann and E. Tan. Defect correction methods for the solution of algebraic Riccati equations. *IEEE Transactions on Automatic Control*, AC-33:695–698, 1988.

[52] A. C. Muresan and Y. Notay. Analysis of aggregation-based multigrid. *SIAM J. Sci. Comput.*, 30(2):1082–1103, 2008.

[53] Y. Notay. Using approximate inverses in algebraic multigrid methods. *Numer. Math.*, 80:397–417, 1998.

[54] Y. Notay. An aggregation-based algebraic multigrid method. *Electr. Trans. Num. Anal.*, 37:123–146, 2010.

[55] Y. Notay and P. S. Vassilevski. Recursive Krylov-based multigrid cycles. *Numer. Lin. Alg. w. Appl.*, 15:473–487, 2008.

[56] A. Reusken. A multigrid method based on incomplete Gaussian elimination. Preprint 95-13, Eindhoven University of Technology, Dept. of Mathematics and Computer Science, October 1995.

[57] A. Reusken. A multigrid method based on incomplete Gaussian elimination. *Numer. Lin. Alg. w. Appl.*, 3:369–390, 1996.

[58] J. Ruge and K. Stüben. Algebraic multigrid. In S. McCormick, editor, *Multigrid Methods*, pages 73–130. SIAM Publications, 1987.

[59] Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM Publications, second edition, 2003.

[60] Y. Saad. Multilevel ILU with reorderings for diagonal dominance. *SIAM J. Sci. Comput.*, 27:1032–1057, 2005.

[61] Y. Saad and B. J. Suchomel. ARMS: An algebraic recursive multilevel solver for general sparse linear systems. *Numer. Lin. Alg. w. Appl.*, 9:359–378, 2002.

[62] O. Schenk and K. Gärtner. Solving unsymmetric sparse systems of linear equations with PARDISO. *J. of Future Generation Computer Systems*, 20(3):475–487, 2004.

[63] B. Smith, P. Bjorstad, and W. Gropp. *Domain Decomposition, Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge University Press, Cambridge, 1996.

[64] K. Stüben. *An Introduction to Algebraic Multigrid*, pages 413–532. In Trottenberg et al. [67], 2001. Appendix A.

[65] J. M. Tang and Y. Saad. Domain-decomposition-type methods for computing the diagonal of a matrix inverse. *SIAM J. Sci. Comput.*, 33(5):2823–2847, 2011.

[66] J. M. Tang and Y. Saad. A probing method for computing the diagonal of a matrix inverse. *Numerical Linear Algebra with Applications*, 19(3):485–501, 2012.

[67] U. Trottenberg, C. W. Oosterlee, and A. Schüller. *Multigrid*. Academic Press, London, 2001.

[68] A. Van der Ploeg, E. Botta, and F. Wubs. Nested grids ILU–decomposition (NGILU). *J. Comput. Appl. Math.*, 66:515–526, 1996.

[69] H. A. Van der Vorst. Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.*, 13(2):631–644, 1992.

[70] P. Vanek, J. Mandel, and M.Brezina. Algebraic multigrid by smoothed aggregation for second order and fourth order elliptic problems. *Computing*, 56:179–196, 1996.

[71] P. S. Vassilevski. *Multilevel Block Factorization Preconditioners*. Springer, 2008.

[72] C. Wagner and G. Wittum. Adaptive filtering. *Numer. Math.*, 78:305–328, 1997.

[73] C. Wagner and G. Wittum. Filtering decompositions with respect to adaptive test vectors. In W. Hackbusch and G. Wittum, editors, *Multigrid Methods V*, volume 3 of *Lecture Notes in Computational Science and Engineering*, pages 320–334. Springer-Verlag, 1998.

[74] J. Xu and J. Zou. Some nonoverlapping domain decomposition methods. *SIAM Review*, 40(4):857–914, 1998.