Technische Universität Chemnitz

Sonderforschungsbereich 393

Numerische Simulation auf massiv parallelen Rechnern

Matthias Bollhöfer Volker Mehrmann[1]

A new approach to algebraic multilevel methods based on sparse approximate inverses

Preprint SFB393/99-22

Preprint-Reihe des Chemnitzer SFB 393

SFB393/99-22 **August 1999**

Contents

Author's addresses:

Matthias Bollhöfer
Volker Mehrmann
TU Chemnitz
Fakultät für Mathematik
D-09107 Chemnitz

bolle@mathematik.tu-chemnitz.de
mehrmann@mathematik.tu-chemnitz.de
http://www.tu-chemnitz.de/∼bolle/
http://www.tu-chemnitz.de/∼mehrmann/

`http://www.tu-chemnitz.de/sfb393/`

Abstract

In this paper we introduce a new approach to algebraic multilevel methods and their use as preconditioners in iterative methods for the solution of symmetric positive definite linear systems. The multilevel process and in particular the coarsening process is based on the construction of sparse approximate inverses and their augmentation with corrections of smaller size. We present comparisons of condition estimates and numerical results.

Keywords: sparse approximate inverse, large sparse matrices, algebraic multilevel method.

AMS subject classification: 65F05, 65F10, 65F50, 65Y05, 93B40.

1 Introduction

For the solution of large sparse linear systems of the form

$$Ax = b,\ A \in \mathrm{GL}\,(n, \mathbb{R}), b \in \mathbb{R}^n \tag{1}$$

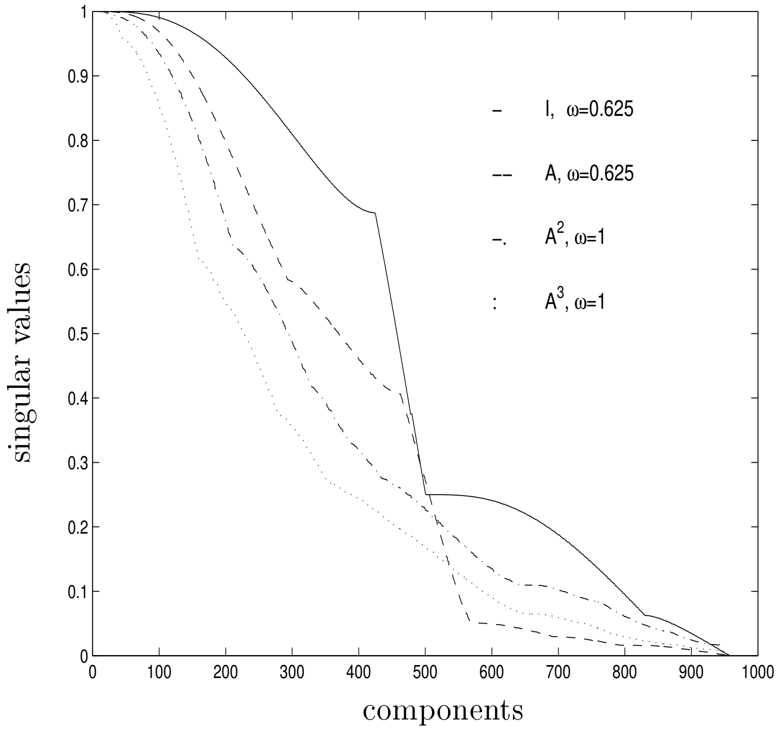
sparse approximate inverse approximations [20, 19, 7, 15, 5, 21] have become popular as preconditioners for Krylov–subspace [10, 25, 14] techniques. The main idea is to construct a matrix B that approximates A^{-1} but B is still sparse. Several techniques have been developed, for example minimizing the norm of $\|AB - I\|$ subject to some prescribed pattern [19, 7, 15] or biconjugate techniques $ZAW \approx D$, where $Z, W^\top$ are lower triangular matrices and D is diagonal [5].

1.1 Sparse approximate inverses are smoothing

While sparse approximate inverse matrices are quite powerful as preconditioners for a large class of matrices, there are cases when the sparse approximate inverse needs a large number of nonzero entries to become a suitable approximation to the inverse of A. An observation, which one can make quite often when using sparse approximate inverses based on norm–minimizing techniques, is that many singular values [13] of the residual matrix $E = I - AB$ are quite small while a small number of singular values stay big. This effect also occurs, when allowing more fill–in for B.

Example 1 *Consider the symmetric positive definite matrix LANPRO/NOS2 from the Harwell–Boeing collection [9] and apply the sparse approximate inverse suggested in [20, 19] with sparsity pattern as* A^k, $k = 0, 1, 2, 3$. *Since this approach explicitly constructs approximate inverses* L *to the Cholesky factors* $U^\top U = A$ *of* A *we set* $E = I - \omega L^\top AL$ *for a suitably chosen* ω. *Again* E *is our residual, adapted to the symmetric positive case. The parameter* ω *is chosen such that as many eigenvalues of* $L^\top AL$ *as possible are in a vicinity of* 1. *In Figure 1 we display the singular values of* E *in decreasing order. We observe that most of the singular values tend to* 0 *while a few singular values stay close to* 1 *even when increasing the number of nonzeros for* L.

Figure 1: **Singular values of the residual matrix E**



The observation that many singular values are small but some stay large can be interpreted that B approximates A on a subspace of large size while there is almost no approximation on the complementary subspace. In the numerical treatment of partial differential equations this effect is typically referred as smoothing property [16].

Under the hypothesis that many singular values are small and some stay close to 1 it follows that $E = I - AB = UV^\top + F$ for matrices $U, V^\top \in \mathbb{R}^{n,p}$ with $p \ll n$ and $\|F\| \ll 1$, i.e., the residual can be approximated by a low rank matrix. Figure 1 also shows that one can not expect that p is independent of the dimension n of A. More realistic is the assumption, that $p \leqslant cn$ with $c < 1$, e.g. $c = 0.5$. But in this case it is important to have a sparse representation of the low rank part. From this point of view to take the singular vectors of E with respect to its large singular values may not be efficient. This does not necessary mean that U, V have to be sparse. E.g. $UV^\top = PZ^{-1}Q$ for sparse matrices P, Q, Z may also be a sparse representation while their product need not to be sparse at all.

1.2 Sparse approximation of the invariant subspace of E associated with the large singular values

The assumption that B is a good approximation to A^{-1} on a sufficiently large subspace of $E = I - AB$ implies that we will have a clustering of most of the small singular values. Only few of the singular values will stay close to 1. In this sense B has the properties of a good smoother in the sense of multigrid methods for elliptic partial differential equations. Since only few of the singular values are close to one, we may write E as

$$E = UV^\top + F \equiv E_0 + F, \tag{2}$$

with E_0 of rank p and $\|F\| \leqslant \eta < 1$ and we have that the entries of E_0 only slightly differ from those of E. So we may expect that taking an appropriate selection of columns of E will be a good choice for augmenting AB by $UV^\top$. This is underscored by the following lemma.

Lemma 2 *Let $E \in \mathbb{R}^{n,n}$ and let $E = [U_1, U_2]\,\mathrm{diag}\,(\Sigma_1, \Sigma_2)[V_1, V_2]^\top$ be the singular value decomposition of E with U_1, V_1 having k columns and $\|\Sigma_2\|_2 \leqslant \varepsilon$. Then there exist a permutation matrix $\Pi = [\Pi_1, \Pi_2]$ with analogous partitioning such that*

$$\inf_{Z \in \mathbb{R}^{k,k}} \|U_1 Z - E\Pi_1\| \leqslant \varepsilon. \tag{3}$$

Proof:
Introduce

$$E_0 := U_1 \Sigma_1 V_1^\top.$$

Using the QR decomposition with column pivoting we can construct a $QR\Pi$ decomposition [13] of E_0 such that

$$E_0 \Pi = QR,$$

where $Q, R^\top \in \mathbb{R}^{n,k}$ $Q^\top Q = I$, $R = [R_1, R_2]$. Here $R_1 \in \mathbb{R}^{k,k}$ is upper triangular and $\Pi = [\Pi_1, \Pi_2]$ is a permutation matrix with Π_1 having k columns. It immediately follows that $E_0 \Pi_1 = QR_1$ and thus there exists a nonsingular $k \times k$ matrix Z such that $E_0 \Pi_1 = QR_1 = U_1 Z$. But then we have

$$\|E\Pi_1 - U_1 Z\|_2 = \|(E - E_0)\Pi_1\|_2 \leqslant \varepsilon.$$

$\square$

Note that Lemma 2 applied to $E^\top$ instead of E we can analogously approximate V_1 by suitably chosen rows of E. Using Lemma 2 we can determine subspaces that consist of suitably chosen columns or rows of E, which are close to the subspaces U_1, V_1 of E associated with the large singular values of E. But even if the distance between the exact invariant subspaces associated with the large singular values of E and the set of columns/rows of E is small, this does not mean that the updated preconditioner is improved. We have to update in a particular way. Recall that we wish to solve a linear system $Ax = b$ with A or with its preconditioned variant AB. We cannot just add $UV^\top$ from (2) to AB like $AB + UV^\top$ but we have to modify AB by $AB(I + (AB)^{-1} UV^\top)$. A small error between the exact subspace U and a set of columns of E may blow up the norm of $UV^\top$. Instead of simply using a QR–decomposition of E like it was essentially done in the proof of Lemma 2 we need a decomposition of the form $E = (AB)QR + F$ with $\|F\| \leqslant \eta$ to keep the error $AB(I + UV^\top) - I$ small. But if $E = I - AB$ has a decomposition of the form

$$E = (AB)QR + F, \tag{4}$$

where $Q, R^\top \in \mathbb{R}^{n,p}$, then

$$AB(I + QR) = I - F \tag{5}$$

In this paper we discuss the theoretical background for this problem i.e., preconditioners of this form, and show that they are closely related to algebraic multilevel methods. We derive two types (multiplicative and addtive) algebraic multilevel preconditioners in Section 2.

The approximation properties of the correction term $I + QR$ in (5) for the two multilevel schemes are studied in detail in Section 3.

In order to generate from this idea an efficient linear system solver, in (4) we have to find a sparse representation $QR = PZ^{-1}\hat{P}^\top$ with sparse matrices $P, \hat{P}, Z$ and we also have to find an efficient method to determine the desired columns of E.
For these two problems we describe several (partially heuristic) procedures. In principle, by Lemma 2 and (4), we could use a QR–like decomposition of E. But this has to be done in an efficient way and at the same time the memory needed for the decomposition has to be kept small. The key point in such a QR–like decomposition will be the corresponding pivoting strategy, since we do not want to select too many columns. This can be viewed as the coarsening process of the multilevel scheme and is discussed in Section 4. Finally in Section 5 we present several numerical examples that demonstrate the properties of this new approach and also indicate the effectiveness of the heuristics that have been used.

In the sequel for symmetric matrices A, B we will use the notation $A \geqslant B$, if $A - B$ has nonnegative eigenvalues. We also identify for a matrix $V \in \mathbb{R}^{n,p}$ the matrix V with the space $\{Vx : x \in \mathbb{R}^p\}$ that is spanned by its columns.

2 Two Multilevel Preconditioners

In this section we present two approaches of multilevel methods for symmetric positive definite systems. Let $L, A \in \mathbb{R}^{n,n}$ be nonsingular, where A is symmetric positive definite. Suppose that $LL^\top$ is a symmetric positive definite matrix in factored from that is an approximation to A^{-1}. Here it is understood that L is a sparse matrix for which linear systems are solved with low complexity on the available computer architecture. Suppose now that the approximation of A^{-1} by $LL^\top$ is not satisfactory, e.g., the condition number of $L^\top AL$ is not small enough to get good convergence in the conjugate gradient method, and we wish to improve the approximation (i.e., the preconditioning properties). To do this we like to determine a matrix of the form

$$M^{(1)} = LL^\top + PZ^{-1}P^\top \tag{6}$$

with $P \in \mathbb{R}^{n,p}$, $Z \in \mathbb{R}^{p,p}$ nonsingular and sparse and furthermore, $p \leqslant cn$, $c \ll 1$ so that $M^{(1)}$ is a better approximation to A^{-1} than $LL^\top$. In (6) the sparse approximate inverse $LL^\top$ is augmented as

$$LL^\top \longrightarrow LL^\top + \boxed{\begin{matrix} n \\ \times \\ p \end{matrix}}\ \boxed{}\ \boxed{\quad p \times n \quad}$$

Note that we do not assume that P has low rank, since we do not assume that $p \ll n$ but only that $p \leqslant cn$ with $c < 1$, e.g. $c = 0.5$ as already pointed out in the introduction. It is very important, however, that P, Z are sparse. But observe, that since Z^{-1} is used in (6), $PZ^{-1}P^\top$ need not to be sparse at all!
The particular form (6) is chosen close to the form of an algebraic two–level method, where multiplication with $P, P^\top$ corresponds to the mapping between fine and coarse grid and Z represents the coarse grid system. Note further that using the representation $LL^\top + PZ^{-1}P^\top$ as a preconditioner for A, only a system with Z has to be solved. This is a good point to introduce the terminology coarse and fine grid as well as coarse grid nodes and fine grid nodes. As shown in Lemma 2, skilfully chosen columns/rows of the residual matrix $E = I - L^\top AL$ can be used to approximate the invariant subspace of E associated with its large eigenvalues. As we will see below, precisely this invariant subspace has to be approximated by P. In the sense of the underlying undirected graph of E we refer to those nodes as coarse grid nodes whose columns/rows of E will be used to approximate the invariant subspace of E associated with its largest eigenvalues while the remaining nodes are called fine grid nodes. The process of detecting a suitable set of coarse grid nodes will be called coarsening process. Once we have selected certain nodes as coarse grid nodes they are in a natural way embedded in the initial graph. In addition the subset of coarse grid nodes has its own related graph. As we will show below, $Z = P^\top AP$ will be (almost) optimal. But in this case the graph of Z gives us a natural graph associated with the coarse grid nodes. We will call it coarse grid as an analogy to partial differential equations.

Recalling the well-known techniques of constructing good preconditioners for the conjugate gradient method applied to symmetric positive definite systems, e.g. [13, 17, 26], we should choose P and Z such that

$$\mu A^{-1} \leqslant M^{(1)} \leqslant \mu\kappa^{(1)} A^{-1} \tag{7}$$

with κ as small as possible and $\mu > 0$. Clearly $\kappa \geqslant 1$ is the condition number of $M^{(1)}A$, i.e., the ratio of the largest by the smallest eigenvalue of $M^{(1)}A$ and thus $\kappa^{(1)} = 1$ would be optimal. The importance of the condition number is justified from the well–known error bounds for the conjugate gradient method [13] with preconditioner $M^{(1)}$. The iterate $x^{(k)}$ in the k–th step of the unpreconditioned cg–method satisfies [13]

$$\sqrt{(x^{(k)} - x)^\top A(x^{(k)} - x)} \leqslant 2\left(\frac{\sqrt{\kappa^{(1)}} - 1}{\sqrt{\kappa^{(1)}} + 1}\right)^k \sqrt{(x^{(0)} - x)^\top A(x^{(0)} - x)}.$$

Clearly in the preconditioned case one has to replace A by $(M^{(1)})^{1/2}A(M^{(1)})^{1/2}$. We will discuss the construction of P, Z with minimal $\kappa^{(1)}$ in the next subsection.

For discretized elliptic partial differential equation one can construct optimal preconditioners using multigrid methods [17]. In order to obtain a similar preconditioner augmented with a suitably chosen coarse grid correction, consider the use of $LL^\top$ in a linear iteration scheme with initial guess $x^{(0)} \in \mathbb{R}^n$. The iteration scheme [29] for the solution of $Ax = b$ has the form

$$x^{(k+1)} = x^{(k)} + LL^\top(b - Ax^{(k)}), k = 0, 1, 2, \ldots$$

The error propagation matrix $I - LL^\top A$ satisfies $x - x^{(k+1)} = (I - LL^\top A)(x - x^{(k)})$. In multilevel techniques [16] one could use this iteration for pre and post smoothing and in addition one has to add a coarse grid correction. In terms of the error propagation matrix this means that instead of $I - LL^\top A$ we have $(I - LL^\top A)(I - PZ^{-1}P^\top A)(I - LL^\top A)$ as error propagation matrix. Clearly this product can be rewritten as $I - M^{(2)}A$ with

$$M^{(2)} = 2LL^\top - LL^\top ALL^\top + (I - LL^\top A)PZ^{-1}P^\top(I - ALL^\top). \tag{8}$$

Note that when applying $M^{(2)}$ to a vector x, A and $LL^\top$ have only to be applied twice. So the application of this operator is less expensive than it looks. Again we are interested in choosing P, Z such that .

$$\mu A^{-1} \leqslant M^{(2)} \leqslant \mu\kappa^{(2)} A^{-1} \tag{9}$$

with $\kappa^{(2)}$ as small as possible.

Now we discuss elementary approximation properties of $M^{(1)}, M^{(2)}$. The first step will be the construction of optimal P, Z for given A, L. The construction and analysis will be based on the spectral decomposition

$$E := I - L^\top AL = V\Lambda V^\top, \tag{10}$$

where $\Lambda = \operatorname{diag}(\lambda_1 \ldots, \lambda_n)$ with diagonal elements in decreasing order and $V = [v_1, \ldots, v_n]$ is orthogonal.

Lemma 3 *Let $A, L \in \mathbb{R}^{n,n}$ with A symmetric positive definite, L nonsingular, $E = I - L^\top AL$ positive semidefinite and let $p < n$.*

1. *For $P \in \mathbb{R}^{n,p}$, $\hat{Z} \in \mathbb{R}^{p,p}$ defined via*

$$P := L[q_1, \ldots, q_p], \ Z := P^\top AP \left(I - P^\top AP\right)^{-1}, \tag{11}$$

   we obtain the minimal $\kappa^{(1)}$ in (7).
   In this case we have $\mu = 1 - \lambda_{p+1}, \kappa^{(1)} = \frac{1-\lambda_n}{1-\lambda_{p+1}}$.

2. *For P from (11) and $\hat{Z} \in \mathbb{R}^{p,p}$ defined via*

$$\hat{Z} = P^\top AP \tag{12}$$

   we have

$$\gamma M^{(1)} \leqslant LL^\top + P\hat{Z}^{-1}P^\top \leqslant \Gamma M^{(1)}, \tag{13}$$

   where $\gamma = 2 - \lambda_1 \geqslant 1, \Gamma = 2 - \lambda_p \leqslant 2$.

3. *Matrices P from (11) and $\hat{Z}$ from (12) yield the minimal $\kappa^{(2)}$ in (9) with $\mu = 1 - \lambda_{p+1}^2, \kappa^{(2)} = \frac{1-\lambda_n^2}{1-\lambda_{p+1}^2}$.*

Proof:

1. For P, Z as in (11) we have

$$Z = (I - E)E^{-1} = (I - \text{ diag } (\lambda_1, \ldots, \lambda_p)) \text{ diag } (\lambda_1, \ldots, \lambda_p)^{-1} \equiv (I - \Lambda_p)\Lambda_p^{-1}.$$

Setting $V_p = [v_1, \ldots, v_p]$, (7) is equivalent to

$$\mu(I - E)^{-1} \leqslant I + V_p \Lambda_p (I - \Lambda_p)^{-1} V_p^\top \leqslant \mu\kappa^{(1)} (I - E)^{-1}. \tag{14}$$

By transforming with $V^\top$ from the left and V from the right we obtain an inequality for diagonal matrices as

$$\mu \begin{bmatrix} \frac{1}{1-\lambda_1} & & \\ & \ddots & \\ & & \frac{1}{1-\lambda_n} \end{bmatrix} \leqslant \begin{bmatrix} \frac{1}{1-\lambda_1} & & & & & \\ & \ddots & & & & \\ & & \frac{1}{1-\lambda_p} & & & \\ & & & 1 & & \\ & & & & \ddots & \\ & & & & & 1 \end{bmatrix} \leqslant \mu\kappa^{(1)} \begin{bmatrix} \frac{1}{1-\lambda_1} & & \\ & \ddots & \\ & & \frac{1}{1-\lambda_n} \end{bmatrix}$$

and for $\mu = 1 - \lambda_{p+1}, \kappa^{(1)} = \frac{1-\lambda_n}{1-\lambda_{p+1}}$ the inequalities hold. The optimality of $\kappa^{(1)}$ in (14) follows directly from the Courant–Fischer min–max characterization [13] which implies that $\mu \leqslant 1 - \lambda_{p+1}$ and $\mu\kappa \geqslant 1 - \lambda_n$. Thus the choice of $\kappa^{(1)}$ is optimal and with P, Z we obtain the optimal $\kappa^{(1)}$.

2. For $\hat{Z}$ as in (12), we note that we have $\lambda_i \in [0, 1)$ and therefore inequalities (13) immediately follow.

3. For $M^{(2)}$ we proceed analogously. The desired inequality has the form

$$\mu(I - E)^{-1} \leqslant I + E + EV_p(I - \Lambda_p)^{-1} V_p^\top E \leqslant \mu\kappa^{(2)} (I - E)^{-1}. \tag{15}$$

Transforming with V from the right and its transpose from the left, $I + E + EV_p(I - \Lambda_p)^{-1} V_p^\top E$ changes to

$$\begin{bmatrix} \frac{1}{1-\lambda_1} & & & & & \\ & \ddots & & & & \\ & & \frac{1}{1-\lambda_p} & & & \\ & & & 1 + \lambda_{p+1} & & \\ & & & & \ddots & \\ & & & & & 1 + \lambda_n \end{bmatrix}$$

and the optimal choices are clearly $\mu = 1 - \lambda_{p+1}^2$ and $\mu\kappa = 1 - \lambda_n^2$.

$\square$

A similar result for $M^{(1)}$ was obtained in [3]. Note that with the optimal choice $M^{(1)}$ can be read as approximation to A^{-1} of *first order*, since $\kappa^{(1)} \approx 1/(1-\lambda_{p+1}^1)$, while $M^{(2)}$ is an approximation of *second order*, since $\kappa^{(2)} \approx 1/(1-\lambda_{p+1}^2)$.
Lemma 3 shows how the optimal choices for P, Z may be computed. But in practice we usually cannot determine these optimal choices, since the spectral decomposition is not available and even if it were available, then it would be very expensive to apply, since the matrix P would be a full matrix. Instead we would like to determine P, Z (or $P, \hat{Z}$) that are inexpensive to apply and still produce good approximation properties in $M^{(1)}$ ($M^{(2)}$). By the results of Lemma 3 it seems natural to set $Z = P^\top AP$ or to choose Z such that

$$\gamma Z \leqslant P^\top AP \leqslant \Gamma Z.$$

An inequality of this form is also useful if we intend to recursively repeat the technique in a multilevel way. To do this we replace in

$$LL^\top + P(P^\top AP)^{-1}P^\top \tag{16}$$

the term $(P^\top AP)^{-1}$ by an additive approximation $L_1L_1^\top + P_1(P_1^\top P^\top APP_1)^{-1}P_1^\top$. For the construction of $M^{(2)}$ the procedure is analogous. Recursively applied this idea leads to the following algebraic multilevel schemes.

Definition 4 *Let $A \in \mathbb{R}^{n,n}$ be symmetric positive definite and $n = n_l > n_{l-1} > \cdots > n_0 > 0$. For chosen full rank matrices $P_k \in \mathbb{R}^{n_k,n_{k-1}}$, $k = l, l-1, \ldots, 1$. define A_k via*

$$A_k = \begin{cases} A & k = l \\ P_{k+1}^\top A_{k+1} P_{k+1} & k = l-1, l-2, \ldots, 1 \end{cases}$$

Choose nonsingular $L_k \in \mathbb{R}^{n_k,n_k}$ such that $L_kL_k^\top \approx A_k^{-1}$, $k = 0, \ldots, l$. Then the multilevel sparse approximate preconditioners $M_l^{(1)}, M_l^{(2)}$ are recursively defined via

$$M_k^{(1)} = \begin{cases} A_0^{-1} & k = 0 \\ L_kL_k^\top + P_kM_{k-1}^{(1)}P_k^\top & k = 1, \ldots, l \end{cases}$$

$$M_k^{(2)} = \begin{cases} A_0^{-1} & k = 0 \\ L_k(2I - L_k^\top A_kL_k)L_k^\top + (I - L_kL_k^\top A_k)P_kM_{k-1}^{(2)}P_k^\top(I - A_kL_kL_k^\top) & k = 1, 2, \ldots, l \end{cases}$$

For $l = 1$ we obviously obtain the operators $M^{(1)}$ and $M^{(2)}$ from the previous subsection.

If we exactly decompose $A_0^{-1} = L_0L_0^\top$, e.g. by Cholesky decomposition and set $\Pi_k = P_lP_{l-1}\cdots P_{k+1}$ then we can obviously rewrite $M_l^{(1)}$ as

$$M_l^{(1)} = \sum_{k=0}^{l} \Pi_kL_kL_k^\top\Pi_k^\top. \tag{17}$$

For $M_l^{(2)}$ one obtains that

$$I - M_l^{(2)} A = (I - \Pi_l L_l L_l^\top \Pi_l^\top A) \cdots (I - \Pi_0 L_0 L_0^\top \Pi_0^\top A) \cdots (I - \Pi_l L_l L_l^\top \Pi_l^\top A). \qquad (18)$$

We conclude from (17),(18) that $M_l^{(1)}$ can be read as *additive multilevel method*, since all the projections Π_k are formally performed simultaneously while $M_l^{(2)}$ can be read as *multiplicative multilevel method*, since the projections Π_k are performed successively. In the sequel we also refer to $M_l^{(1)}$ as the additive algebraic multilevel scheme and refer to $M_l^{(2)}$ as the multiplicative algebraic multilevel scheme.

Operator $M_l^{(2)}$ is immediately derived from V–cycle methods in partial differential equations. For operator $M_l^{(1)}$ a special case will be the case when $L_k L_k^\top = \frac{1}{\alpha_k} I$ is a multiple of the identity. In this case $E_k = I - \alpha_k A_k$ and choosing some columns of E_k can be expressed as applying a permutation $\Phi_k \in \mathbb{R}^{n_k, n_{k-1}}$ to E_k, i.e. $P_k = (I - \alpha_k A_k)\Phi_k$. In this case $M_l^{(1)}$ reduces to

$$\begin{aligned} M_l^{(1)} &= \frac{1}{\alpha_l}(I + \alpha_l P_l M_{l-1} P_l^\top) \\ &= \frac{1}{\alpha_l}\left(I + \frac{\alpha_l}{\alpha_{l-1}} P_l \left(I + \alpha_{l-1} P_{l-1} M_{l-2} P_{l-1}^\top\right) P_l^\top\right) \\ &= \cdots \end{aligned}$$

For this type of operator in [18] optimal choices for α_k have been discussed according to a wisely a priori chosen permutation matrix Φ_k. This kind of operator has also been studied in [2, 3]. We refer to [3] for a detailed analysis for this kind of operator.

3 Approximation Properties

In this subsection we discuss the approximation properties of $M^{(1)}, M^{(2)}$ for the case $l = 1$ from (6),(8) and later for arbitrary $l \geqslant 1$ as in Definition 4.

The question is, for given Z, P, what can be said about the approximation properties of $M^{(1)}, M^{(2)}$ in (7),(9) compared with the optimal choice from Lemma 3. To get comparison theorems to the optimal P and Z from Lemma 3 we use of the following theorem.

Theorem 5 ([17]) *Given $M \in \mathbb{R}^{n,n}$ symmetric positive definite, $P_k \in \mathbb{R}^{n,n_k}$ with* $\operatorname{rank} P_k = n_k$ *for $k = 1, \ldots, l$ and* $\operatorname{rank}[P_1, \ldots, P_l] = n$. *Let $B_k \in \mathbb{R}^{n_k,n_k}$ be symmetric positive definite. Let*

$$M_S^{-1} := \sum_{k=1}^{l} P_k B_k^{-1} P_k^\top. \qquad (19)$$

If $c > 0$ is a constant, such that for any $x \in \mathbb{R}^n$ there exists a decomposition $x = \sum_{k=1}^{l} P_k x_k$ satisfying

$$\sum_{k=1}^{l} x_k^\top B_k x_k \leqslant c x^\top M x, \qquad (20)$$

then $M_S \leqslant cM$.

Applying this result in our situation yields the following corollary.

Corollary 6 *Let* $L, A \in \mathbb{R}^{n,n}$ *with* A *symmetric positive definite and* L *nonsingular such that* $M = L^\top AL \leqslant I$. *Set* $E = I - M$ *and* $P = LV$, *where* $V \in \mathbb{R}^{n,p}$ *with* $\operatorname{rank} V = p$. *Furthermore, let* $W \in \mathbb{R}^{n,n-p}$ *be such that* $\operatorname{rank} W = n - p$ *and* $W^\top MV = \mathrm{O}$ *and let* $Z \in \mathbb{R}^{p,p}$ *be symmetric positive definite such that*

$$\gamma P^\top AP \leqslant Z \leqslant \Gamma P^\top AP \tag{21}$$

with positive constants γ, Γ.

1. *If*
$$W^\top W \leqslant \Delta W^\top MW, \tag{22}$$
then for the matrix $M^{(1)}$ *in (6) we have*
$$\frac{\gamma}{\gamma+1} A \leqslant \left(M^{(1)}\right)^{-1} \leqslant \max\{\Gamma, \Delta\} A. \tag{23}$$

2. *If in (21)* $\gamma \geqslant 1$ *and*
$$\begin{bmatrix} \mathrm{O} & \mathrm{O} \\ \mathrm{O} & W^\top MW \end{bmatrix} \leqslant \Delta \left[V, W\right]^\top (M - EME) \left[V, W\right], \tag{24}$$
then for the matrix $M^{(2)}$ *in (8) we have*
$$A \leqslant \left(M^{(2)}\right)^{-1} \leqslant \max\{\Gamma, \Delta\} A. \tag{25}$$

Proof:

1. We apply Theorem 5 to the matrices M, $B_1 = I$, $B_2 = Z$, $P_1 = I, P_2 = L^{-1}P = V$. Set $\Pi = P_2(P_2^\top MP_2)^{-1}P_2^\top M$. Since $\Pi^\top M(I - \Pi) = \mathrm{O}$ we have $I - \Pi = W(W^\top MW)^{-1}W^\top M \equiv \Omega$. It follows that any $x \in \mathbb{R}^n$ can be written as
$$x = \underbrace{(I - \Pi)x}_{x_1} + \underbrace{\Pi x}_{P_2 x_2} = P_1 x_1 + P_2 x_2,$$
where $x_2 = (P_2^\top P_2)^{-1}P_2^\top x$ and $x_1 = \Omega x$. By Theorem 5 it suffices to find a constant $c > 0$ such that
$$x_1^\top x_1 + x_2^\top Z x_2 \leqslant c x^\top Mx.$$
From (21) it follows that
$$\Omega^\top \Omega \leqslant \Delta \Omega^\top M \Omega.$$

Substituting the representation of x_1, x_2 we obtain

$$\begin{aligned} x_1^\top x_1 + x_2^\top Z x_2 &= x^\top \Omega^\top \Omega x + x_2^\top Z x_2 \\ &\leqslant \max\{\Gamma, \Delta\}(x^\top \Omega^\top M \Omega x + x_2^\top (P_2^\top M P_2) x_2) \\ &= \max\{\Gamma, \Delta\}(x^\top \Omega^\top M \Omega x + x^\top \Pi^\top M \Pi x) \\ &= \max\{\Gamma, \Delta\} x^\top (\Omega + \Pi)^\top M (\Omega + \Pi) x \\ &= \max\{\Gamma, \Delta\} x^\top M x. \end{aligned}$$

Thus we have $c = \max\{\Gamma, \Delta\}$ in Theorem 5.

For the other inequality we obtain from

$$M + M^{1/2} P_2 Z^{-1} P_2 M^{1/2} \leqslant M + \frac{1}{\gamma} M^{1/2} P_2 (P_2^\top M P_2)^{-1} P_2^\top M^{1/2} \leqslant M + \frac{1}{\gamma} I$$

that

$$I + P_2 Z^{-1} P_2 \leqslant I + \frac{1}{\gamma} M^{-1} \leqslant (1 + \frac{1}{\gamma}) M^{-1}.$$

Hence we get

$$LL^\top + P Z^{-1} P^\top \leqslant (1 + \frac{1}{\gamma}) A^{-1}.$$

2. To derive the inequalities for $M^{(2)}$ we multiply $M^{(2)}$ by $M^{1/2} L^{-1}$ from the left and its transpose from the right and obtain

$$\begin{aligned} M^{1/2} L^{-1} M^{(2)} L^{-\top} M^{1/2} &= 2M - M^2 + E M^{1/2} V Z^{-1} (M^{1/2} V)^\top E \\ &= I - E \left(I - (M^{1/2} V) Z^{-1} (M^{1/2} V)^\top \right) E. \end{aligned}$$

Setting $\hat{V} = M^{1/2} V$ we have $P^\top A P = \hat{V}^\top \hat{V}$. We set $T = I - \hat{V} (\hat{V}^\top \hat{V})^{-1} \hat{V}^\top$, $\tilde{T} = I - \hat{V} Z^{-1} \hat{V}^\top$. It follows that

$$\begin{aligned} M^{1/2} L^{-1} M^{(2)} L^{-\top} M^{1/2} &= I - E \tilde{T} E \\ &\leqslant I - E \left((1 - \frac{1}{\gamma}) I + \frac{1}{\gamma} T \right) E \\ &\leqslant I - \left(1 - \frac{1}{\gamma} \right) E^2. \end{aligned}$$

If γ is equal to 1 or greater than 1, then the last term is bounded by I. Otherwise the bound will be $\frac{1}{\gamma}$. It follows that

$$(M^{(2)})^{-1} \geqslant \min\{\gamma, 1\} A.$$

For the other direction we can adapt a proof from Theorem 3.1 in [24] to our situation. We have to estimate $E \tilde{T} E$ by a multiple of the identity from above. Note that since $W^\top M^{1/2} \hat{V} = W^\top M V = \mathrm{O}$, (24) is equivalent to

$$M^{1/2} T M^{1/2} \leqslant \Delta (M - E M E)$$

or

$$E^2 \leqslant I - \frac{1}{\Delta} T.$$

Since $\gamma \geqslant 1$, $\tilde{T}^{1/2}$ exists and we have that $E\tilde{T}E \leqslant \beta I$ if and only if $\tilde{T}^{1/2}E^2\tilde{T}^{1/2} \leqslant \beta I$, it follows that

$$\begin{aligned}
\tilde{T} &= T + \hat{V}\left((\hat{V}^\top \hat{V})^{-1} - Z^{-1}\right)\hat{V}^\top \\
&\leqslant T + \left(1 - \frac{1}{\Gamma}\right)\hat{V}(\hat{V}^\top \hat{V})^{-1}\hat{V}^\top
\end{aligned}$$

and since $\tilde{T}T = T = T\tilde{T}$ we obtain

$$\begin{aligned}
\tilde{T}^{1/2}E^2\tilde{T}^{1/2} &\leqslant \tilde{T} - \frac{1}{\Delta}\tilde{T}^{1/2}T\tilde{T}^{1/2} \\
&= \tilde{T} - \frac{1}{\Delta}T \\
&\leqslant (1 - \frac{1}{\Delta})T + \left(1 - \frac{1}{\Gamma}\right)\hat{V}(\hat{V}^\top \hat{V})^{-1}\hat{V}^\top \\
&\leqslant \max\{1 - \frac{1}{\Delta}, 1 - \frac{1}{\Gamma}\}\left(T + \hat{V}(\hat{V}^\top \hat{V})^{-1}\hat{V}^\top\right) \\
&= \max\{1 - \frac{1}{\Delta}, 1 - \frac{1}{\Gamma}\}I.
\end{aligned}$$

From this it follows that

$$(M^{(2)})^{-1} = L^{-\top}M^{1/2}(I - E\tilde{T}E)^{-1}M^{1/2}L^{-1} \leqslant \max\{\Delta, \Gamma\}L^{-\top}ML^{-1} = \max\{\Delta, \Gamma\}A.$$

□

For the operator $M^{(1)}$ one can also estimate the condition number of $M^{(1)}A$ in terms of the angle between the invariant subspace associated with the p smallest eigenvalues of M and V. We refer to [3] for this approach. Note that in (22),(24) we always have $\Delta \geqslant 1$, since $M \leqslant I$. Thus if we set $Z = P^\top AP$ in Corollary 6, then $\gamma = \Gamma = 1$ and the bounds for $M^{(1)}$ are determined by Δ only. Via (22) we see that the inequality for M is only needed on the subspace W which is the M–orthogonal complement of $\operatorname{span} V$. Especially for the choice P in Lemma 3 it is easy to verify that $\Delta = \frac{1}{1-\lambda_{p+1}}$. Thus we obtain a condition number $\kappa^{(1)} = \frac{2}{1-\lambda_{p+1}}$ in Corollary 6, which is only slightly worse than the optimal condition number obtained via Lemma 3, which would give $\kappa^{(1)} = \frac{(1-\lambda_n)(2-\lambda_p)}{(1-\lambda_{p+1})(2-\lambda_1)}$. In a similar way we can compare the bound for $M^{(2)}$ obtained by Corollary 6 with the result of Lemma 3. In this case we obtain $\Delta = \frac{1}{1-\lambda_{p+1}^2}$ and thus $\kappa^{(2)} = \frac{1}{1-\lambda_{p+1}^2}$. Again this is almost the bound of Lemma 3, which would give $\kappa^{(2)} = \frac{1-\lambda_n^2}{1-\lambda_{p+1}^2}$. In this respect, the bounds in Corollary 6 are (almost) as sharp as the optimal bounds in Lemma 3. But the bounds in Corollary 6 can be obtained for any choice of P that has full rank!

Our next result will even sharpen the bounds for $M^{(1)}, M^{(2)}$ in Corollary 6.

Corollary 7 *Let $L, A \in \mathbb{R}^{n,n}$ with A symmetric positive definite and L nonsingular such that $M = L^\top AL \leqslant I$. Set $E = I - M$ and $P = LV$, where $V \in \mathbb{R}^{n,p}$ with $\operatorname{rank} V = p$ Let $W \in \mathbb{R}^{n,n-p}$ with $\operatorname{rank} W = n-p$ and $W^\top MV = \mathrm{O}$ and let $Z \in \mathbb{R}^{p,p}$ be symmetric positive definite such that*

$$\gamma P^\top AP \leqslant Z \leqslant \Gamma P^\top AP \tag{26}$$

for positive constants γ, Γ.

1. *If $\Delta, \hat{\Delta}$ are constants satisfying*

$$W^\top W \leqslant \Delta W^\top MW, \quad W^\top MW \leqslant \hat{\Delta} W^\top M^2 W, \tag{27}$$

then for the matrix $M^{(1)}$ in (6) we have

$$\frac{\gamma}{\gamma+1} A \leqslant (M^{(1)})^{-1} \leqslant \max\{\Gamma, 2\frac{(\Gamma+1)\Delta\hat{\Delta}}{\Delta+\Gamma\hat{\Delta}}\} A. \tag{28}$$

2. *If in (26) $\gamma \geqslant 1$ and $\hat{\Delta}$ is a constant satisfying*

$$W^\top MW \leqslant \hat{\Delta} W^\top (M - EME) W, \tag{29}$$

then for the matrix $M^{(2)}$ in (8) we have

$$A \leqslant (M^{(2)})^{-1} \leqslant \Gamma\hat{\Delta} A. \tag{30}$$

Proof:

1. We set $\Delta = \max\{\Gamma, 2\frac{(\Gamma+1)\Delta\hat{\Delta}}{\Delta+\Gamma\hat{\Delta}}\}$. In view of the proof of Corollary 6 it suffices to show that

$$\left(M^2 + MP_2 Z^{-1} P_2^\top M\right)^{-1} \leqslant \Delta M^{-1}.$$

Multiplying with $[W, P_2]^{-1}$ from the left and its transpose from the right and using the fact that $W^\top MP_2 = \mathrm{O}$ we obtain

$$\begin{pmatrix} W^\top M^2 W & W^\top M^2 P_2 \\ P_2^\top M^2 W & P_2^\top M^2 P_2 + P_2^\top MP_2 Z^{-1} P_2^\top MP_2 \end{pmatrix}^{-1} \leqslant \Delta \begin{pmatrix} (W^\top MW)^{-1} & \mathrm{O} \\ \mathrm{O} & (P_2^\top MP_2)^{-1} \end{pmatrix}.$$

The diagonal blocks of the left hand side matrix are the inverses M_{11}^{-1}, M_{22}^{-1} of the Schur–complements M_{11}, M_{22}, where

$$\begin{aligned} M_{22} &= P_2^\top MP_2 Z^{-1} P_2^\top MP_2 + P_2^\top M \left(I - MW(W^\top M^2 W)^{-1} W^\top M\right) MP_2 \\ &\geqslant \frac{1}{\Gamma} P_2^\top MP_2, \end{aligned}$$

$$\begin{aligned}
M_{11} &= W^\top M^2 W - W^\top M^2 P_2 \left(P_2^\top M^2 P_2 + P_2^\top M P_2 Z^{-1} P_2^\top M P_2\right)^{-1} P_2^\top M^2 W \\
&\geqslant W^\top M^2 W - W^\top M^2 P_2 \left(P_2^\top M^2 P_2 + \frac{1}{\Gamma} P_2^\top M P_2\right)^{-1} P_2^\top M^2 W \\
&\geqslant W^\top M^2 W - \frac{\Gamma}{\Gamma+1} W^\top M^2 P_2 \left(P_2^\top M^2 P_2\right)^{-1} P_2^\top M^2 W \\
&= \frac{1}{\Gamma+1} W^\top M^2 W + \frac{\Gamma}{\Gamma+1} W^\top M \left(I - M P_2 \left(P_2^\top M^2 P_2\right)^{-1} P_2^\top M\right) M W \\
&= \frac{1}{\Gamma+1} W^\top M^2 W + \frac{\Gamma}{\Gamma+1} W^\top M \left(W \left(W^\top W\right)^{-1} W^\top\right) M W \\
&\geqslant \left(\frac{1}{(\Gamma+1)\hat{\Delta}} + \frac{\Gamma}{(\Gamma+1)\Delta}\right) W^\top M W.
\end{aligned}$$

Since for all symmetric positive definite matrices we have

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{12}^\top & A_{22} \end{pmatrix} \leqslant 2 \begin{pmatrix} A_{11} & \mathrm{O} \\ \mathrm{O} & A_{22} \end{pmatrix}$$

inequality (28) follows.

2. To show (30) now we set $\Delta = \Gamma\hat{\Delta}$. Again using the notation of Corollary 6 we have to find $\Delta > 0$ such that

$$\Delta(I - E\tilde{T}E) = \Delta M^{1/2} L^{-1} M^{(2)} L^{-\top} M^{1/2} \geqslant M^{1/2} L^{-1} A^{-1} L^{-\top} M^{1/2} = I$$

or equivalently

$$E\tilde{T}E \leqslant (1 - \frac{1}{\Delta}) I.$$

Note that (29) is equivalent to

$$T E^2 T \leqslant (1 - \frac{1}{\hat{\Delta}}) I$$

This can be seen by multiplying with $M^{1/2}$ from both sides and with $[V, W]$ from the right and its transpose from the left:

$$\begin{aligned}
TE^2T \leqslant (1 - \frac{1}{\hat{\Delta}}) I \quad &\Leftrightarrow \quad [V,W]^\top M^{1/2} T E^2 T M^{1/2} [V,W] \leqslant (1 - \frac{1}{\hat{\Delta}}) [V,W]^\top M [V,W] \\
&\Leftrightarrow \quad \begin{bmatrix} \mathrm{O} & \mathrm{O} \\ \mathrm{O} & W^\top E M E W \end{bmatrix} \leqslant (1 - \frac{1}{\hat{\Delta}}) \begin{bmatrix} V^\top M V & \mathrm{O} \\ \mathrm{O} & W^\top M W \end{bmatrix}.
\end{aligned}$$

It follows that

$$ETE = ET^2E \leqslant (1 - \frac{1}{\hat{\Delta}}) I.$$

Since $\tilde{T} \leqslant (1 - \frac{1}{\Gamma}) I + \frac{1}{\Gamma} T$ it suffices to choose $\Delta > 0$ such that

$$(1 - \frac{1}{\Gamma}) EME + \frac{1}{\Gamma} E M^{1/2} T M^{1/2} E \leqslant \left(1 - \frac{1}{\Gamma} + \frac{1}{\Gamma}(1 - \frac{1}{\hat{\Delta}})\right) M \overset{!}{\leqslant} (1 - \frac{1}{\Delta}) M.$$

$\square$

Note that if $W^\top W \leqslant \Delta W^\top MW$, then we already have $W^\top MW \leqslant \Delta W^\top M^2 W$. Thus $\hat{\Delta} \leqslant \Delta$ and

$$\frac{(\Gamma+1)\Delta\hat{\Delta}}{\Delta+\Gamma\hat{\Delta}} \leqslant \Delta.$$

In this sense the bounds of Corollary 6 are sharper than the bounds of Corollary 6 by a constant factor 2. But if $\Delta_2 \ll \Delta_1$ then we have

$$\frac{(\Gamma+1)\hat{\Delta}}{1+\frac{\Gamma\hat{\Delta}}{\Delta}} \leqslant (\Gamma+1)\hat{\Delta}$$

and this is almost an equality. So if Z is scaled such that $\Gamma = 1$, then we obtain the sharper bound

$$(M^{(1)})^{-1} \leqslant 4\hat{\Delta}A.$$

In other words, the inequality $W^\top MW \leqslant \hat{\Delta}W^\top M^2 W$ gives much better information on the approximation properties than the inequality $W^\top W \leqslant \Delta W^\top MW$.

A similar result holds for $M^{(2)}$. Clearly $\hat{\Delta} \leqslant \Delta$, but $\hat{\Delta} \ll \Delta$ is possible.

In our next theorem we show that we can extend Corollary 6 to the case $l \geqslant 1$.

Theorem 8 *Let $A \in \mathbb{R}^{n,n}$ be symmetric positive definite and consider the algebraic multilevel operators $M_l^{(1)}$, $M_l^{(2)}$ from Definition 4 for some $l \geqslant 1$. Suppose that the matrices L_k are chosen such that $M_k = L_k^\top A L_k \leqslant I$ for all $k = 1, \ldots, l$. Set $E_k := I - M_k$, $P_k = L_k V_k$ and let $W_k \in \mathbb{R}^{n_k, n_k - n_{k-1}}$ with $\operatorname{rank} W_k = n_k - n_{k-1}$ and $W_k^\top M V_k = \mathrm{O}$, for all $k = 1, \ldots, l$.*

1. *If Δ is a constant such that*

$$W_k^\top W_k \leqslant \Delta W_k^\top M_k W_k, \tag{31}$$

for all $k = 1, \ldots, l$, then we have

$$\frac{1}{l+1} A \leqslant \left(M_l^{(1)}\right)^{-1} \leqslant \Delta A. \tag{32}$$

2. *If Δ is a constant such that*

$$\begin{bmatrix} \mathrm{O} & \mathrm{O} \\ \mathrm{O} & W_k^\top M_k W_k \end{bmatrix} \leqslant \Delta \left[V_k, W_k\right]^\top \left(M_k - E_k M_k E_k\right) \left[V_k, W_k\right], \tag{33}$$

for all $k = 1, \ldots, l$, then we have

$$A \leqslant \left(M_l^{(2)}\right)^{-1} \leqslant \Delta A. \tag{34}$$

Proof:
We use induction on l. For $l=1$ the result is given by Corollary 6 applied to $Z = P_{1,0}^\top A P_{1,0}$. Next we apply Corollary 6 to $A_{l-1}, M_{l-1}^{(1)}$, i.e., let Δ be a constant such that

$$\frac{1}{l} A_{l-1} \leqslant (M_{l-1}^{(1)})^{-1} \leqslant \Delta A_{l-1}.$$

Then with $Z = (M_{l-1}^{(1)})^{-1}$ we obtain $\gamma = \frac{1}{l}, \Gamma = \Delta$. But $\frac{\gamma}{1+\gamma} = \frac{1}{l+1}$ and hence (32) follows.

Inequality (34) is proved analogously. □

In a similar way we can derive bounds based on Corollary 7. But this is a lot more technical, since Γ is not as well isolated in (28) as in (23), so we omit this construction here.

By Theorem 8 we only loose a factor $\frac{1}{l+1}$ compared with the case $l=1$. If the reduction of the size in A_k in any step is sufficient, i.e., for example if the size of A_{k-1} is half the size of A_k or less, then this corresponds to a logarithmic factor in n. In this case the factor $1/(l+1)$ is neglectible.

For the construction of the multilevel method we still need an algorithm for the construction of a well–suited matrix P_k in each step. This will be the topic of the next section.

4 The Coarsening Process

So far we have not discussed the concrete construction of P for given L, A. As before we set $L^\top A L = M$, $E = I - M$ and assume that $E \geqslant \mathrm{O}$.

We have already seen in Lemma 3, that an invariant subspace V of E associated with the large eigenvalues of E is a good candidate for $P = LV$. But in practice we neither have such an invariant subspace nor is this a favourable choice, since this choice of P will be typically full and a further coarsening of $P^\top AP$ will be almost impossible since this matrix is no longer sparse. So we need a different choice of $P = LV$.

4.1 Coarsening via $QR\Pi$ decomposition

By Lemma 2 we can use a suitably chosen set of columns of E to approximate its singular vectors associated with the large singular values. According to (4) this can be achieved via a QR–like decomposition of the form $E = MQR + F$ with a matrix F of small norm. To illustrate that E or more precisely selected columns of E may be a good candidate for constructing P we have the following lemma.

Lemma 9 *Let $M, E \in \mathbb{R}^{n,n}$ nonsingular. Suppose we have a decomposition*

$$E \underbrace{[\Pi_1, \Pi_2]}_{\Pi} = \underbrace{[V, W]}_{Q} \underbrace{\begin{bmatrix} R_{11} & R_{12} \\ \mathrm{O} & R_{22} \end{bmatrix}}_{R}, \tag{35}$$

where Π *is a permutation matrix* $[V,W]$ *is nonsingular and* $V^\top MW = \mathrm{O}$. *Suppose that* $\hat{\Delta}$ *satisfies (29). Then there exist* R, F *such that*

$$E = M(E\Pi_1)R + F \tag{36}$$

with $\|F\|_2^2 \leqslant 1 - \frac{1}{\hat{\Delta}}$.

Proof:
Since $[V,W]$ is nonsingular and $W^\top MV = \mathrm{O}$ we have

$$I = M^{1/2}V(V^\top MV)^{-1}V^\top M^{1/2} + M^{1/2}W(W^\top MW)^{-1}W^\top M^{1/2}.$$

We set $R = R_{11}^{-1}(V^\top MV)^{-1}V^\top E$. With this choice of R we have

$$\begin{array}{rcl} F & \equiv & E - M(E\Pi_1)R \\ & = & E - MVR_{11}R \\ & = & E - MV(V^\top MV)^{-1}V^\top E \\ & = & E - M^{1/2}V(V^\top MV)^{-1}V^\top M^{1/2}E \\ & = & M^{1/2}W(W^\top MW)^{-1}W^\top M^{1/2}E. \end{array}$$

From this it follows that

$$\|F\|_2^2 = \|(W^\top MW)^{-1/2}W^\top M^{1/2}E\|_2^2 = \sup_{x\neq 0} \frac{x^\top W^\top EMEWx}{x^\top W^\top MWx}.$$

$\square$

As a consequence of Lemma 9, from a QR–decomposition of E

$$E[\Pi_1,\Pi_2] = [V,W]R,\ V^\top MW = \mathrm{O} \tag{37}$$

we obtain a projection matrix $P = LE\Pi_1 = LVR_{11}^{-1}$ such that the remaining error matrix F has small norm. Clearly there is no restriction in replacing V by $E\Pi_1$, since by $V = E\Pi_1R_{11}^{-1}$ both sets of columns span the same space. But $M^{(1)}, M^{(2)}$ do not change if we replace V by VR_{11}. In contrast to V, $E\Pi_1$ is typically sparse. Moreover, we obtain $P \equiv LE\Pi_1$ as coarse grid projection matrix from the QR–decomposition (37) for which the bounds of Lemma 7 hold. Here the columns of V,W are not required to be orthogonal in the standard inner product as one typically requires in a QR–decomposition, but they are orthogonal with respect to the inner product defined by M.

It remains to discuss the pivoting strategy. Clearly the best we can do is to locally maximize Δ or $\hat{\Delta}$ in the inequalities (22),(24) or (27),(29) to obtain a feasible coarse grid matrix $P = LE\Pi_1$ for $M^{(1)}$ in (6) and for $M^{(2)}$ in (7).

We still have to derive a method to succesively adapt the pivoting strategy to the QR–decomposition. For fixed p there exist $\binom{n}{p}$ permutations that have to be checked and for any of these choices one has to compute a QR decomposition of an $n \times p$ matrix $E\Pi_1$ and the corresponding Δ or $\hat{\Delta}$. Already for small p this is expensive, so in practice not more than $p = 1$ can be achieved in one step. To this end in step k of a QR–decomposition we have a decomposition

$$E\Pi^{(k)} = \left[V^{(k)}, W^{(k)}\right] R^{(k)}, \tag{38}$$

where $V^{(k)}$ is an $n \times k$ matrix, $W^{(k)}$ is an $n \times (n-k)$ matrix such that $(V^{(k)})^\top M W^{(k)} = \mathrm{O}$. To find the optimal next pivot column, i.e., $\Pi^{(k+1)}$, we would have to calculate for example Δ from $(W^{(k+1)})^\top W^{(k+1)} \leqslant \Delta (W^{(k+1)})^\top M W^{(k+1)}$ for any choice $W^{(k+1)}$ depending on all $n-k$ possible permutations. This is expensive and cannot be done efficiently in practice. But let us for a while delay this problem and simply formulate an abstract algorithm that computes this decomposition. We use **MATLAB** notation [1], i.e., for $A = (a_{ij})_{i=1,\ldots,m,j=1,\ldots,n} \in \mathbb{R}^{m,n}$ and $1 \leqslant i_1, i_2 \leqslant m$, $1 \leqslant j_1, j_2 \leqslant n$ we set $A(i_1{:}i_2, j_1{:}j_2) = (a_{ij})_{i_1 \leqslant i \leqslant i_2, j_1 \leqslant j \leqslant j_2}$. The : denotes every column/row.

An adapted modified Gram–Schmidt process [13] then has the following form.

Algorithm 10 (Modified Gram–Schmidt with column pivoting)
Let $E \in \mathbb{R}^{n,n}$ and set $R = \mathrm{O} \in \mathbb{R}^{n,n}$.
Set $\pi = (1, \ldots, n)$.
for $p = 1, 2, \ldots, n$
  Choose $k \in \{p, \ldots, n\}$ according to some strategy from (27), (24), (29).
  Interchange $\pi(p)$ and $\pi(k)$.
  $q = ME(:, \pi(p))$
  $R(p, \pi(p)) = \sqrt{q^\top E(:, \pi(p))}$
  $E(:, \pi(p)) = E(:, \pi(p))/R(p, \pi(p)), q = q/R(p, \pi(p))$
  $R(p, \pi(p+1{:}n)) = q^\top E(:, \pi(p+1{:}n))$
  $E(:, \pi(p+1{:}n)) = E(:, \pi(p+1{:}n)) - E(:, \pi(p))R(p, \pi(p+1{:}n))$
end

Note that for $M = I$ this is the standard modified Gram–Schmidt procedure with column pivoting. It is easy to see that successively we construct a decomposition of the form

$$E[\Pi_1, \Pi_2] = [V, W] \begin{bmatrix} R_{11} & R_{12} \\ \mathrm{O} & I \end{bmatrix}, \tag{39}$$

where $V^\top M V = I, W^\top M V = \mathrm{O}$ and increasing number of columns in V. Using $V^\top M V = I$ we set

$$T = I - VV^\top M \tag{40}$$

and it is easy to see that

$$W = TE\Pi_2. \tag{41}$$

In Algorithm 10 the columns of V are stored in the leading columns $1, \ldots, p$ of E, that are overwritten in the algorithm. Likewise W is stored in the remaining columns of the overwritten E.

Two major items still need to be discussed. We have to fix a column pivoting strategy that is computable with a moderate amount of computing time and we have to approximate the "full" QR–decomposition by an incomplete, sparse QR–decomposition, since otherwise this computation obviously dominates our construction of a preconditioner in an infeasible way. For the choice of the pivoting strategy there are many possibilities. We will describe here only briefly one partially heuristic technique. Optimal strategies are currently not known.

4.2 The Incomplete QR Decomposition

Let us first study the approximation of the QR decomposition by an approximate incomplete, sparse QR decomposition. Here use an idea of Stewart [28] for a truncated QR approximation.

Suppose we have a QR decomposition of the form

$$E \equiv [E_1, E_2] = \underbrace{[V, W]}_{Q} \underbrace{\begin{bmatrix} R_{11} & R_{12} \\ \mathrm{O} & R_{22} \end{bmatrix}}_{R}, \tag{42}$$

where Q is orthogonal, $R_{11} \in \mathbb{R}^{p,p}$ is nonsingular and the remaining matrices have corresponding sizes.

Obviously we have

$$E_1 = VR_{11},\ R_{12} = V^\top E_2. \tag{43}$$

The main observation of Stewart is that we do not have to compute V explicitly. It suffices to locally recompute the p–th column v_p of V from (43) by

$$v_p = E_1 R_{11}^{-1} e_p$$

and the last row p of R_{12} can be obtained from $v_p^\top E_2$. We can also discard the part R_{12} for further computations. It will only be required to update the norm of any column of the projected $(I - VV^\top)E$. Since it is necessary to have R_{11} available when going from step p to $p+1$ one column of R_{12} is required. Parts of R_{12} that have been discarded can be recomputed from the relation

$$R_{11}(1:p, p+1) = V^\top E(:, p+1) = R_{11}^{-\top}(E_1^\top E_2(:, 1)). \tag{44}$$

For details see [28]. What is remarkable in this approach is that we can compute the QR decomposition of E without storing the Q–part. This saves a lot of memory since even in large sparse calculations Q is typically not sparse. Clearly this method can be generalized to the case when column pivoting is used. And for our problem here, where $V^\top MV = I$ instead of $V^\top V = I$, the changes are obvious.

Algorithm 11 (MGS with implicit Q)
Let $E \in \mathbb{R}^{n,n}$ *and set* $R = \mathrm{O} \in \mathbb{R}^{n,n}$.
Set $\pi = (1, \dots, n)$ *and* $\nu = \big(E(:,1)^\top ME(:,1), \dots, E(:,n)^\top ME(:,n)\big)$.
for $p = 1, 2, \dots, n$

Choose $k \in \{p, \dots, n\}$ *according to some strategy from (27), (24), (29).*
Interchange $\pi(p)$ *and* $\pi(k)$
if $p > 1$, *solve* $R(1:p-1, \pi(1:p-1))^\top v = E(:, \pi(1:p-1))^\top M E(:, \pi(p))$
$R(1:p-1, \pi(p)) = v$
$R(p, \pi(p)) = \sqrt{\nu(\pi(p))}$
Solve $R(1:p, \pi(1:p))z = \begin{bmatrix} 0 & \cdots & 0 & 1 \end{bmatrix}^\top$
$q = ME(:, \pi(1:p))z / R(p, \pi(p))$
$S(1, \pi(p+1{:}n)) = q^\top E(:, \pi(p+1{:}n))$
$\nu(\pi(p+1{:}n)) = \nu(\pi(p+1{:}n)) - S(1, \pi(p+1{:}n))^2$
end

In this algorithm $S(1, \pi(p+1{:}n))^2$ has to be performed component wise. S temporarily stores $R(p, \pi(p+1{:}n))$. Note that since we are working with M–orthogonal projections , like $(I - M^{-1}qq^\top)$, the square of the M–norm of the projected matrix $(I - M^{-1}qq^\top)E$ can be obtained by taking the squares of the initial M–norm (initial value of ν) and subtracting the squares of the angles between the columns of E and q. But these values are stored in $S(1, \pi(p+1{:}n))^2$. From Algorithm 11 as well as from Algorithm 10 we see that in the QR–decomposition we never need the first p M–orthonormal columns explicitely, but only M times these columns. For this reason, if we have enough memory available, q need not be recomputed from E via z, but if we store

$$\Phi(:, \pi(p)) = q, \tag{45}$$

then obviously $\Phi(:, \pi(1:p))$ from (45) corresponds to the updated value of $ME(:, \pi(1;p))$ in Algorithm 10. Thus clearly q can be recomputed via

$$q = [ME(:, \pi(p)) - \Phi(:, \pi(1:p-1))R(1:p-1, \pi(p))] / \sqrt{\nu(\pi(p))}. \tag{46}$$

As long as we have full matrices, to save memory when omitting Φ from (45) and the R part of the QR decomposition is advantageous. In practice, we have to sparsify this algorithm using some strategy, e.g. a drop tolerance for the entries of Φ, R. But in this case we may have only a few number of nonzero entries in $R(1:p-1, \pi(p))$. So (46) reduces to a selective reorthogonalization. If in addition Φ is not too full, (46) may turn out to be quite cheap. Conversely, computing z via $R(1:p, \pi(1:p))z = \begin{bmatrix} 0 & \cdots & 0 & 1 \end{bmatrix}^\top$ may still be relatively cheap if R is sparse. But even if $R(1:p, \pi(1:p))$ is sparse, then all entries of z have to be computed. We do not know in advance whether some entries of z are small in magnitude. This reduces the efficiency of computing z. Also z may be full in this computation, since typically the inverse of $R(1:p, \pi(1:p))$ will be full even if R is sparse. Thus computing q from $q = ME(:, \pi(1:p))z/R(p, \pi(p))$ may turn out to be more expensive in this case, more expensive than computing q from (46). In contrast to this, computing v from $R(1:p-1, \pi(1:p-1))^\top v = E(:, \pi(1:p-1))^\top ME(:, \pi(p))$ will typically be cheap, since $E(:, \pi(1:p-1))^\top ME(:, \pi(p))$ only has nonzeros at most in those positions corresponding to the nodes with distance less than or equal to 3 from $\pi(p)$ in the graph theoretical sense. This essentially follows from the fact that M and E have the same graph and the nonzeros of E^3 correspond to the paths with length $\leqslant 3$. If these neighbours of $\pi(p)$ have been recently used as pivots in the QR decomposition, then only the last few entries of $E(:,\pi(1:p-1))^\top ME(:,\pi(p))$ will be different from 0. Thus solving a system with the lower triangular matrix $R(1:p-1,\pi(1:p-1))^\top$ will be very cheap.

According to these (of course heuristic) arguments, the most natural strategy seems to combine Algorithm 11 with Algorithm 10. Instead of dropping parts of Φ, R we may only drop certain entries, if the memory is exceeded. In this case we can drop several columns of Φ, R to release parts of the memory. If it turns out that these column of Φ, R are needed in a later step, then we can recompute them using Stewart's method. The only question is, which columns should be dropped. A simple and natural measure seems to be to take into account how many steps ago a column of Φ, R has been touched, i.e. used to compute q, $S(1,\pi(p+1:n))$. Since we will sparsify this QR decomposition later on, only a few entries will be touched in each step. In addition we should take into account how many nodes in a neighbourhood of a node belong to the coarse grid. If there are many nodes in the neighbourhood belonging to the coarse grid, then possibly the actual node will no longer be needed to detect further coarse grid nodes.
As a consequence a good measure or cost functional to select columns of Φ to be dropped is the number of coarse grid nodes in the neighbourhood of the already selected columns $\pi(1:p)$ combined with the number of steps that have been passed when this column has been touched last time in the Algorithm. For the columns of the upper triangular matrix R, especially for those entries not belonging to the coarse grid, i.e. those entries, which do not have a column stored in Φ, an analogous strategy will be applicable.

Algorithm 12 (MGS with partially implicit Q)
Let $E \in \mathbb{R}^{n,n}$ *and set* $R = \mathrm{O} \in \mathbb{R}^{n,n}$.
Set $\pi = (1,\dots,n)$.
Set $\nu = \big(E(:,1)^\top ME(:,1),\dots,E(:,n)^\top ME(:,n)\big)$.
for $p = 1,2,\dots,n$
  Choose $k \in \{p,\dots,n\}$ *according to some strategy from (27), (24), (29).*
  Interchange $\pi(p)$ *and* $\pi(k)$
  if $p > 1$ *and column* $\pi(k)$ *of* R *is dropped*
    Solve $R(1:p-1,\pi(1:p-1))^\top v = E(:,\pi(1:p-1))^\top ME(:,\pi(p))$
    $R(1:p-1,\pi(p)) = v$
  end
  $R(p,\pi(p)) = \sqrt{\nu(\pi(p))}$
  if *any column of* Φ *associated with the nonzero entries of* $R(1:p-1,\pi(p))$ *is dropped*
    Solve $R(1:p,\pi(1:p))z = \begin{bmatrix} 0 & \cdots & 0 & 1 \end{bmatrix}^\top$
    $q = ME(:,\pi(1:p))z$
  else
    $q = [ME(:,\pi(p)) - \Phi(:,\pi(1:p-1))R(1:p-1,\pi(p))]/R(p,\pi(p))$.
  end
  $\Phi(:,\pi(p)) = q$
  $R(p,\pi(p+1:n)) = q^\top E(:,\pi(p+1:n))$
  $\nu(\pi(p+1:n)) = \nu(\pi(p+1:n)) - R(p,\pi(p+1:n))^2$
  if necessary, drop some columns of $\Phi, R(:,\pi(p+1:n))$
end

In this form the algorithm still requires a lot of computational work In addition, neither Φ nor R will be sparse even if ME is sparse. What we can do to is to reduce the computational costs as well as the storage requirements by introducing a drop tolerance for Φ, R. We suggest the following strategy. Suppose that a tolerance tol is given. Then the following entries of q are kept in Algorithm 12

- any entry greater than $tol \max |q|$
- any entry greater than $tol^2 \max |q|$ for components i such that their distance to p is less or equal to 2 (i.e. a neighbourhood of p)

To use a smaller tolerance in the neighbourhood of p is only justified by a heuristic argument that typically q is essentially reorthogonalized with respect to nodes from a neighbourhood of p since already in the initial matrix ME these entries are not orthogonal.

For R we can compare for any $i > p$ the entry $R(p, \pi(i))$ with $\nu(\pi(i))$. If in the next step i would become a pivot column, then $\nu(\pi(i))$ would become the diagonal entry in R. This gives a natural dropping criterion.

- Keep $R(p, \pi(i))$, if $R(p, \pi(i))^2 > tol^2 \nu(\pi(i))$

In this subsection we have discussed (partially heuristic) strategies for the computation of an efficient sparse, incomplete QR-factorization and it remains to discuss the pivoting strategy.

4.3 Approximate Solutions to the Eigenvalue Problems

The final step is the construction of the pivoting strategy. Basically this strategy can be based on the approximation of eigenvectors associated with the smallest eigenvalues of $M = L^\top AL$, leading to the inequalities (22) and (24), i.e.,

$$\frac{1}{\Delta} = \min_{y \neq 0} \frac{y^\top W^\top M W y}{y^\top W^\top W y}, \ \frac{1}{\hat{\Delta}} = \min_{y \neq 0} \frac{y^\top W^\top M^2 W y}{y^\top W^\top M W y} \tag{47}$$

or analogous eigenvalue problems from (24), (29).

A simple observation is that using the matrix T from (40) we can rewrite these eigenvalue problems (47) as

$$\frac{1}{\Delta} = \min_{Tx \neq 0} \frac{x^\top T^\top M T x}{x^\top T^\top T x}, \ \frac{1}{\hat{\Delta}} = \min_{Tx \neq 0} \frac{x^\top T^\top M^2 T x}{x^\top T^\top M T x}. \tag{48}$$

Likewise we can reformulate (24),(29) as

$$\frac{1}{\Delta} = \min_{Tx \neq 0} \frac{x^\top (M - EME) x}{x^\top T^\top M T x}, \ \frac{1}{\hat{\Delta}} = \min_{Tx \neq 0} \frac{x^\top T^\top (M - EME) T x}{x^\top T^\top M T x}. \tag{49}$$

Suppose that we have a good approximation to the eigenvector x of M associated with the smallest eigenvalue. Initially we have $T = I$ and clearly x in (49) is the eigenvector of M with respect to its smallest eigenvalue. We could proceed in the next steps by replacing x by the projection $\hat{x} = Tx$ which is the orthogonal projection with respect to M and use this fixed vector $\hat{x}$ instead of solving the whole eigenvalue problem (48), (49). The question is, whether $\hat{x} = Tx$ is a good approximate solution of the eigenvalue problem. Clearly, even if x is the exact eigenvector of M with respect to its smallest eigenvalue, x need not to be an eigenvector of the projected eigenvalue problems (48), (49). But we may expect that x may still serve as a good estimate for the projected eigenvalue problems as the following Lemma will show.

Lemma 13 *Let $M \in \mathbb{R}^{n,n}$ be symmetric positive definite and let $V \in \mathbb{R}^{n,p}$ be such that $V^\top MV = I$. Suppose, furthermore, that x is an eigenvector of M associated with its smallest eigenvalue λ. Let $T = I - VV^\top M$ and define τ via*

$$\tau = \sup_{y \neq 0} \frac{|y^\top V^\top Mx|}{\|Vy\|_{M^{1/2}} \|x\|_{M^{1/2}}} = \frac{\sqrt{\lambda}}{\|x\|_2} \|V^\top x\|_2 \tag{50}$$

Then

$$\tau\sqrt{\lambda} \leqslant \frac{\|Tx - x\|_2}{\|x\|_2} \leqslant \|V\|_2 \tau\sqrt{\lambda}, \; \frac{\|Tx - x\|_{M^{1/2}}}{\|x\|_{M^{1/2}}} = \tau. \tag{51}$$

Proof:
We have $Tx - x = VV^\top Mx = VV^\top Mx$. Since $I = V^\top MV \leqslant V^\top V$ it follows that

$$\|V^\top Mx\|_2 \leqslant \|Tx - x\|_2 \leqslant \|V\|_2 \|V^\top Mx\|_2$$

But

$$\|V^\top Mx\|_2 = \tau \|V\|_{M^{1/2}} \|x\|_{M^{1/2}} = \tau\sqrt{\lambda}\|x\|_2.$$

Analogously we can prove the inequality with respect to the $M^{1/2}$–norm. □
Note that τ can be viewed as cosine of the angle between x and V in the M–inner product.

From Lemma 13 we conclude that as long as the smallest eigenvalue of M is small and the angle between x and V is large, x and Tx are almost identical. But clearly this is the only interesting case if we want to add an additional coarse grid correction to M. As long as the coarsening process has not constructed a coarse grid matrix V which covers x, x will be an ideal candidate to approximately solve the eigenvalue problems (48), (49).

Based on Lemma 13 we can also conclude that as long as $x \approx Tx$, we will have $\frac{x^\top Mx}{x^\top x} \approx \frac{x^\top T^\top MTx}{x^\top T^\top Tx}$. From this point of view it is advisable to to examine this ratio more precisely. We will see that successively maximizing $\|V^\top x\|_2$ will be almost equivalent. This confirms the observation made by Lemma 13 that the angle between V and x is a measure that we have to look for.

Lemma 14 *Under the assumptions of Lemma 13 we have*

$$\lambda \frac{1 - \tau^2}{1 - (2 - \lambda\|V\|^2)\tau^2} \leqslant \frac{x^\top T^\top MTx}{x^\top T^\top Tx} \leqslant \lambda \frac{1 - \tau^2}{1 - (2 - \tau^2)\tau^2} \tag{52}$$

and

$$\lambda \leqslant \frac{x^\top T^\top M^2 T x}{x^\top T^\top M T x} \leqslant \frac{\lambda + (1-2\lambda)\tau^2}{1-\tau^2}. \tag{53}$$

Proof:
For simplicity we may assume that $\|x\|_2 = 1$. Since $Mx = x\lambda$ we have

$$\begin{aligned} x^\top T^\top T x &= x^\top (I - MVV^\top - VV^\top M + MVV^\top VV^\top M)x \\ &= 1 - 2\lambda \|V^\top x\|_2^2 + \lambda^2 x^\top (VV^\top)^2 x \\ &= 1 - 2\tau^2 + \lambda^2 x^\top (VV^\top)^2 x \end{aligned}$$

Using the Cauchy–Schwarz inequality we get $\|V^\top x\|_2^4 = (x^\top VV^\top x)^2 \leqslant x^\top (VV^\top)^2 x$. From this it follows that

$$x^\top T^\top T x \left\{ \begin{array}{cc} \leqslant & 1 - (2 - \lambda \|V\|_2^2)\tau^2 \\ \geqslant & (1-\tau)^2 \end{array} \right.$$

Finally we consider $x^\top T^\top M T x$. We immediately obtain

$$x^\top T^\top M T x = x^\top (M - MVV^\top M)x = \lambda(1-\tau^2).$$

Next estimating $x^\top T^\top M^2 T x$ gives

$$x^\top T^\top M^2 T x = \lambda^2 - 2\lambda^3 \|V^\top x\|_2^2 + x^\top (MVV^\top M)^2 x = \lambda^2 - 2\lambda^2\tau^2 + x^\top (MVV^\top M)^2 x.$$

Using $M \leqslant I$ we obtain

$$x^\top T^\top M^2 T x \left\{ \begin{array}{cc} \leqslant & \lambda^2 - 2\lambda^2\tau^2 + \lambda\tau^2 \\ \geqslant & \lambda^2 - \lambda^2\tau^2 \end{array} \right.$$

Combining the inequalities yields the assertion of the lemma. □

It follows that successively maximizing $\|V^\top x\|_2^2$ is almost equivalent to successively maximizing the smallest eigenvalue of eigenvalue problem (48). It is clear that one can easily derive analogous bounds for (49).

We know that the eigenvector x associated with the smallest eigenvalue of M can serve as an estimate for the approximate solution of eigenvalue problems (49), (13). It will be a reliable estimate if the initial matrix M is not ill–conditioned and as long as the angle between the coarse grid space V and x is large. Minimizing the angle, i.e. maximizing $\|V^\top x\|$ will be essentially equivalent to maximizing the smallest eigenvalue in (49), (13). It remains to find a good estimate for x. There are many ways to do this. For example we could approximate x via a few steps of a Lanczos procedure [23]. We will also present a graph based heuristic in subsection 4.5 But before we discuss this heuristic eigenvector approximation, in the next subsection we describe an indicator for the pivoting strategy which is not based on the eigenvector x.

4.4 An Eigenvector Independent Indicator

We now discuss how to find an indicator that is (essentially) independent on the choice of x. To this end we note that if we want to successively maximize $\|V^\top x\|_2$, then in principle we have to replace x by $\hat{x}$, where $\hat{x}$ is the exact eigenvector of one of the projected eigenvalue problems in (48), (49). Although we know that x may serve as an estimate for $\hat{x}$, we can measure how far x is away from $\hat{x}$ in terms of the residual.

Lemma 15 *Let $M \in \mathbb{R}^{n,n}$ be symmetric positive definite, $V \in \mathbb{R}^{n,p}$ with $V^\top MV = I$. Set $T = I - VV^\top M$. If $Mx = x\lambda$, then we have*

$$T^\top MTx - T^\top Tx\lambda = (V - MVV^\top V)V^\top x\lambda^2. \tag{54}$$

Let $V = [\hat{V}, v], v \in \mathbb{R}^n$ and let $\hat{T} = I - \hat{V}\hat{V}^\top M$. If $\hat{T}^\top M\hat{T}y = \hat{T}^\top\hat{T}y\mu$, then we have

$$T^\top MTy - T^\top Ty\mu = (v - MVV^\top v)\, v^\top My\mu. \tag{55}$$

Proof:
Identity (54) is clear. Since $\hat{V}^\top Mv = 0$ we have

$$\begin{aligned} T^\top MTy &= (M - M[\hat{V}, v][\hat{V}, v]^\top M)y \\ &= \hat{T}^\top M\hat{T}y - Mvv^\top My \end{aligned}$$

and

$$\begin{aligned} T^\top Ty\mu &= (\hat{T}^\top\hat{T} - Mvv^\top\hat{T}^\top\hat{T} - \hat{T}^\top\hat{T}vv^\top M + Mvv^\top\hat{T}^\top\hat{T}vv^\top M)y\mu \\ &= \hat{T}^\top M\hat{T}y - Mvv^\top\hat{T}^\top M\hat{T}y - \hat{T}^\top vv^\top My\mu + Mvv^\top vv^\top My\mu \\ &= \hat{T}^\top M\hat{T}y - Mvv^\top My - \hat{T}^\top vv^\top My\mu + Mvv^\top vv^\top My\mu. \end{aligned}$$

It follows that

$$\begin{aligned} T^\top MTy - T^\top Ty\mu &= \hat{T}^\top vv^\top My\mu - Mvv^\top vv^\top My\mu \\ &= (v - MVV^\top v)v^\top My\mu. \end{aligned}$$

□

Now we want to successively maximize $\|V^\top x\|$. In this case the components of $V^\top x$ will typically significantly increase. But if this is the case, then $V^\top x$ will be almost a multiple of e_p, which is the p–th unit vector. In this case the residual will be approximately

$$T^\top MTx - T^\top Tx\lambda \approx \alpha(v - MVV^\top v) \tag{56}$$

for some $\alpha \in \mathbb{R}$. Analogously if we replace in step $p-1$ the initial exact eigenvector x of M by y, where y is the exact eigenvector of the projected eigenvalue problem $\hat{T}^\top M\hat{T}y - \hat{T}^\top\hat{T}y\mu$ the relation

$$T^\top MTy - T^\top Ty\mu = \beta(v - MVV^\top v) \tag{57}$$

holds for some $\beta \in \mathbb{R}$. Both relations show that the residual is essentially independent on x, respectively y, i.e. we can compute $v - MVV^\top v$ without ever knowing x, y! We could carry out similar computations with respect to the eigenvalue problem $T^\top M^2 Ty - T^\top MTy\mu$. It is easy to verify that analogous relations hold and the corresponding residuals in this case are

$$(M^2V - MVV^\top M^2V)V^\top x\lambda,\ (M^2v - MVV^\top M^2v)v^\top My. \tag{58}$$

For the eigenvalue problem $(M - EME)y - T^\top MTy\mu(2-\mu)$ we obtain the residual

$$MVV^\top x\lambda^2(2-\lambda),\ Mvv^\top My\mu(2-\mu). \tag{59}$$

and finally for the eigenvalue problem $T^\top EMETy - T^\top MTy\mu(2-\mu)$ we obtain the residual

$$(E^2MV - MVV^\top EMEV)V^\top x\lambda,\ (E^2Mv - MVV^\top EMEv)v^\top My. \tag{60}$$

Knowing that we can compute the residual without explicit knowledge of x, y, the residual will give us information on the direction in which the previous exact eigenvector looses its accuracy once we change over to the projected eigenvalue problem. To get a good coarse grid matrix it would be nice if $Tx \not\approx x$ or $V^\top x$ has a large new component. In terms of the residual this would mean that a large residual indicates that the eigenvector y of the projected eigenvalue strongly differs from the eigenvector x of the unprojected eigenvalue problem. If we have chosen one column of W to become v in the sense of Lemma 15 at some step p of the QR–Algorithm 10, then the residual tells us a posteriori the direction where x has lost its accuracy compared with y from the projected eigenvalue problem. So instead of directly maximizing $V^\top x$ in the next step we can use the large components of the residual to predict the next pivot column and validate this choice by evaluating the approximate eigenvalue problem with x.

The question is how to find a good approximation to the eigenvector with respect to smallest eigenvalue. Many strategies are possible for this, we could for example use a few steps of a Lanczos procedure [23]. Here we present a different heuristic which is based on the graph of the matrix.

4.5 A Graph Based Heuristic for Computing an Approximate Eigenvector

Suppose that we choose x such that $Mx = e_i$, where e_i is the i-th unit vector and i is suitably chosen. Clearly with this choice we would typically have $\|Mx\| \ll \|x\|$ and we could use x as an approximate eigenvector. The question is how to choose i and how to find a good initial guess for x. Here we make use of a heuristic that works in many but clearly not in all applications of symmetric positive definite matrices.

Since we have a sparse matrix and therefore also the graph of the matrix has only few edges, we make use of the undirected graph of the matrix to determine an approximate eigenvector of M with respect to the smallest eigenvalue.

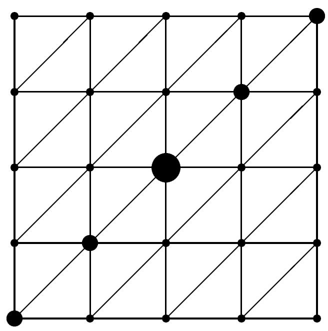
Definition 16 *An undirected graph $G = (V, E)$ is given by a set of nodes $V = \{1, \ldots, n\}$, $n \in \mathbb{N} \setminus \{0\}$, and a set of edges $E \subseteq \{\{i, j\} : i \neq j, 1 \leqslant i, j \leqslant n\}$.*
For $A = (a_{ij})_{i,j} \in \mathbb{R}^{n,n}$ the associated undirected graph $G = (V, E)$ is given by $V = \{1, \ldots, n\}$ and $E = \{\{i, j\} : i \neq j, 1 \leqslant i, j \leqslant n, a_{ij} \neq 0\}$.

The heuristic that we are using is based on the observation that in many applications in elliptic partial differential equations the entries of the eigenvector associated with the smallest eigenvalue increase the more the corresponding node is lying in the center of the graph, that is a node which minimizes the distance to all other nodes in the graph (in Definition 17 we will precisely define center and distance). Picking i to be the index of such a central node may be a good choice. This can be viewed as algebraic justification of the importance of a node in the graph. So to pick a vector for which its components decrease with its distance growing when leaving the center seems to be a good candidate. In what follows we briefly introduce an algorithm that computes a center of a graph and a corresponding initial guess.

Definition 17 *Let $G = (V, E)$ be an undirected graph.*
*A **path** from node i to j is a sequence of edges $\{k_0, k_1\}, \{k_1, k_2\}, \ldots, \{k_{l-1}, k_l\}$ such that $i = k_0, j = k_l$ and l is the **length** of the path, for $i = j$ we set $l = 0$.*
*The **distance** $\operatorname{dist}(i, j)$ is the smallest length of any path between node i and j and for $C \subseteq V$ we set $\operatorname{dist}(i, C) = \min_{j \in C} \operatorname{dist}(i, j)$.*
*A **central node** c of the graph is a node such that $\sum_{i \in V} \operatorname{dist}(i, c) \stackrel{!}{=} \min$.*

We also say that a central node is in the center of the graph.

Example 18 *Consider the following graph*



The five nodes on the diagonal from the lower left corner to the upper right corner all satisfy $\max_{i \in V} \operatorname{dist}(i, c) \stackrel{!}{=} \min$ but only the node in the middle satisfies $\sum_{i \in V} \operatorname{dist}(i, c) \stackrel{!}{=} \min$.

The question is now how to find a node in the center of a graph or how to find a node that is almost in the center of the graph. A technique which is well–known [11, 8] in sparse matrix computation for direct solution methods helps us to find a central node.

Definition 19 *Let* $G = (V, E)$ *be an undirected graph and* i *be a node. The* ***level structure*** *of* $\mathcal{L}_0 \subseteq V$ *is a collection of level sets* $\mathcal{L}_0, \dots, \mathcal{L}_k$ *such that* $\mathcal{L}_0 \cup \dots \cup \mathcal{L}_k = V$ *and* $\mathrm{dist}(\mathcal{L}_0, j) = l$ *for any* $j \in \mathcal{L}_l$.

For $\mathcal{L}_0 = \{i\}$ we identify the level structure of i with that of $\{i\}$. We will not present an algorithm for computing the level structure of a given node, since this is a well-known technique, see e.g. [11, 6, 8]. We only note that this can be done in $\mathcal{O}(\#E)$ steps. Why do we calculate this level set? It is easy to see that if $\mathcal{L}_0, \dots, \mathcal{L}_k$ is the level structure of node i, then for any $j \in \mathcal{L}_l$ we have

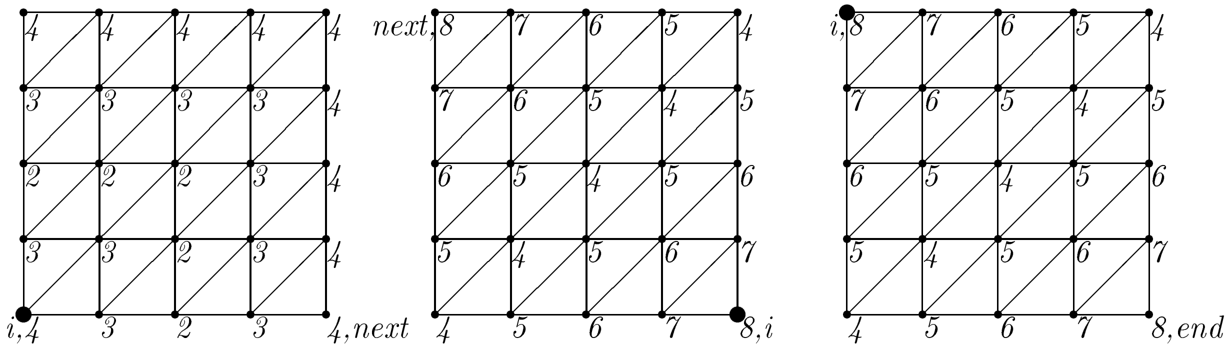
$$\max_{q \in V} \mathrm{dist}(q, j) \geqslant \max\{l, k - l\}, \tag{61}$$

since any node $j \in \mathcal{L}_l$ satisfies $\mathrm{dist}(i, j) = l$ and in addition we have that $\mathrm{dist}(r, j) = k - l$ for any node $r \in \mathcal{L}_k$. From this it follows that the level structure of a node i gives us an upper bound for the measure $\max_{q \in V} \mathrm{dist}(q, j)$ for any node $j \in V$ and thus indicates where central nodes may be. To calculate a suitable estimate for $\max_{q \in V} \mathrm{dist}(q, j)$, we should choose a node which is as far away as possible from the center of the graph. Thus after taking an arbitrary initial node i for calculating the level structure of i the next test node should be one of $\mathcal{L}_k$, since these nodes are at least far away from i. This technique is also well–known [12] and such nodes are called pseudoperiphal nodes. As consequence we obtain the following algorithm for calculating an estimate for $\max_{q \in V} \mathrm{dist}(q, j)$. For later purposes we restrict the collection of test nodes to a certain subset C which we will later specify more precisely. Initially we may assume that $C = V$.

Algorithm 20 *Let* $G = (V, E)$ *be an undirected graph and let* $C \subseteq V$. *Let* d *be a vector of same size as* $\#C$ *and suppose that initially we have* $d(j) = 0$ *for any* $j \in C$.
Choose a test node $i \in C$.
do
  Compute the level structure $\mathcal{L}_0, \dots, \mathcal{L}_k$ *of node* i.
  For any $j \in C \cap \mathcal{L}_l$ *update* $d(j)$ *by* $\max\{d(j), l, k - l\}$.
  $q_{old} = q$, *choose* $i \in C \cap \mathcal{L}_q$ *with* $q > q_{old}$ *(if* i *does not exist, leave* i *unchanged).*
while *node* i *has not yet been used as test node.*

Using $C = V$, this algorithm computes an upper bound for $\max_{q \in V} \mathrm{dist}(q, j)$. The algorithm in theory may need $\#C$ steps but it stops when q does not increase any more. q is the maximal distance from node $i \in C$ to any other node $j \in C$. So if q does not increase any more, then i is an almost pseudoperiphal node, i.e., a node in C which is as far away as possible from all the other nodes in C.

Example 21 *Consider the graph from Example 18 and use as first test node the node in the lower left corner. Starting with* $C = V$, *Algorithm 20 calculates the following values for* d



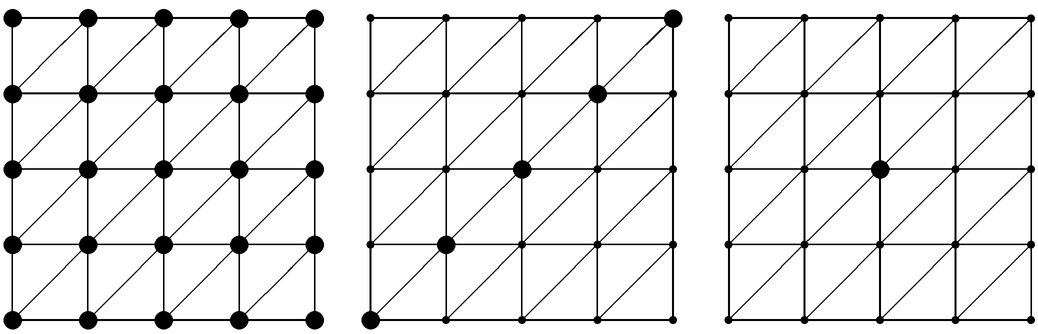
The nodes on the diagonal from the lower left corner to the upper right corner have minimal d.

The example shows that possibly more than one node may stay after the algorithm is completed. Clearly Algorithm 20 only computes an estimate for $\max_{q\in V} \operatorname{dist}(q,j)$. Although in this example the chosen nodes are correct with respect to this measure we would like to have a node that almost minimizes $\sum_{q\in V} \operatorname{dist}(q,j)$. To achieve this we replace C by the set of nodes with minimal positive d and repeat Algorithm 20.

Algorithm 22 *Let $G=(V,E)$ be an undirected graph and set $C=V$.*
do
Apply Algorithm 20.
Replace C by the set of nodes with minimal positive d.
while *C has changed*

Again we illustrate this by an example

Example 23 *Consider the graph from Example 18. We illustrate the set C.*

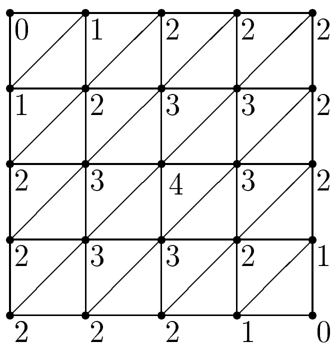


By successively repeating Algorithm 20 with a shrinking set C we obtain a node j which not only has a small value for $\max_{q\in V} \operatorname{dist}(q,j)$ but recursively we obtain that $\max_{q\in C} \operatorname{dist}(q,j)$ for any new choice of C. Even if $\sum_{q\in V} \operatorname{dist}(q,j)$ is not minimized we have calculated a node or possibly some nodes for which $\sum_{q\in V} \operatorname{dist}(q,j)$ must be small.

Using Algorithm 22 we usually select a very small amount of nodes (possibly only one) which are located in the center of the graph. Recall that the center of the graph and an algorithm to compute this center has been introduced in order to compute an approximate eigenvector x for eigenvalue problem (47). The main heuristic argument is that typically the importance of the nodes in the graph increases the closer they are located near the center. And computing x such that $Mx = e_i$, where e_i is zero outside the center of the graph typically ends up in a vector x such that $\|Mx\| \ll x$. Based on these central nodes we define an initial guess $x^{(0)}$ for an approximate eigenvector x for the eigenvalue problem (47) by computing the level structure of our set of central nodes. If, say $\mathcal{L}_0, \ldots, \mathcal{L}_k$ is the corresponding level structure with $\mathcal{L}_0$ denoting the central nodes and $\mathcal{L}_l$ the nodes with distance l from $\mathcal{L}_0$ we define our initial guess $x^{(0)}$ for the computation of an approximate eigenvector x via

$$x_j^{(0)} = (k-l)/k, \forall j \in L_l. \tag{62}$$

Example 24 *Consider the graph from Example 18. We illustrate the values of* $4 \cdot x$.



Now with this initial guess $x^{(0)}$ for x we may start then an iterative method for the solution of $Mx = e_i$. In principle we could try to solve $Mx = e_i$ with the *cg*–algorithm [13, 4]. Here we use a block version of the *cg*–algorithm.

Algorithm 25 *Let* $A \in \mathbb{R}^{n,n}$ *be symmetric positive definite,* $X, B \in \mathbb{R}^{n,l}$, $R = B - AX$, $\rho = 0$.
for $i = 1, 2, \ldots$
  $\rho_{old} = \rho$, $\rho = R^\top R$
  if $i \equiv 1$
    $P = R$
  else
    $\beta = \rho_{old}^{-1}\rho$, $P = R + P\beta$
  end
  $\sigma = P^\top AP$, $\alpha = \sigma^{-1}\rho$
  $X = X + P\alpha$, $R = R - AP\alpha$
end

Algorithm 25 is a 'block'–*cg* algorithm and for $l = 1$ it reduces to the well–known standard *cg* method. See e.g. [27] for the block version and [22] for an analysis of this kind of method.

In principle we could try to solve $Mx = e_i$ with our initial guess $x^{(0)}$. An ill–conditioned matrix M will slow down this method. But what we need is not x itself but any multiple of

x suffices too, since later we may scale our solution such that $\|x\| = 1$. The corresponding Krylov subspace $\mathcal{K}_l(A, r) = \text{span}\{r, Ar, \ldots, A^l r\}$, where $r = b - Ax^{(0}$ which essentially determines the solution computed by the *cg*–algorithm [13] does not detect whether we use $x^{(0)}$ or a multiple $\lambda x^{(0)}$ as initial guess. For example if we would use $x^{(0)} = 2x$ as initial guess or $x^{(0)} = 0$ then the Krylov space $\mathcal{K}_l(A, r)$ will be the same and the number of iterations will typically not essentially change. But this may be bad for our problem here, since we only expect that several components of our initial choice $x^{(0)}$ for x in (62) give the correct direction. For this reason we employ a block version of the *cg*–method. For the solution of $Mx = \lambda e_i$ where λ is some nonzero parameter which we do not know in advance, we have

$$\mathcal{K}_l(A, \lambda e_i - Ax^{(0)}) \subseteq \mathcal{K}_l(A, \left[e_i - Ax^{(0)}, e_i\right]). \tag{63}$$

The latter does not depend on λ and $\mathcal{K}_l(A, \left[e_i - Ax^{(0)}, e_i\right])$ is the corresponding Krylov subspace for the block *cg*–algorithm where $l = 2$, $B = [e_i, e_i]$ and initial guess $X = \left[x^{(0)}, 0\right]$. It immediately follows that the block cg–algorithm with these values for B and X is independent on the scaling applied to x and therefore we may expect that convergence is much faster. We confirm this in our numerical examples.

Note that the analysis in [22] essentially shows that some isolated eigenvalues at the boundary of the spectrum do not effect the convergence behaviour if the block size l of the block *cg*–method is adapted to the number of isolated extremal eigenvalues. In our experiments we do not have one or two isolated eigenvalues but nevertheless already after a few number of steps the computed solution often gives a good approximate eigenvector. Possibly an explanation for this effect may be that our initial guess was already close enough to a small invariant subspace containing the solution and so most of the eigenvalues from the spectrum where not visible any more. To obtain the desired solution from the block *cg*–process we have to replace the $n \times 2$ matrix X by Xv, where $v \in \mathbb{R}^2$ has to be suitably chosen. In principle we could choose v such that $\|Xv - x\| \stackrel{!}{=} \min$ in some norm. In our case the energy norm induced by $A^{1/2}$ seems appropriate, since the minimization of

$$f(v) = \|A^{1/2}(Xv - x)\|_2^2 = v^\top X^\top AXv - 2v^\top X^\top b + \underbrace{x^\top b}_{constant} \tag{64}$$

does not explicitly require knowledge of the solution x. From (64) we can easily calculate $v \in \mathbb{R}^2$ by solving a 2×2 linear system. More elegant than (64) would be to choose v such that

$$g(v) = \frac{v^\top X^\top AXv}{v^\top X^\top Xv} \stackrel{!}{=} \min, \tag{65}$$

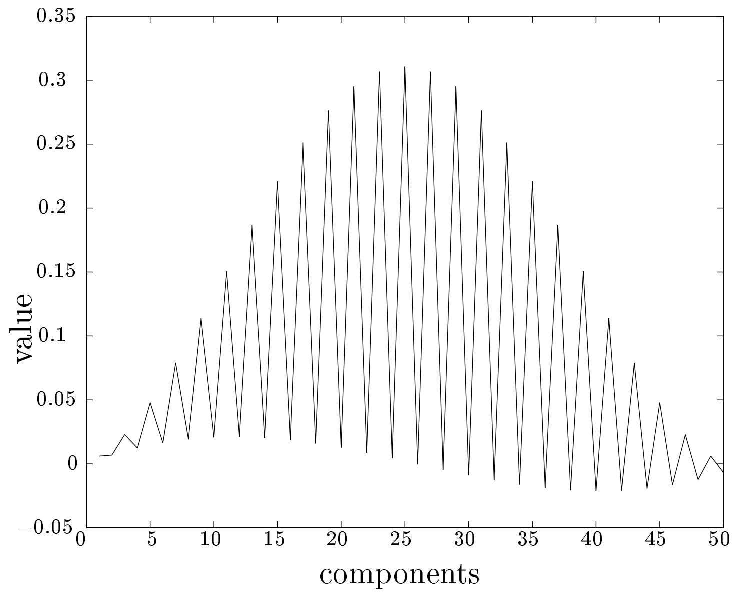
since we are interested in computing an approximate eigenvector of A associated with the smallest eigenvalue. Again one can immediately compute v from (65) by solving a 2×2 eigenvalue problem.

Example 26 *Consider the block matrix*

$$A = \begin{bmatrix} D & B & & O \\ B^\top & D & \ddots & \\ & \ddots & \ddots & B \\ O & & B^\top & D \end{bmatrix}, \quad D = \begin{bmatrix} 786432 & 0 \\ 0 & 256 \end{bmatrix}, \quad B = \begin{bmatrix} -393216 & 6144 \\ -6144 & 64 \end{bmatrix},$$

where A has size 50 *is scaled to* $D^{1/2}AD^{1/2}$ *which has unit diagonal. This matrix is essentially the matrix* ***LANPRO/NOS1*** *from the Harwell–Boeing collection [9]. The center of the underlying graph is in this case* $\{25, 26\}$*. Using MATLAB [1] the exact eigenvector associated with the smallest eigenvalue is given in Figure 2. We can compare the two*

Figure 2: **exact eigenvector**



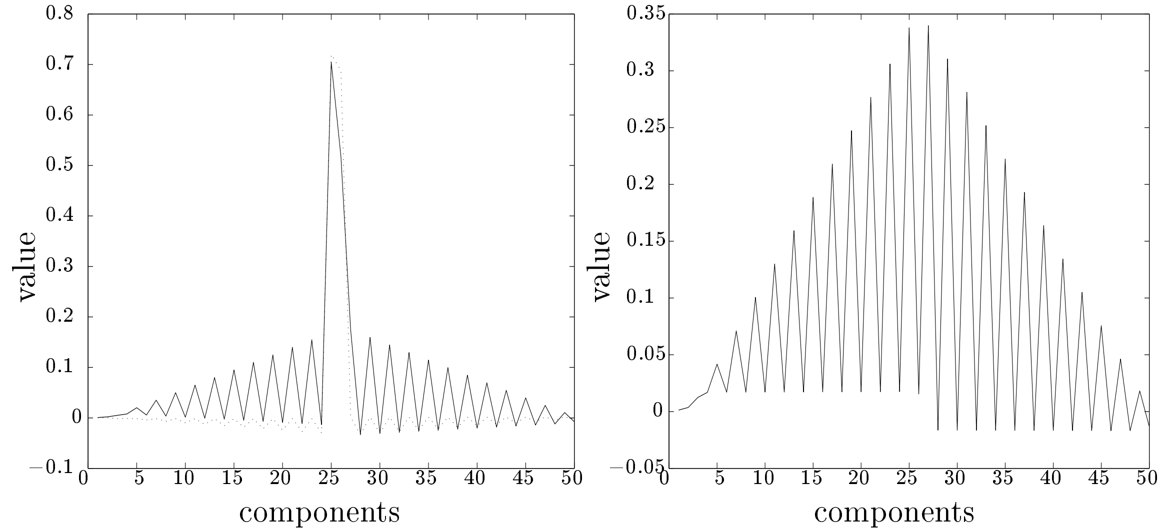
columns of X with the solution obtained via (65). The first column of X is depicted in the left diagram of Figures 3, 4, 5 with a solid line while the second column of X is depicted in the same figures with a dotted line. For comparison the solution obtained from (65) is plotted on the right side of these figures. Figures 3, 4, 5 correspond to the iteration steps $1, 10$ *and* 20*, respectively. We observe in Figures 3, 4, 5 that the two different solutions strongly differ from each other and also strongly differ from the exact solution which is in this case almost the eigenvector in Figure 2. In contrast to this behaviour the approximate eigenvector with respect to (65) almost does not change anymore after the first step. Clearly this approximate eigenvector still differs from the exact eigenvector in Figure 2 but the difference is small and the trend is already correct after the first step.*

4.6 The Pivoting Process

It remains to describe in detail the pivoting based on the approximate eigenvector or the eigenvector independent criterion.

Clearly the best we can do is to locally maximize Δ or $\hat{\Delta}$ in the inequalities (48), (48) to obtain a feasible coarse grid matrix $P = LV$ for $M^{(1)}$ in (6) or $M^{(2)}$ in (7). Since by (37) $E\Pi_1 = VR_{11}$ clearly one does not choose the full matrix V but instead one chooses $E\Pi_1$ instead, since the columns of this matrix span the same space, but they are sparse. So $P = LE\Pi_1$ will be the equivalent coarse grid matrix. As shown in Lemma 13 maximizing Δ,

Figure 3: **Step 1, approximate solutions/approximate eigenvector**



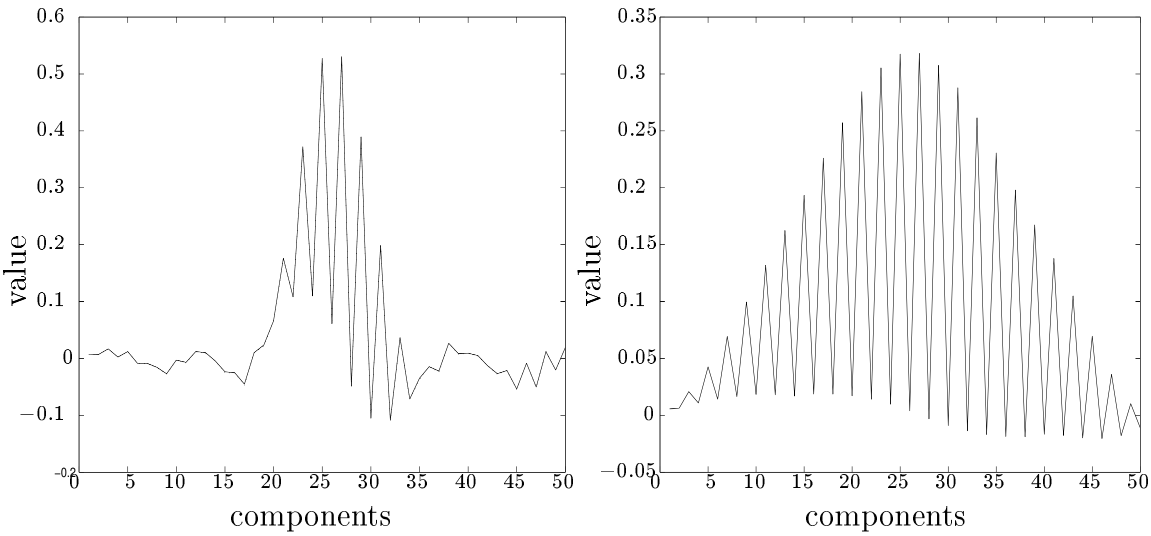
$\hat{\Delta}$ can be simplified to using the eigenvector x of M associated with the smallest eigenvalue and in this case successively maximizing $\tau = \|V^\top x\|_2^2$ is equivalent to maximizing (48), (49) as shown by Lemma 14. Since we cannot expect to precisely compute x in general, we have introduced an eigenvector independent indicator by using the residual in Lemma 15. The interesting observation is that even if one does not know the exact eigenvector, the residual in (55) for the eigenvalue problem (54) is independent of the eigenvector up to a scalar! Analogous properties hold for the other eigenvalue problems. Instead of choosing the next pivot such that $V^\top x$ is locally maximized we can choose the pivot with respect to largest component of the residual. Only to validate this choice we use $V^\top x$. This extremely reduces the dependence of an inaccurately computed approximate eigenvector x.

To summarize this double key choice for finding the next coarse grid node we describe an abstract algorithm that defines the pivoting strategy for Algorithm 12. Note that clearly to increase $V^\top x$ in the 2–norm we have to consider the normalized components of $W^\top x$ with W from (37) and consider its components or equivalently, since $TE\Pi_2 = W$, where $T = I - VV^\top M$ we have to consider $x^\top(E\Pi_2 - VV^\top ME\Pi_2)$. Note that the computation of $x^\top$ can be extremely simplified if one initially computes $x^\top E$ and then updates from step to step this expression by $x^\top TE$. This is only a rank–1 correction. If $x^\top TE$ is given at step $p-1$, then the updated vector will be $x^\top(I - \hat{v}\hat{v}^\top M)TE$. In the Gram–Schmidt process $\hat{v}$ will just be $TE(:,\pi(p-1))$ divided by $R(p-1,\pi(p-1))$. It follows that

$$\begin{aligned} x^\top(I-\hat{v}\hat{v}^\top M)TE(:,\pi(p{:}n)) &= x^\top TE(:,\pi(p{:}n)) - x^\top\hat{v}\hat{v}^\top MTE(:,\pi(p{:}n)) \\ &= x^\top TE(:,\pi(p{:}n)) - x^\top\hat{v}\hat{v}^\top ME(:,\pi(p{:}n)) \\ &= x^\top TE(:,\pi(p{:}n)) - \frac{x^\top TE(:,\pi(p-1))}{R(p-1,\pi(p-1))}R(p-1,\pi(p{:}n)). \end{aligned}$$

For the indicator we simply take q from Algorithm 12 since in Lemma 15 the vector $v - MVV^\top v$ is dominated by $MVV^\top v$. This is a consequence of $I = V^\top MV \ll V^\top V$

Figure 4: **Step 10, approximate solutions/approximate eigenvector**

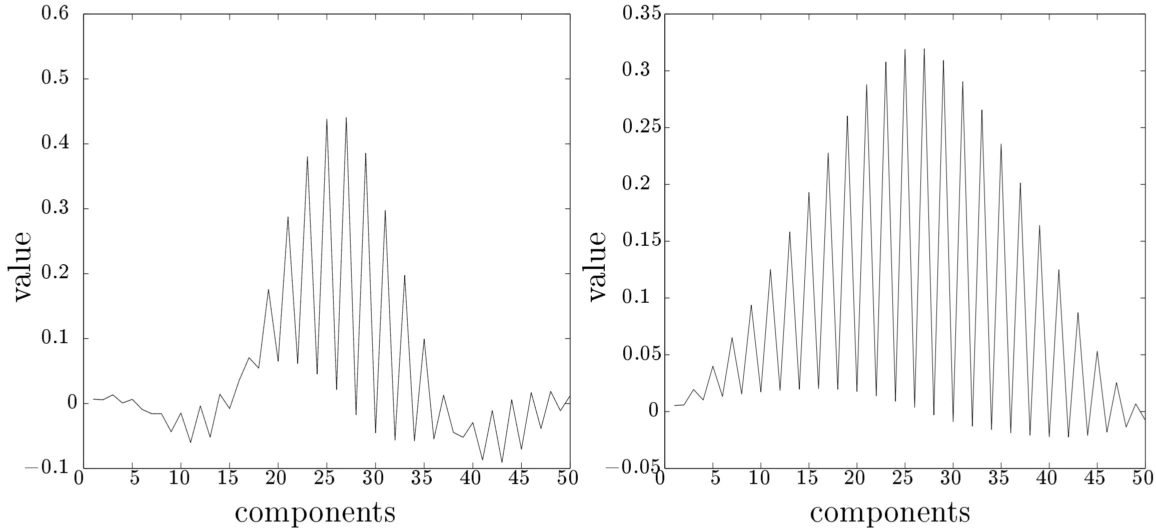


for ill–conditioned problems. Since we try to improve our operator step by step using the coarsening process $V^\top v$ itself will be characterized by its last component and this gives (up to a scalar) our choice q.

Algorithm 27 (Pivoting Strategy for the Gram–Schmidt Algorithm)
Let $x \in \mathbb{R}^n$ *be an approximate eigenvector of* M *associated with its smallest eigenvalue. Let* $tol \in (0, 1)$, *e.g.* $tol = 0.1$ *and suppose that initially* $\alpha = x^\top E$ *is computed.*
if $p \equiv 1$
    Choose p
else
    Update $\alpha(\pi(p-1{:}n))$ *by*
    $\alpha(\pi(p-1)) = \alpha(\pi(p-1))/R(p-1, \pi(p-1))$
    $\alpha(\pi(p{:}n)) = \alpha(\pi(p{:}n)) - \alpha(\pi(p-1))R(p-1, \pi(p{:}n))$
    Compute $\tau(\pi(p{:}n)) = \alpha(\pi(p{:}n))^2/\nu(\pi(p{:}n))$ *for any* $k = p, \ldots, n$.
    Find those k, *for which* $|q_k|$ *and* τ_k *are big.*
    Among those k *take the one for which* τ_k *is maximal.*
    If such k *does not exist take a node for which at least* τ_k *is big and the number of coarse grid points in its neighbourhood is minimal.*
end
Interchange $\pi(p)$ *and* $\pi(k)$.

The strategy to define k is left quite abstract in Algorithm 27. In principle one could check whether those k with large $|q_k|$ also have large τ_k or not. As long as both criteria agree in their choice of k, it is only natural to continue the coarsening process close to the node where it stopped in the step before. But once the criteria disagree, we have to find a new node. In this case it seems to be sensible not to be too close to the current coarse

Figure 5: **Step 20, approximate solutions/approximate eigenvector**



grid nodes. In principle $|q|$ only gives a direction where to find coarse grid nodes, since it demonstrates the difference between x and Tx. But if $|q|$ only indicates a change in x it will not necessarily give large values for τ. In this case the coarsening process should move into another direction. This is the main idea behind the choice for k.

Finally one additional restriction is necessary among those nodes which are selected to become coarse grid nodes. Since the best we can do is to work ideally with the accurately computed eigenvector x, for problems which are almost block diagonal τ as well as q can fail as estimates for the solution of eigenvalue problems (47),(48). Due to rounding and approximation errors it may happen that both criteria suggest only coarse grid nodes which are related to one (almost) diagonal block. To prevent this coarsening into the wrong direction some of the fine grid nodes which have been selected not to become coarse grid nodes have to be taken away from the set of remaining nodes. In this case they obviously cannot be chosen as coarse grid nodes. A simple criterion to deactivate several nodes is the hypothesis that if we once have locally defined coarse grid nodes, then the remaining nodes in the neighbourhood are no longer needed as coarse grid nodes. To end up in a deterministic strategy, we disable those nodes k for which k and all its neighbours are connected to coarse grid nodes.

5 Numerical results

In this section we illustrate the effectiveness of the algorithm. Our computations are done in MATLAB 5.2 [1] on a LINUX PC with a PENTIUM 2/350 processor. Here we restrict ourselves to the coarsening process generated by the algorithm and show the condition number and sparsity of the generated QR decomposition.

We consider three types of approximate inverses for the initial matrix. These are a) the diagonal of the matrix as preconditioner, b) the factored sparse approximate inverse suggested by [19, 20] with the same sparsity pattern as the initial matrix and c) a preconditioner that we call Jacobi–squared Preconditioner. For this latter preconditioner we combined Jacobi's eigenvalue algorithm and Jacobi preconditioning. For a set of $n/2$ disjoint pairs $\{i,j\} \subset \{1,\dots,n\}$ we apply a Jacobi rotation to diagonalize $\begin{pmatrix} a_{ii} & a_{ij} \\ a_{ji} & a_{jj} \end{pmatrix}$ i.e., a_{ij}, a_{ji} are annihilated. For this transformed system we then use diagonal preconditioning.

In the multigrid process we always use diagonal preconditioning on the coarser levels.

Example 28 *Consider the matrix*

$$A = \begin{bmatrix} T & -e_n & \mathrm{O} \\ -e_n^\top & 1+\alpha & -\alpha e_1^\top \\ \mathrm{O} & -\alpha e_1 & \alpha T \end{bmatrix}, \text{ where } T = \begin{bmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{bmatrix} \in \mathbb{R}^{n,n}.$$

for $\alpha = 1, 100$. This example can be read as discretization of the problem $-au'' = f$ in $[0,1]$ with $a(x) = 1$ for $x \in [0, .5]$ and $a(x) = \alpha$ for $x \in [.5, 1]$.

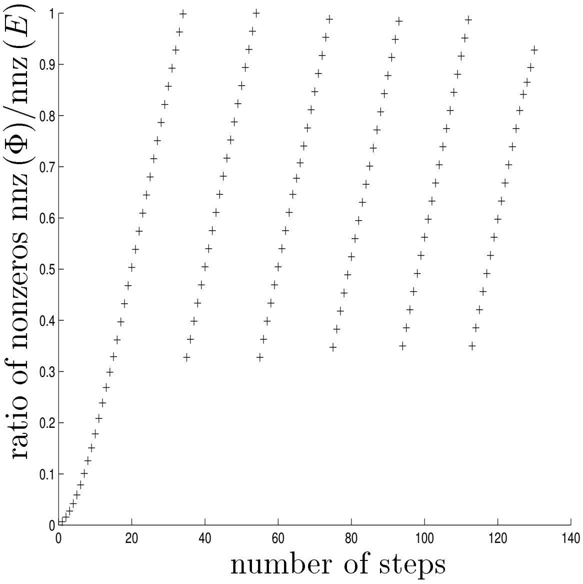
First we demonstrate some simple techniques of discarding and recovering columns of Φ from (45). The graph of the preconditioned matrix corresponds to a chain and we will illustrate those nodes which have been selected to be discarded if necessary. There are two classes of nodes to be discarded. The first class consists of those computed columns of Φ, which have not been touched for the last five steps of the QR factorization and their neighbours are disabled. The second class of nodes only consist of those nodes which have not been touched for the last five steps but have more than average disabled nodes and more thena average coarse grid nodes in their neighbourhood. In figure 6 the ratio of fill–in of Φ with respect to the fill–in of E. Recall that columns of Φ are discarded once this ratio is bigger than 1. Note that without discarding some columns of Φ the memory requirement for Φ will be approximately three times the memory required for E.

In Figure 7 we will give some snapshots of the graph for $n = 255$ and starting node $\pi(1) = 128$ during the coarsening process. By $\circ$ we denote the regular coarse grid nodes and by $\blacksquare$ those nodes for which the corresponding columns of Φ have been discarded. Here the nodes are placed on a circle with a gap at the extremal right point. Since the graph of A is a chain, thisgives a compact representation of the graph. So the starting point $p = 128$ will be at the extremal left point of the circle. The nodes that will be automatically detected in this case are the even nodes. This confirms studies in [18], where this was suggested a priori. In this example $\alpha = 1$ was used, but for the other values of α the algorithm behaves similar.

In Figure 8 we consider the same problem but this time with a tridiagonal sparse approximate inverse, starting with the sparsity of Q. In Figure 9) we again depict the discarded columns of Φ and we see that this time blocks consisting of two neighbouring nodes are used as coarse grid nodes and the next two node are left out.

To study the automatic coarsening process we begin with diagonal preconditioning for two kinds of coefficient jumps. We show some snapshots of the coarsening process, i.e., we show

Figure 6: **Memory requirement for Φ**



the graph of the reduced linear systems A_k from Definition 4 for $n = 511$. For level 1,2 the graph is too dense, the initial system is a chain and the first level essentially consists of every second node up to some noise. In Figure 10 we show the grids for levels 3–6.

Next we consider the same system with tridiagonal preconditioning. In Figure 11 the coarsening snapshots of the graph are given for level 3 and 4. Finally we consider Jacobi–squared preconditioning. In Figure 24 we illustrate the levels 2–5. On level 1 only every fourth nodes has been taken as coarse grid node!

All figures show that an effective reduction of the number of nodes is achieved by the coarsening algorithm. Clearly depending on the quality of the initial preconditioner as a smoother more or less nodes will be required on coarser levels. More interesting then the graphical visualization of the coarser grids is the size of the system and the related fill–in for the different levels. This gives information on how much the system is reduced in each step of the coarsening process.

Figure 7: **Coarse grid nodes, active and discarded**

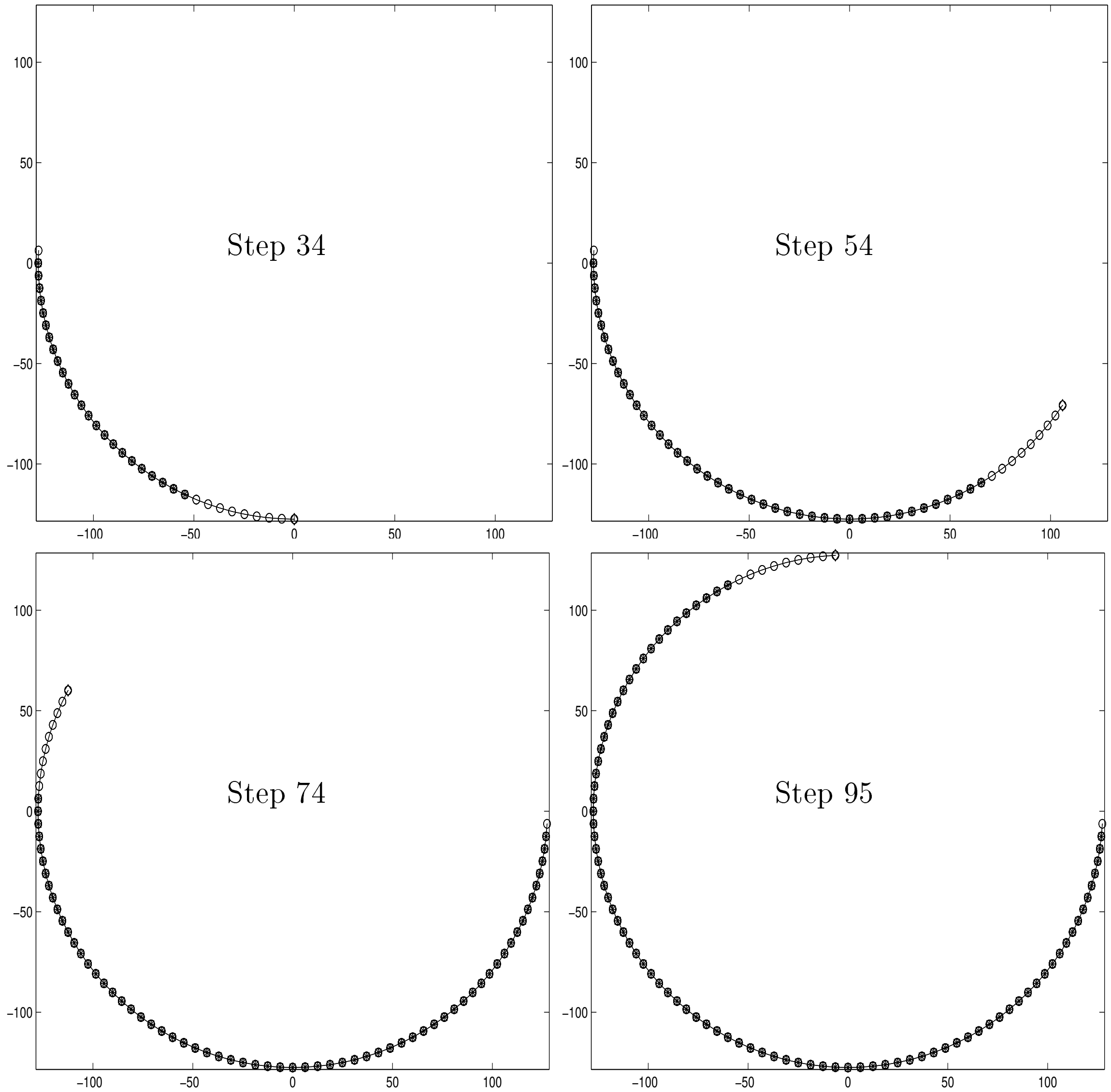


Figure 8: **Memory requirement for Φ**

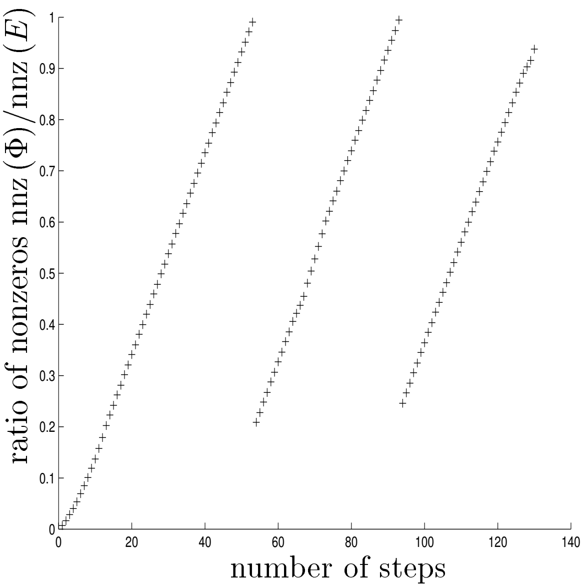


Figure 9: **Coarse grid nodes, active and discarded**

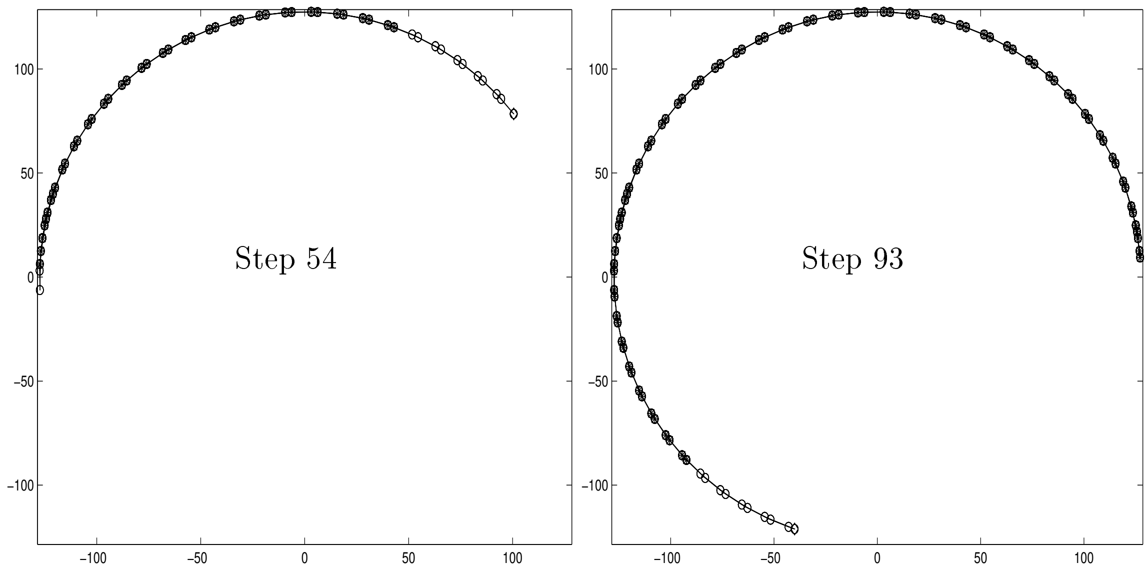


Figure 10: **Coarsening Snapshots $\alpha = 1$, diagonal preconditioning, level 3–6**

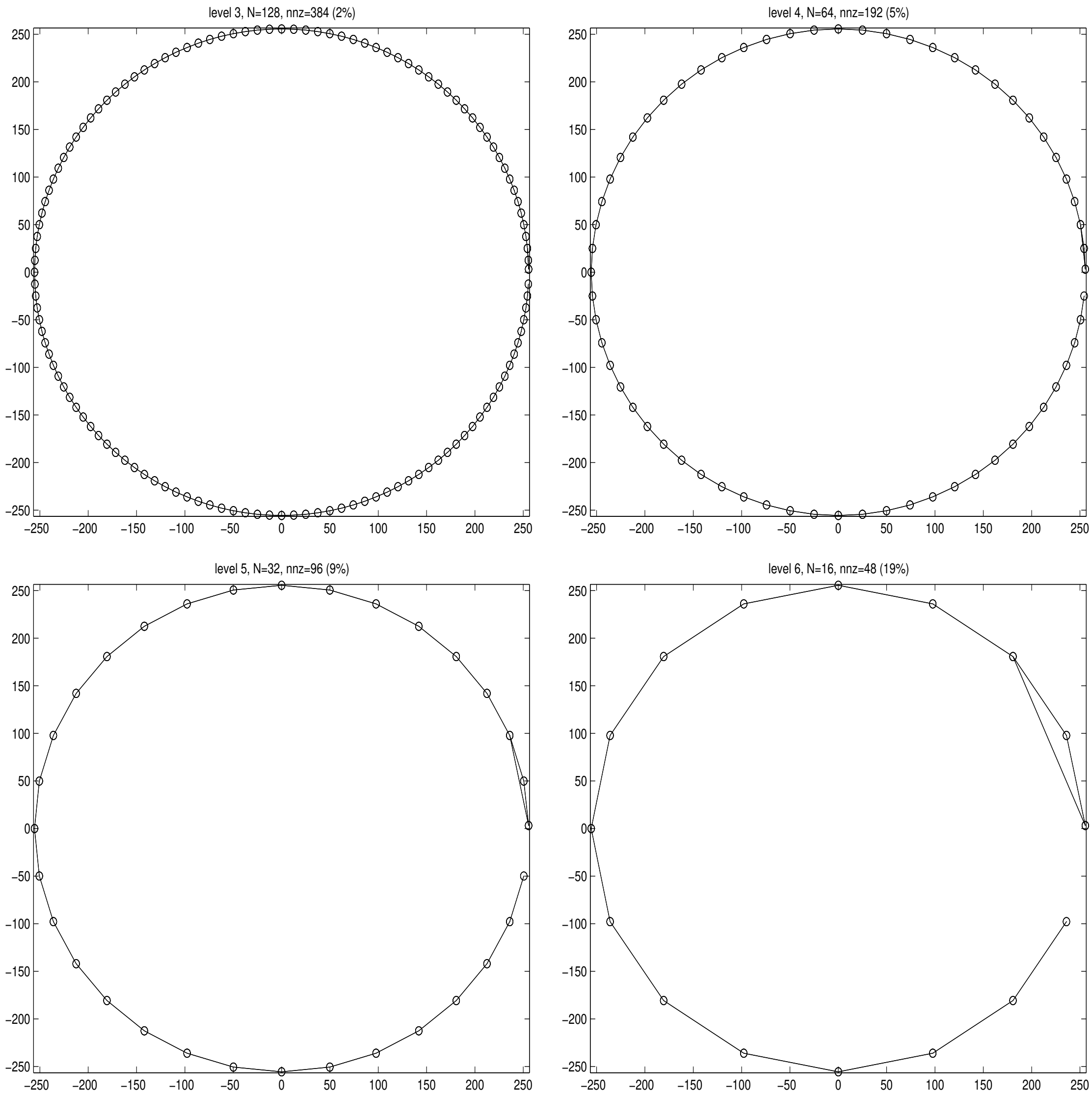
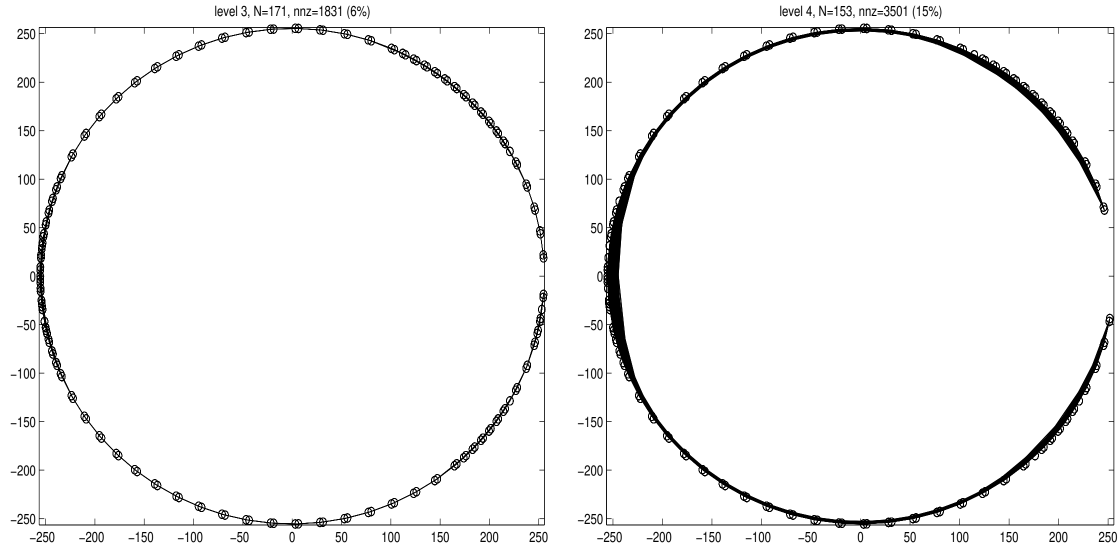


Figure 11: **Coarsening Snapshots $\alpha = 1$, tridiagonal preconditioning, level 3–4**

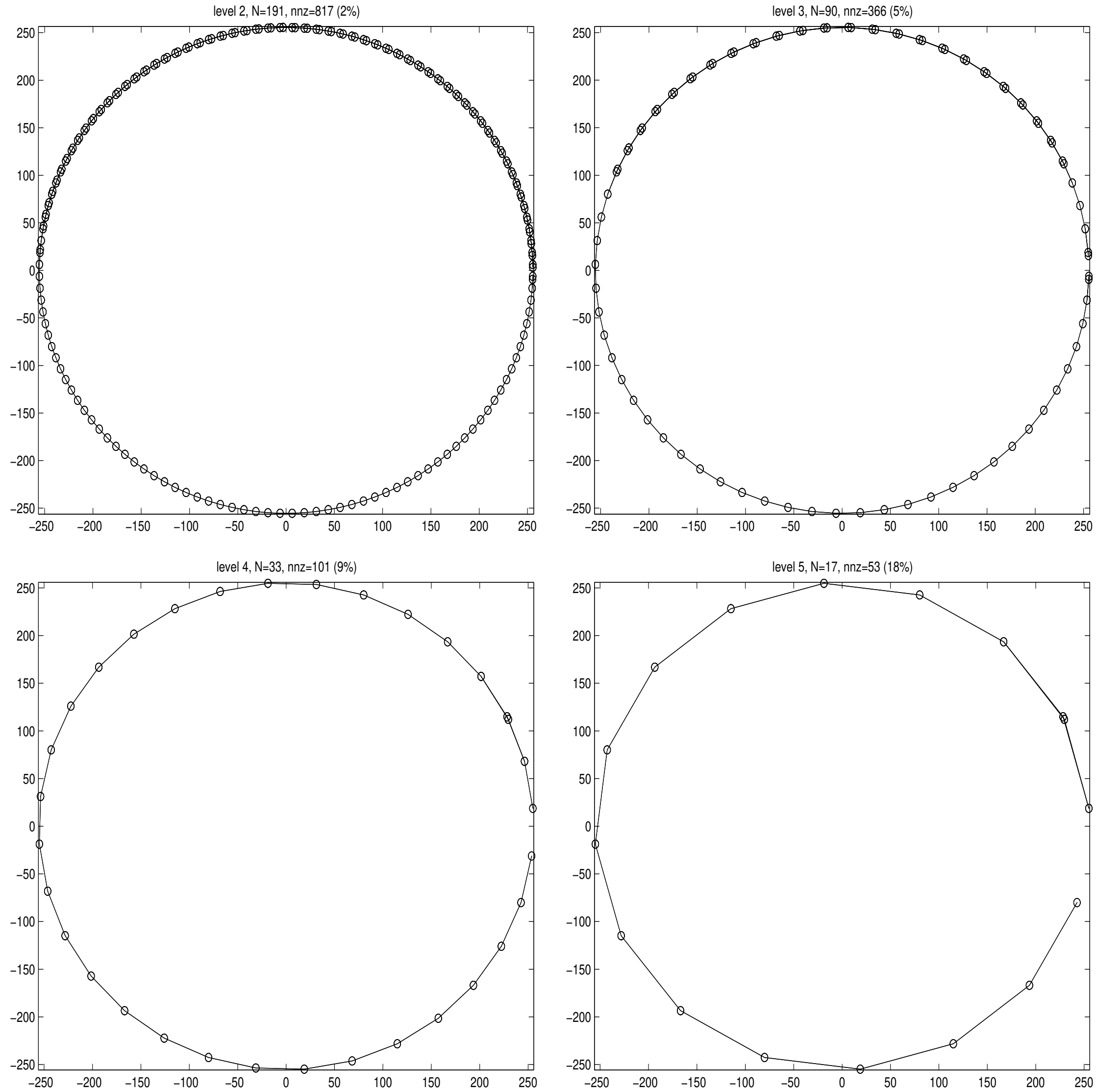


					Diagonal Preconditioning, $\alpha = 1$					
initial size	*Levels, size and fill–in (%)* 1	2	3	4	5	6	7	8	9	10
2047	2047, <1	1025, <1	513, 1	257, 1	130, 2	66, 5	39, 11	28, 23	—	—
1023	1023, <1	512, <1	256, 1	128, 2	64, 5	32, 9	16, 19	—	—	—
511	511, 1	256, 1	128, 2	64, 5	32, 9	16, 19	—	—	—	—
					Tridiagonal Preconditioning, $\alpha = 1$					
2047	2047, <1	1028, 1	512, 1	198, 3	84, 6	72, 14	—	—	—	—
1023	1023, <1	525, 2	329, 3	281, 7	—	—	—	—	—	—
511	511, 1	267, 2	171, 6	153, 15	—	—	—	—	—	—
					Jacobi–squared Preconditioning, $\alpha = 1$					
2047	2047, <1	586, 1	282, 1	129, 2	65, 5	34, 9	18, 18	—	—	—
1023	1023, <1	383, 1	184, 3	86, 5	33, 9	16, 20	—	—	—	—
511	511, 1	191, 3	90, 5	33, 10	17, 19	—	—	—	—	—

An interesting observations is that in the case of diagonal preconditioning precisely the grid which is used for geometric multigrid method is obtained by the coarsening process. But the other two choices of coarsening also give a moderate reduction of the system size. In particular the Jacobi–squared preconditioning shows an extreme reduction on the initial level. Note that Jacobi–squared is only used on the initial grid, on all other grids diagonal preconditioning is used.

We are also interested in the condition number of the preconditioned system $M_l^{(1)}A, M_l^{(2)}A$ compared with the initial system A and the initial system M. Since the condition number only partially reflects the improvement we also present the number of cg iteration for the

Figure 12: **Coarsening Snapshots $\alpha = 1$, Jacobi–squared preconditioning, level 2–5**



system preconditioned with the multilevel preconditioner.

Diagonal Preconditioning, $\alpha = 1$								
size	*condition number*				*CG steps*			
	A	M	$M_l^{(1)}A$	$M_l^{(2)}A$	A	M	$M_l^{(1)}A$	$M_l^{(2)}A$
2047	—	—	—	—	2047	2047	80	37
1023	$5.2 \cdot 10^5$	$5.2 \cdot 10^5$	$3.2 \cdot 10^1$	$2.2 \cdot 10^0$	1023	1023	24	9
511	$1.3 \cdot 10^5$	$1.3 \cdot 10^5$	$1.4 \cdot 10^1$	$2.2 \cdot 10^0$	511	511	22	9
Tridiagonal Preconditioning, $\alpha = 1$								
2047	—	—	—	—	2047	1094	218	94
1023	$5.2 \cdot 10^5$	$1.7 \cdot 10^5$	$9.8 \cdot 10^3$	$1.8 \cdot 10^2$	1023	551	122	55
511	$1.3 \cdot 10^5$	$4.4 \cdot 10^4$	$1.3 \cdot 10^3$	$2.7 \cdot 10^2$	511	280	66	30
Jacobi–squared Preconditioning, $\alpha = 1$								
2047	—	—	—	—	2047	1400	42	17
1023	$5.2 \cdot 10^5$	$3.4 \cdot 10^5$	$5.9 \cdot 10^1$	$9.8 \cdot 10^0$	1023	711	39	16
511	$1.3 \cdot 10^5$	$1.0 \cdot 10^5$	$4.6 \cdot 10^1$	$7.9 \cdot 10^0$	511	364	34	15

The results show that the automatic coarsening process has generated a coarse grid hierarchy that is as multilevel method much superior to the initial sparse approximate inverse preconditioner.

Our next set of tests are for the case of a coefficient jump with $\alpha = 100$

In Figure 13 coarsening snapshots for levels 3–6 and diagonal preconditioning are presented. For the sparse approximate inverse with the same sparsity as the initial matrix the results are shown in Figure 14 and for Jacobi–squared in Figure 15. Figures13, 14, 15 look similar to the situation of $\alpha = 1$*, except that there are more nodes taken in some parts of the graph, i.e., the coarse grid looks less regular then in the case* $\alpha = 1$*.*

Figure 14 shows the tridiagonal case for level 3,4. Figure 15 considers the case of Jacobi–squared. Like in the case $\alpha = 1$ *on the initial level only every fourth node is taken. In Figures 13, 14 and 15 the reduction in the matrix size and the fill–in during the coarsening*

Figure 13: **Coarsening Snapshots $\alpha = 100$, diagonal preconditioning, level 3–6**

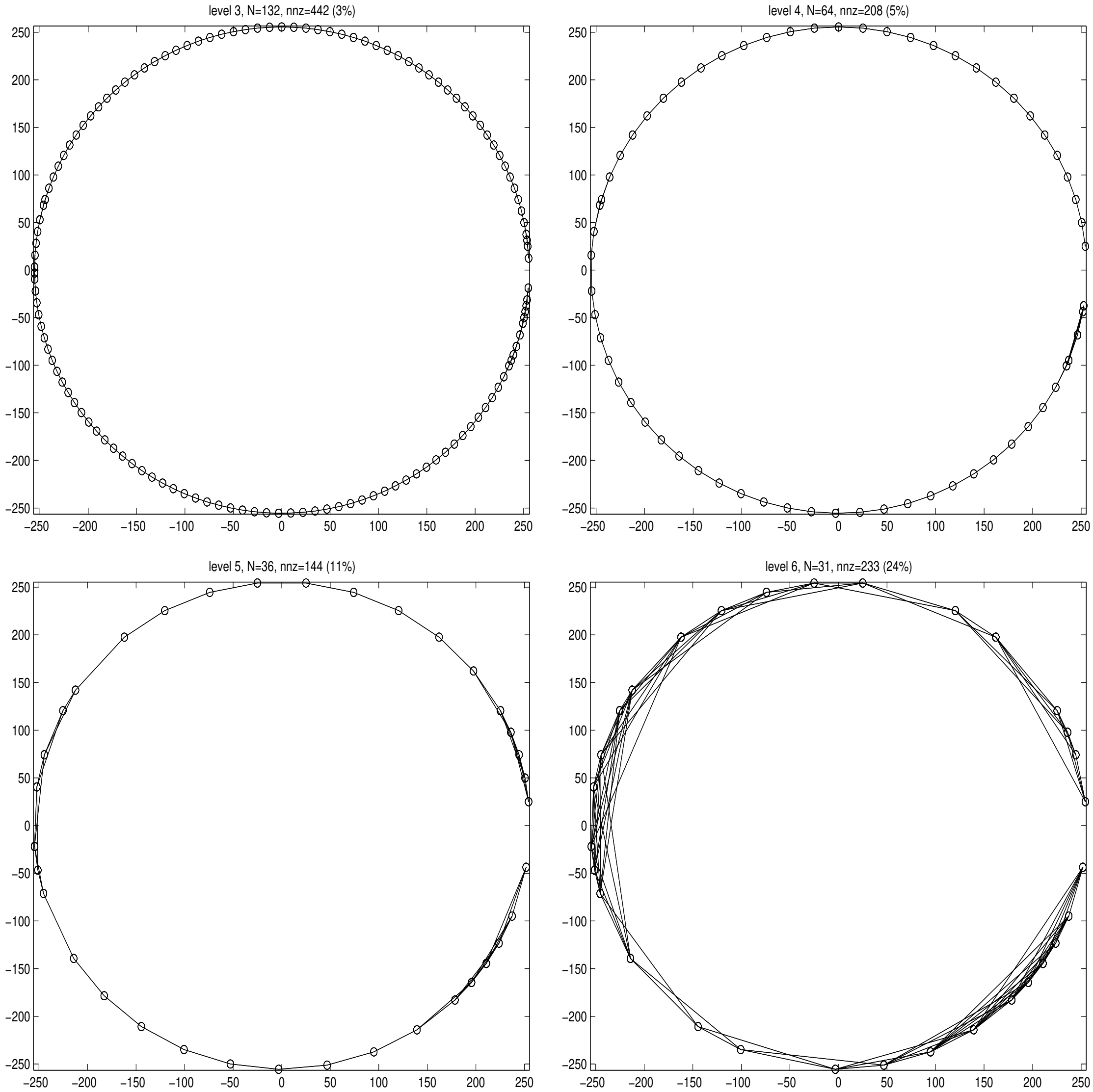
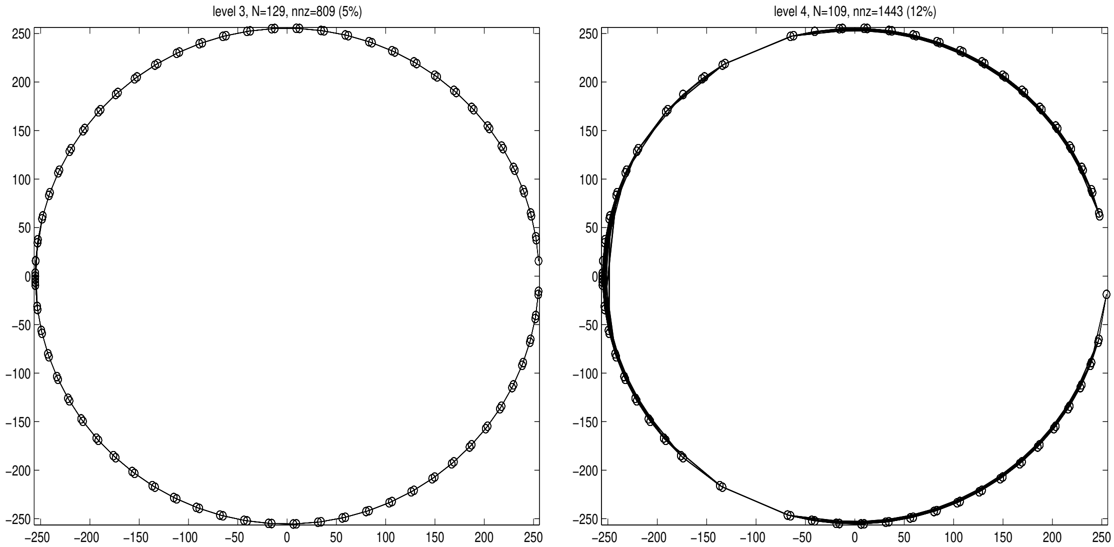


Figure 14: **Coarsening Snapshots** $\alpha = 100$**, tridiagonal preconditioning, level 3–4**



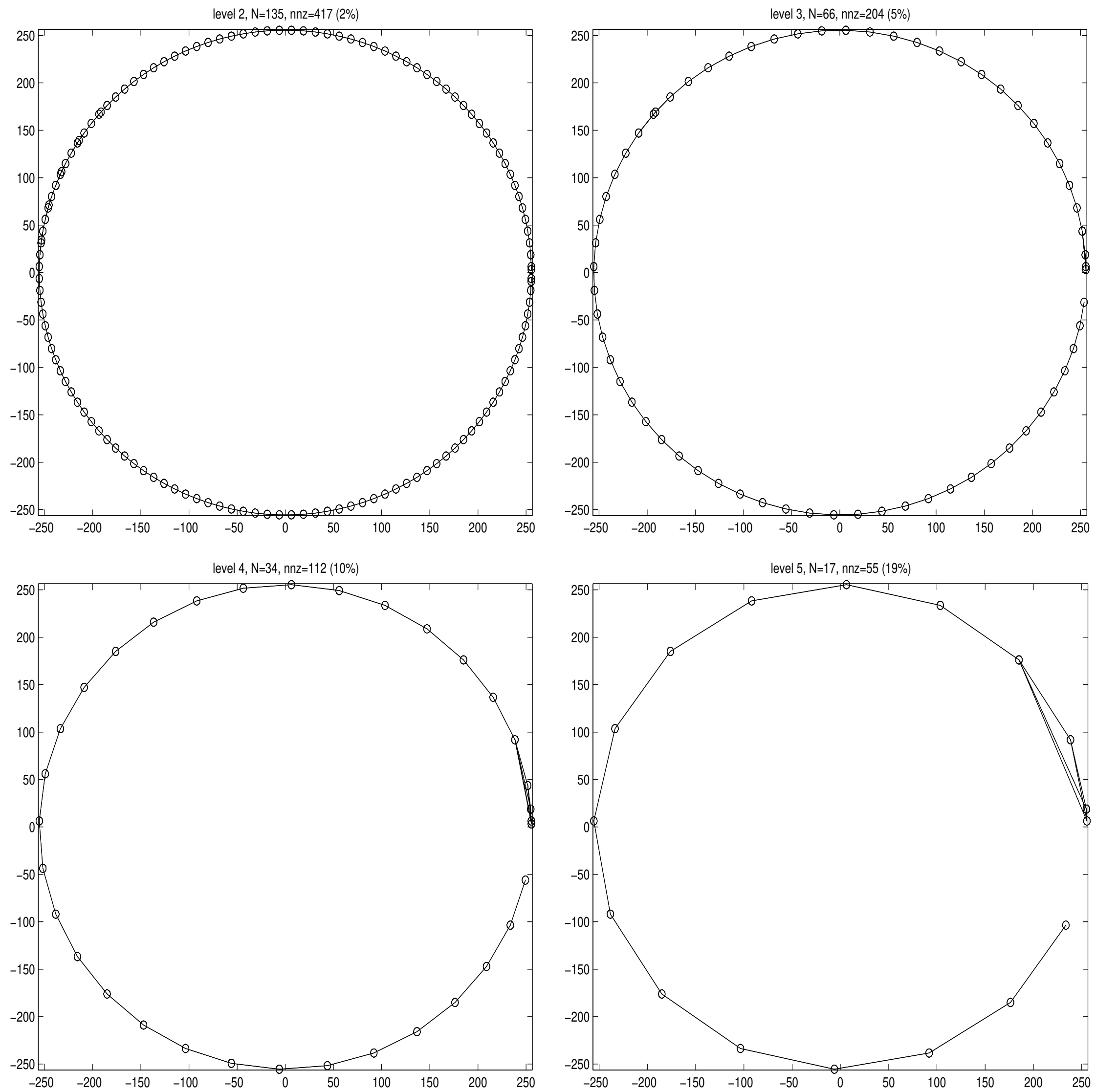
process is depicted. The situation is still comparable to the case $\alpha = 1$.

initial size	1	2	3	4	5	6	7	8	9	10
Diagonal Preconditioning, $\alpha = 100$										
2047	2047, <1	1190, <1	854, 1	335, 2	164, 5	134, 12	—	—	—	—
1023	1023, <1	525, 1	262, 1	131, 3	66, 5	42, 11	33, 23	—	—	—
511	511, 1	269, 1	132, 3	64, 5	36, 11	31, 24	—	—	—	—
Tridiagonal Preconditioning, $\alpha = 100$										
2047	2047, <1	1026, 1	514, 1	489, 3	—	—	—	—	—	—
1023	1023, <1	514, 1	259, 2	231, 6	—	—	—	—	—	—
511	511, 1	258, 2	129, 5	109, 12	—	—	—	—	—	—
Jacobi–squared Preconditioning, $\alpha = 100$										
2047	2047, <1	529, 1	273, 1	188, 3	85, 8	49, 20	30, 41	—	—	—
1023	1023, <1	283, 1	154, 3	73, 6	39, 13	24, 31	—	—	—	—
511	511, 1	135, 2	66, 5	34, 10	17, 19	—	—	—	—	—

(Column header spanning levels 1–10: *Levels, size and fill–in (%)*)

Finally we compare the condition number and the number of CG steps.

Figure 15: **Coarsening Snapshots $\alpha = 100$, Jacobi–squared preconditioning, level 2–5**



size	condition number				CG steps			
	A	M	$M_l^{(1)}A$	$M_l^{(2)}A$	A	M	$M_l^{(1)}A$	$M_l^{(2)}A$
Diagonal Preconditioning, $\alpha = 100$								
2047	—	—	—	—	10632	2047	381	158
1023	$1.4{\cdot}10^7$	$6.3{\cdot}10^5$	$1.4{\cdot}10^4$	$2.6{\cdot}10^3$	5037	1023	157	70
511	$3.4{\cdot}10^6$	$1.6{\cdot}10^5$	$5.8{\cdot}10^3$	$1.1{\cdot}10^3$	2293	511	109	50
Tridiagonal Preconditioning, $\alpha = 100$								
2047	—	—	—	—	10632	1097	164	72
1023	$1.4{\cdot}10^7$	$2.2{\cdot}10^5$	$6.3{\cdot}10^3$	$1.2{\cdot}10^3$	5037	552	104	45
511	$3.4{\cdot}10^6$	$5.5{\cdot}10^4$	$1.7{\cdot}10^3$	$3.2{\cdot}10^2$	2293	281	68	30
Jacobi–squared Preconditioning, $\alpha = 100$								
2047	—	—	—	—	10632	1403	135	54
1023	$1.4{\cdot}10^7$	$3.7{\cdot}10^5$	$3.1{\cdot}10^3$	$5.0{\cdot}10^2$	5037	711	104	44
511	$3.4{\cdot}10^6$	$9.3{\cdot}10^4$	$9.4{\cdot}10^1$	$2.2{\cdot}10^1$	2293	364	37	16

The results show that using the algebraic coarsening process a reduction of the number of cg iterations is obtained. Although the improvement is not as strong as for geometric multigrid method the the number of iterations is still much less than using only the sparse approximate inverse preconditioner. For a purely algebraic coarsening process which does neither make use of any geometric information nor of the underlying analytic problem this reduction is remarkable.

Example 29 *The second example is the two dimensional version of example 28. We consider the problem*

$$\begin{aligned} -\Delta u &= f \text{ in } [0,1]^2 \\ u &= g \text{ on } \partial[0,1]^2 \end{aligned}$$

We allow different weights in parts of the domain.

1

α_{11}	α_{12}
α_{21}	α_{22}

0 1

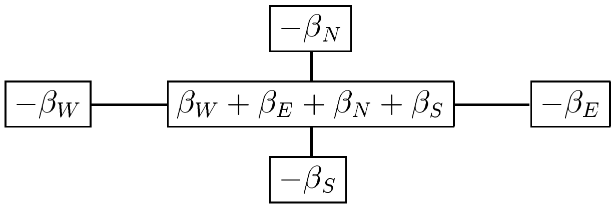
In detail we consider the weights

1	1
1	1

100	10000
1	100

100	1
1	100

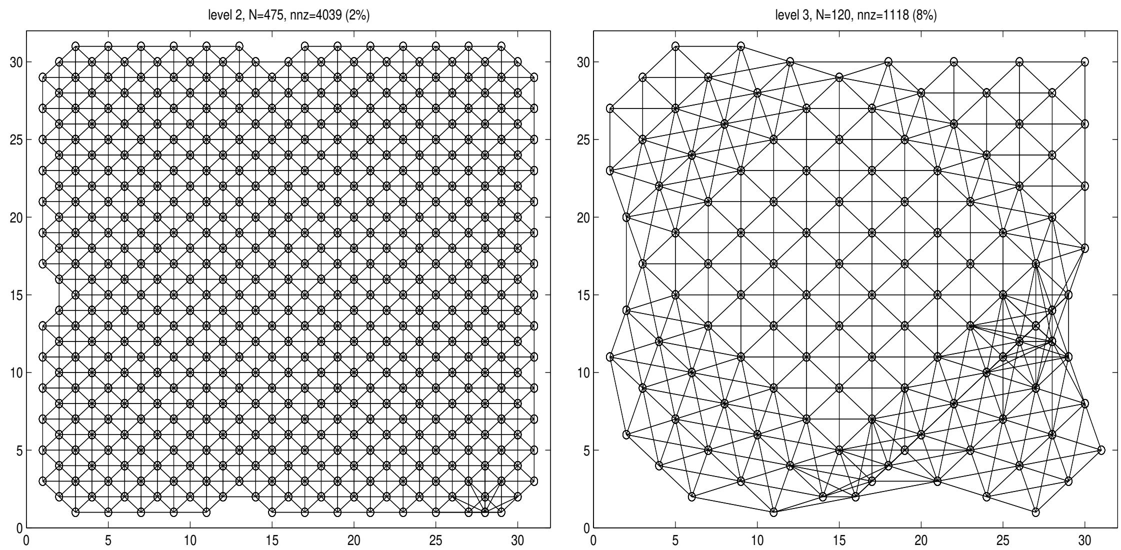
The discretization is done via a uniform grid and a five point star difference discretization. If we locally have weights $\beta_N, \beta_W, \beta_E, \beta_S$, then the discretization is described by



In any of the subdomains the value of β is identical to the weights. For nodes on the interface between the subdomain the arithmetic mean is used.

Again we use same preconditioners as in the previous example. Figures 16, 17 and 18 depict the grids obtained by the coarsening process.

Figure 16: **Coarsening Snapshots $\alpha_{ij} = 1$, diagonal preconditioning, level 2–3**



We see that for the sparse approximate inverse preconditioner, the automatic coarsening process produces a very dense grid. The main reason for this is that the condition number is overestimated and too many nodes are taken. Note that the condition number of the preconditioned matrix is not very large, so we do not expect a large improvement by including a coarse grid. However, on the third level the coarsening process stops after taking the first node. This is due to the fact the coarse grid system is already well–conditioned so no further grid is needed. We again also depict the reduction in the matrix size and the fill–in during

Figure 17: **Coarsening Snapshots $\alpha_{ij} = 1$, sparsity as initial matrix used for approx. inverse, level 2–3**

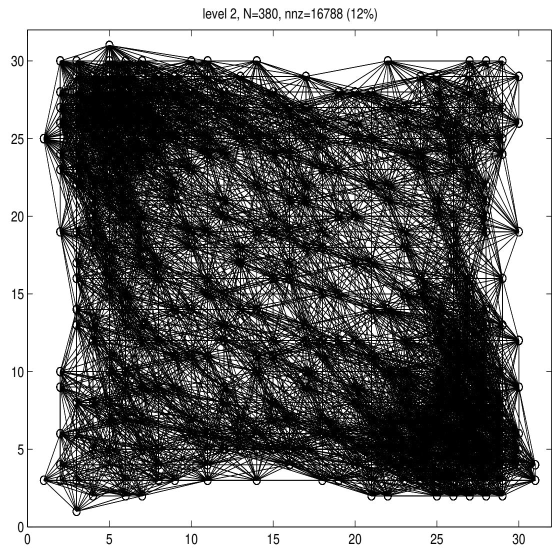
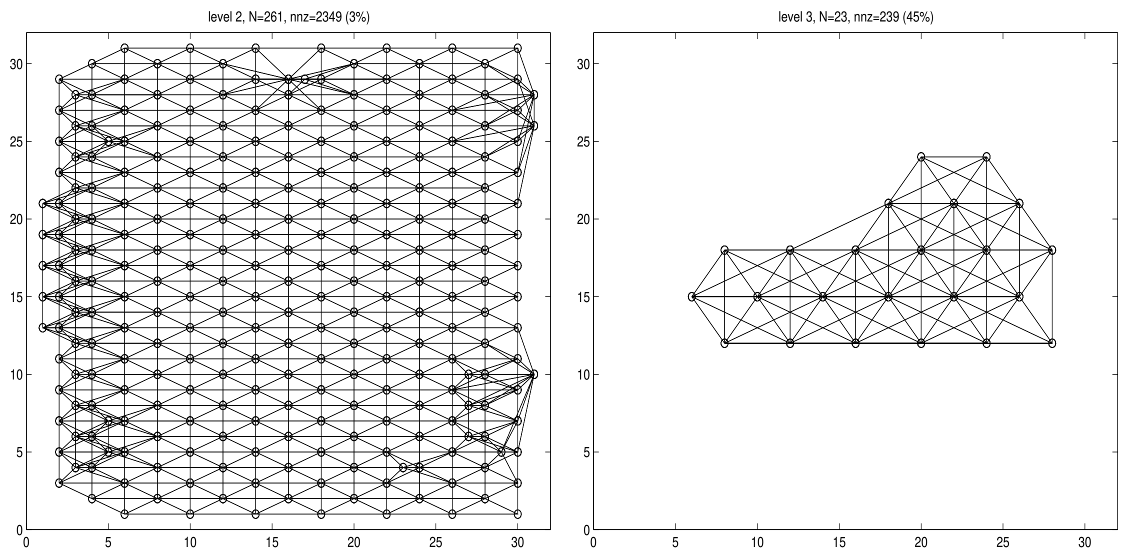


Figure 18: **Coarsening Snapshots $\alpha_{ij} = 1$, Jacobi–squared, level 2–3**



the coarsening process.

Diagonal Preconditioning, $\alpha_{ij} = 1$				
initial size	*Levels, size and fill–in (%)*			
	1	2	3	4
961	961, 1	475, 2	120, 8	1, 100
Sparse approx. inv. with same pattern as the initial matrix, $\alpha_{ij} = 1$				
961	961, 2	380, 12	1, 100	—
Jacobi–squared Preconditioning, $\alpha_{ij} = 1$				
961	961, 1	261, 4	23, 45	—

The reduction of the system size confirms the observation that not too many coarse grids are necessary.

Finally we compare the condition numbers and the number of CG steps.

Diagonal Preconditioning, $\alpha_{ij} = 1$								
size	*condition number*				*CG steps*			
	A	M	$M_l^{(1)}A$	$M_l^{(2)}A$	A	M	$M_l^{(1)}A$	$M_l^{(2)}A$
961	$6.0{\cdot}10^2$	$6.0{\cdot}10^2$	$2.2{\cdot}10^2$	$3.6{\cdot}10^1$	97	97	42	22
sparse appr. inv. with same pattern as the initial matrix, $\alpha_{ij} = 1$								
961	$6.0{\cdot}10^2$	$2.3{\cdot}10^2$	$2.3{\cdot}10^2$	$5.1{\cdot}10^1$	97	56	42	24
Jacobi–squared Preconditioning, $\alpha_{ij} = 1$								
961	$6.0{\cdot}10^2$	$5.5{\cdot}10^2$	$2.5{\cdot}10^2$	$5.8{\cdot}10^1$	97	95	56	27

Since already the initial preconditioned system shows a quite moderate condition number and a small number of iterations, the improvement using the algebraic multilevel method is not as impressive as in Example 28.

For the coefficients $(\alpha_{ij}) = \begin{pmatrix} 100 & 10^4 \\ 1 & 10 \end{pmatrix}$ *we depict the graphs of the coarse grid systems in Figures 19, 20, 21. We see that the main number of coarse grid nodes are taken in the subdomain with the largest coefficient, while in the other subdomains only a few nodes are needed.*

Again we examine the reduction of the system size by the coarsening process. As motivated by the graphical illustration less nodes are used for the coarse grid system.

Diagonal Preconditioning, $(\alpha_{ij}) = \begin{pmatrix} 100 & 10^4 \\ 1 & 10 \end{pmatrix}$				
initial size	*Levels, size and fill–in (%)*			
	1	2	3	4
961	961, 1	474, 2	76, 32	1, 38
Sparse approx. inv. with same pattern as the initial matrix, $(\alpha_{ij}) = \begin{pmatrix} 100 & 10^4 \\ 1 & 10 \end{pmatrix}$				
961	961, 2	70, 53	1, 100	
Jacobi–squared Preconditioning, $(\alpha_{ij}) = \begin{pmatrix} 100 & 10^4 \\ 1 & 10 \end{pmatrix}$				
961	961, 1	43, 20	1, 100	—

Figure 19: **Coarsening Snapshots** $(\alpha_{11}, \alpha_{12}, \alpha_{21}, \alpha_{22}) = (100, 10^4, 1, 100)$, **diagonal preconditioning, level 2–3**

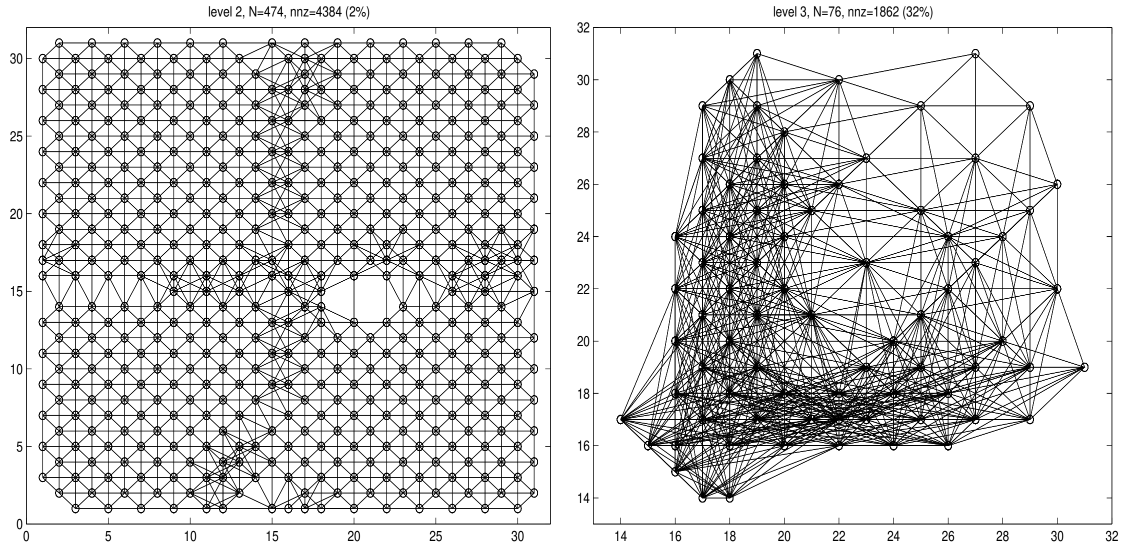


Figure 20: **Coarsening Snapshots** $(\alpha_{11}, \alpha_{12}, \alpha_{21}, \alpha_{22}) = (100, 10^4, 1, 100)$**, sparsity as initial matrix used for approx. inverse, level 2–3**

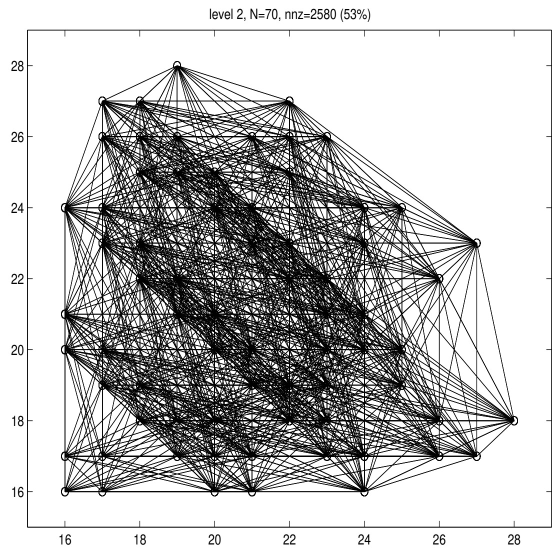
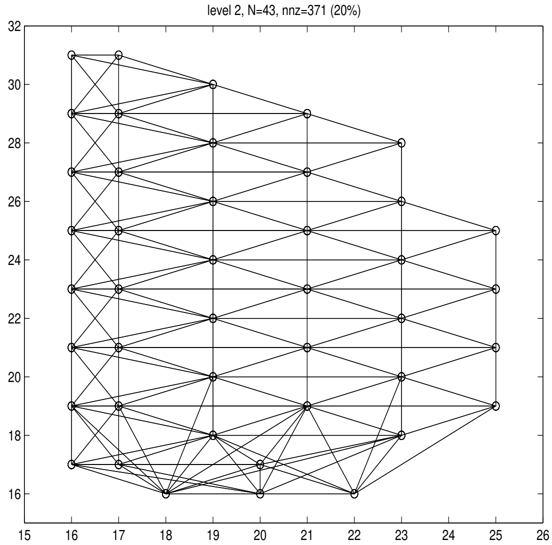


Figure 21: **Coarsening Snapshots** $(\alpha_{11}, \alpha_{12}, \alpha_{21}, \alpha_{22}) = (100, 10^4, 1, 100)$**, Jacobi–squared, level 2**



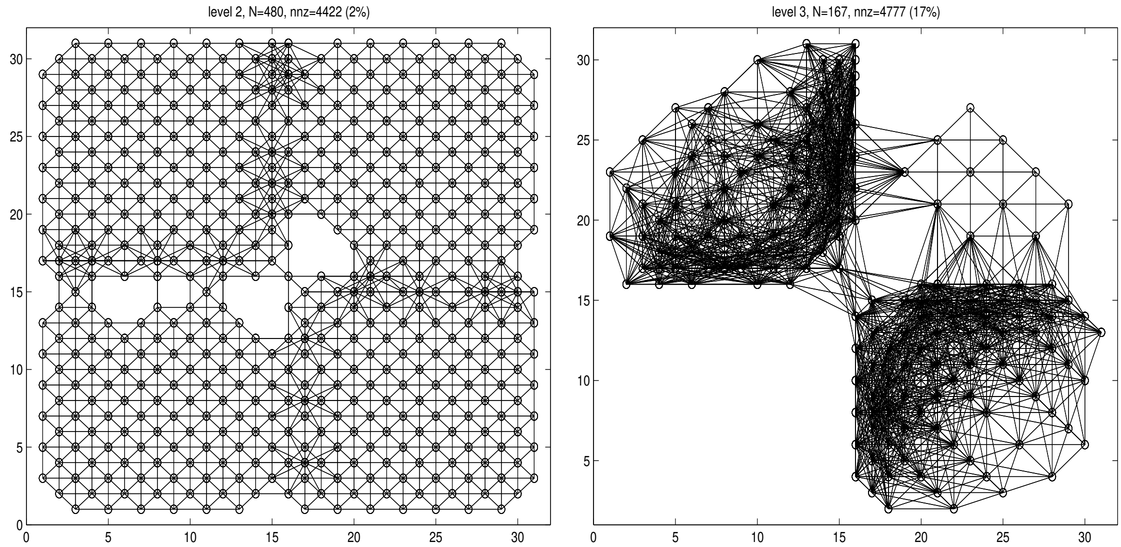
While for diagonal preconditioning case still about 50% of the nodes are used on the second level, the situation is better for the other two preconditioners. Here mainly the coarsening process is restricted to the domain with the largest coefficient.

Diagonal Preconditioning, $(\alpha_{ij}) = \begin{pmatrix} 100 & 10^4 \\ 1 & 10 \end{pmatrix}$								
size	*condition number*				*CG steps*			
	A	M	$M_l^{(1)}A$	$M_l^{(2)}A$	A	M	$M_l^{(1)}A$	$M_l^{(2)}A$
961	$1.5{\cdot}10^6$	$8.0{\cdot}10^2$	$3.0{\cdot}10^2$	$7.0{\cdot}10^1$	3168	103	67	34
sparse appr. inv. with same pattern as the initial matrix, $(\alpha_{ij}) = \begin{pmatrix} 100 & 10^4 \\ 1 & 10 \end{pmatrix}$								
961	$1.5{\cdot}10^6$	$3.2{\cdot}10^2$	$3.3{\cdot}10^2$	$7.5{\cdot}10^1$	3168	58	68	37
Jacobi–squared Preconditioning, $(\alpha_{ij}) = \begin{pmatrix} 100 & 10^4 \\ 1 & 10 \end{pmatrix}$								
961	$1.5{\cdot}10^6$	$8.4{\cdot}10^2$	$7.4{\cdot}10^2$	$2.0{\cdot}10^2$	3168	101	110	64

In contrast to Example 28 using an algebraic multilevel method does not greatly improve the number of iterations, even more so for the sparse approximate inverse with the same pattern as the initial matrix as well as for Jacobi–squared preconditioning, where the number of iteration slightly increases. Since one step of the multigrid scheme is more expensive, in this case the multilevel method slows down the process. As mentioned above this is essentially due to the fact that the system is not very ill–conditioned and from this point of view we cannot expect a great improvement by the multilevel scheme.

Finally we choose $(\alpha_{ij}) = \begin{pmatrix} 100 & 1 \\ 1 & 100 \end{pmatrix}$. *The coarsening process is illustrated in Figure 22, 23, 24.*

Figure 22: **Coarsening Snapshots** $(\alpha_{11}, \alpha_{12}, \alpha_{21}, \alpha_{22}) = (100, 1, 100, 1)$, **diagonal preconditioning, level 2-3**



The main problem using the automatic coarsening process is that the matrix is approximately block diagonal which causes the coarsening process to take too many nodes in the subdomains with the large coefficients.

initial size	1	2	3	4
Diagonal Preconditioning, $(\alpha_{ij}) = \begin{pmatrix} 1 & 100 \\ 100 & 1 \end{pmatrix}$				
	Levels, size and fill–in (%)			
961	961, 1	480, 2	167, 17	1, 100
Sparse approx. inv. with same pattern as the initial matrix, $(\alpha_{ij}) = \begin{pmatrix} 1 & 100 \\ 100 & 1 \end{pmatrix}$				
961	961, 2	238, 23	14, 100	—
Jacobi–squared Preconditioning, $(\alpha_{ij}) = \begin{pmatrix} 1 & 100 \\ 100 & 1 \end{pmatrix}$				
961	961, 1	388, 5	1, 100	—

¿From the reduction of the system size we can see that essentially not more then 2 grids are required. The third level only consists of a very few number of nodes. Only in the case

Figure 23: **Coarsening Snapshots** $(\alpha_{11}, \alpha_{12}, \alpha_{21}, \alpha_{22}) = (100, 1, 100, 1)$**, sparsity as initial matrix used for approx. inverse, level 2**

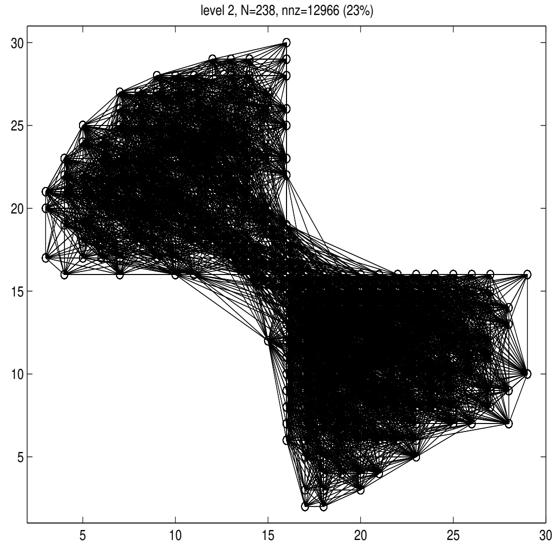
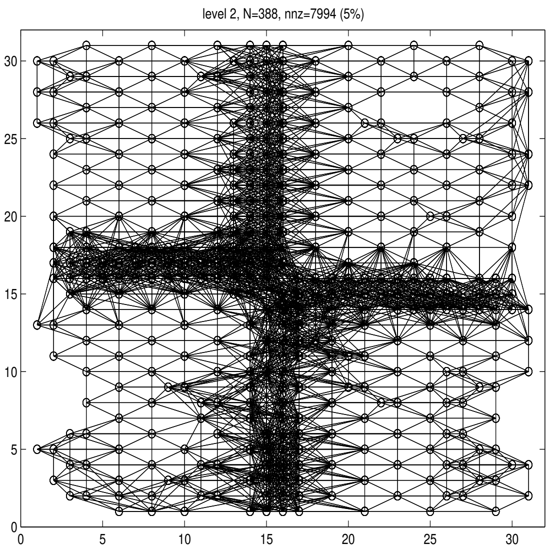


Figure 24: **Coarsening Snapshots** $(\alpha_{11}, \alpha_{12}, \alpha_{21}, \alpha_{22}) = (100, 1, 100, 1)$**, Jacobi–squared, level 2**



of diagonal preconditioning more nodes are used.

size	condition number				CG steps			
	A	M	$M_l^{(1)}A$	$M_l^{(2)}A$	A	M	$M_l^{(1)}A$	$M_l^{(2)}A$
Diagonal Preconditioning, $(\alpha_{ij}) = \begin{pmatrix} 1 & 100 \\ 100 & 1 \end{pmatrix}$								
961	$1.5{\cdot}10^4$	$7.4{\cdot}10^2$	$3.3{\cdot}10^2$	$6.6{\cdot}10^1$	516	113	58	28
sparse appr. inv. with same pattern as the initial matrix, $(\alpha_{ij}) = \begin{pmatrix} 1 & 100 \\ 100 & 1 \end{pmatrix}$								
961	$1.5{\cdot}10^4$	$2.8{\cdot}10^2$	$2.7{\cdot}10^2$	$6.1{\cdot}10^1$	516	59	60	30
Jacobi–squared Preconditioning, $(\alpha_{ij}) = \begin{pmatrix} 1 & 100 \\ 100 & 1 \end{pmatrix}$								
961	$1.5{\cdot}10^4$	$7.2{\cdot}10^2$	$5.2{\cdot}10^2$	$1.8{\cdot}10^2$	516	110	57	33

The condition number as well as the number of iterations show a slight improvement using the multilevel scheme.

Example 30 *The matrix in this example is that of Example 26 and essentially corresponds to the matrix LANPRO/NOS2 from the Harwell–Boeing collection [9] with* $n = 190$. *Although this matrix is not very big, its condition number is already huge. Applying the coarsening process to this matrix using diagonal preconditioning we obtain the graphs shown in Figure 25.*

The analogous situation when using the sparse approximate inverse with same sparsity as the initial matrix is given in Figure 26.

At last the coarsening snapshots for the Jacobi–squared preconditioner are given in Figure 27.

We have the following reduction in the size of the system and the related fill–in.

initial size	Levels, size and fill–in (%)			
	1	2	3	4
Diagonal Preconditioning,				
190	190, 3	94, 6	56, 15	30, 40
Preconditioning with pattern as the initial matrix				
190	190, 3	90, 14	40, 42	19, 81
Jacobi–squared preconditioning				
190	190, 5	66, 20	26, 40	—

Here the comparison of the condition numbers and the number of cg steps is quite interest-

Figure 25: **Coarsening Snapshots diagonal preconditioning, level 1–4**

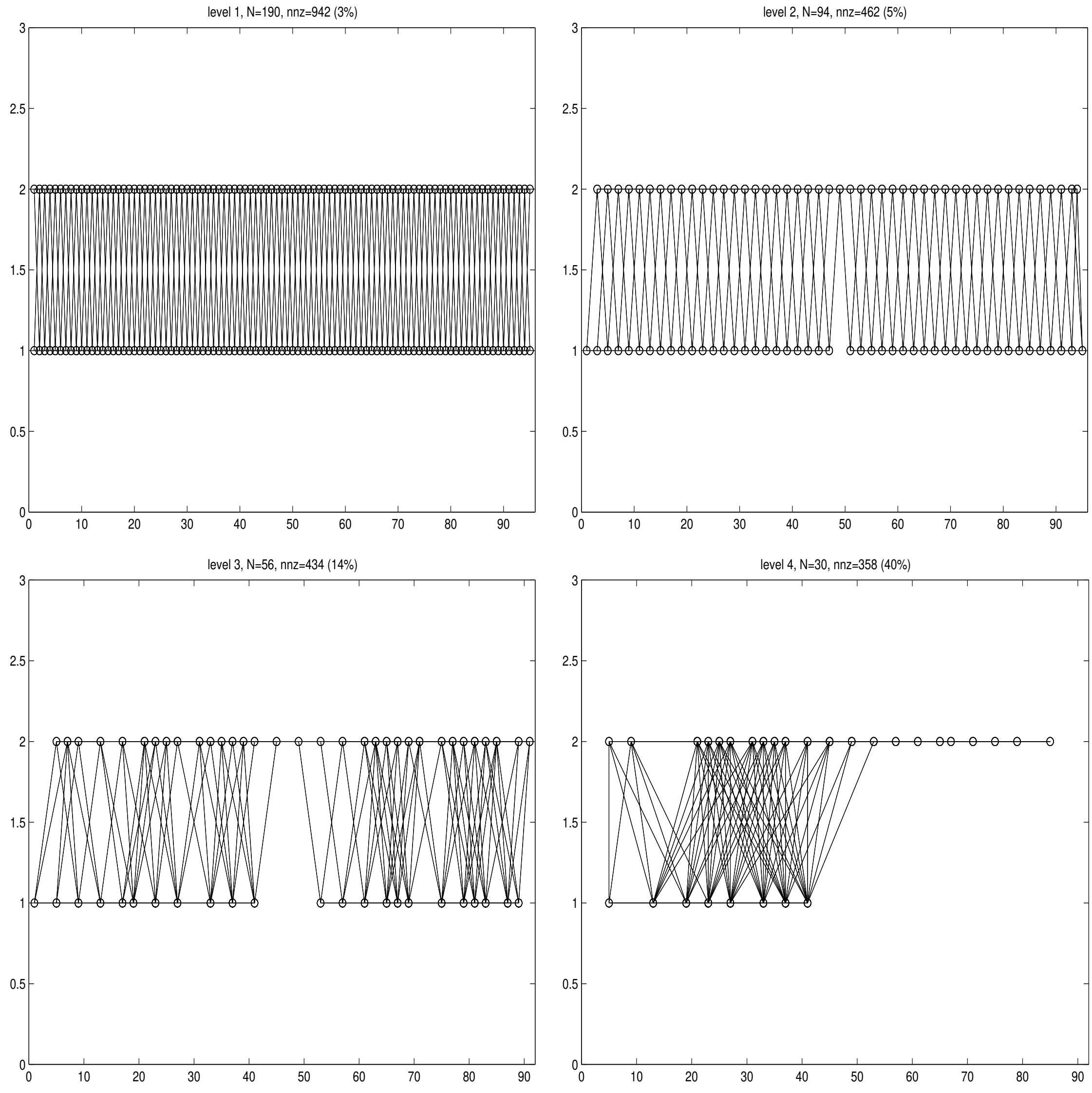


Figure 26: **Coarsening Snapshots, preconditioning with same sparsity pattern as the initial matrix, level 1–4**

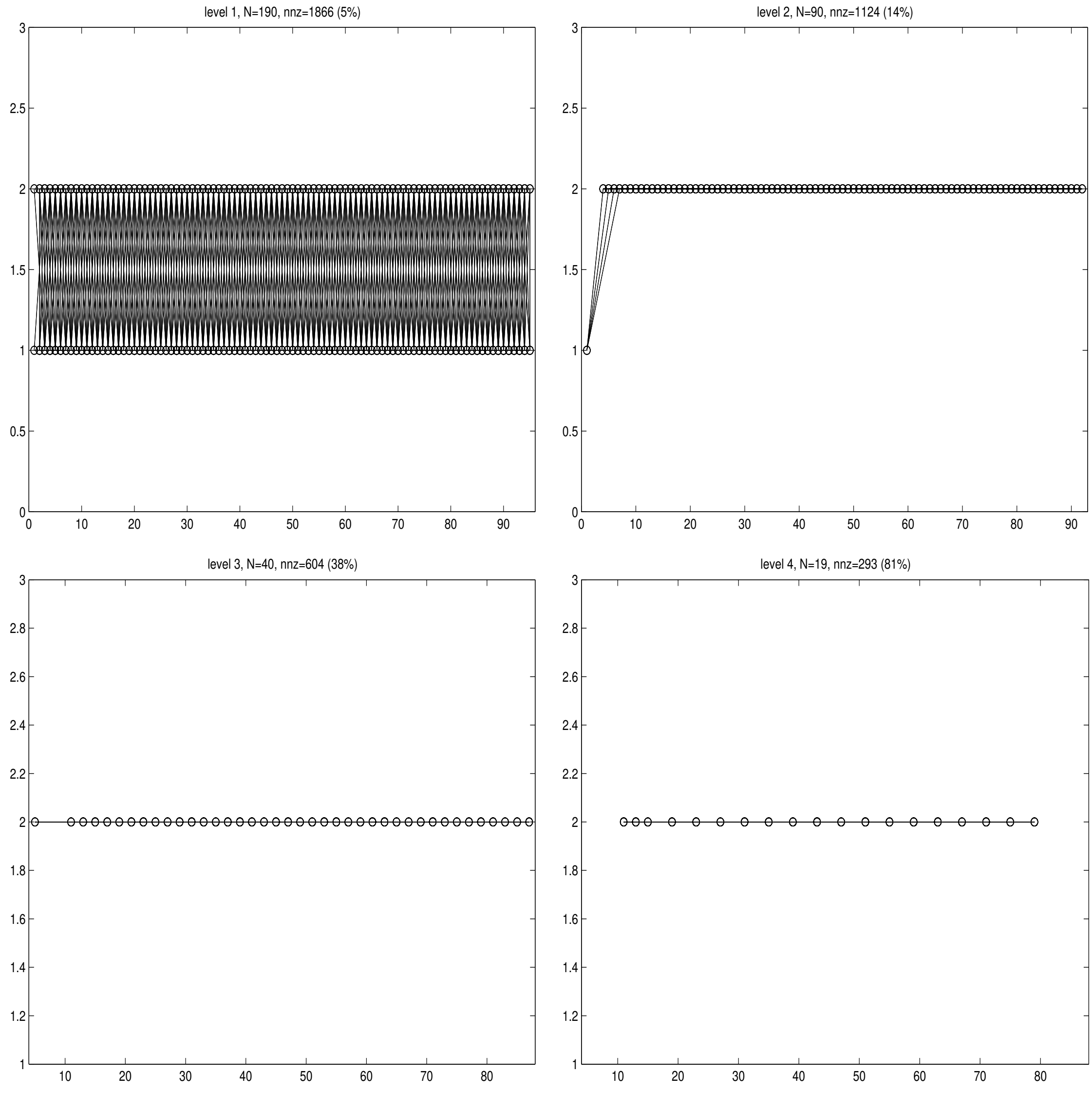
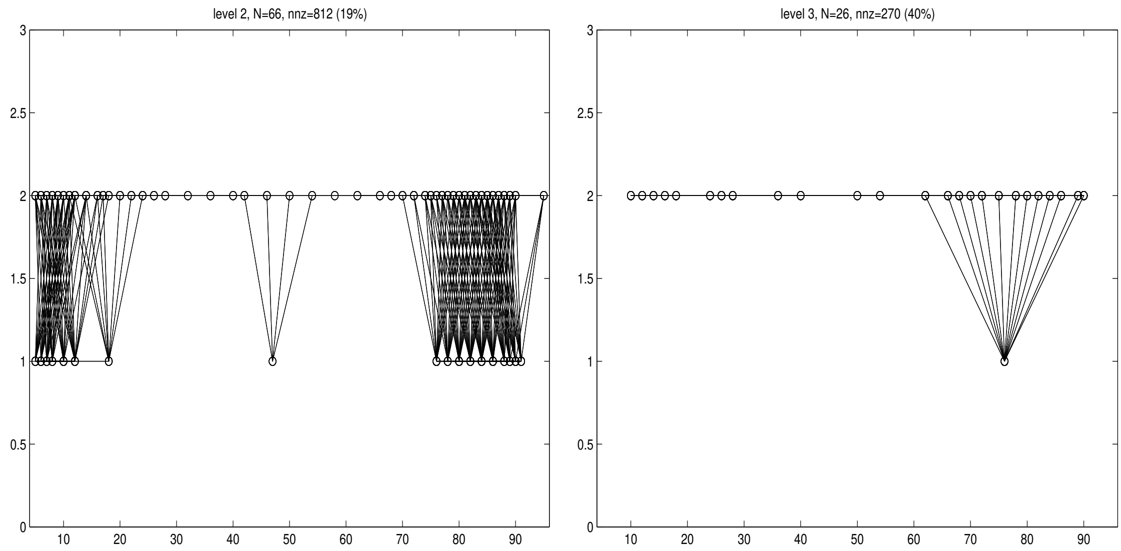


Figure 27: **Coarsening Snapshots, preconditioning with same sparsity pattern as the initial matrix, level 2–3**



ing.

Diagonal Preconditioning								
size	*condition number*				*CG steps*			
	A	M	$M_l^{(1)}A$	$M_l^{(2)}A$	A	M	$M_l^{(1)}A$	$M_l^{(2)}A$
190	$2.6{\cdot}10^7$	$1.6{\cdot}10^7$	$8.8{\cdot}10^3$	$1.8{\cdot}10^3$	1062	540	82	38
Preconditioning with same sparsity as the init. matrix								
190	$2.6{\cdot}10^7$	$4.3{\cdot}10^6$	$1.1{\cdot}10^4$	$2.2{\cdot}10^3$	1062	325	72	33
Jacobi–squared preconditioning								
190	$2.6{\cdot}10^7$	$1.3{\cdot}10^7$	$7.2{\cdot}10^6$	$1.6{\cdot}10^6$	1062	571	352	188

Although the condition number of the preconditioned system is still large, the number of iteration is small. Here almost all eigenvalues are clustered near 1. Only a few exceptional eigenvalues are small, which lead to a large condition number but fortunately only have a minor impact on the iterative process.

In this case no great improvement was obtained by the Jacobi–squared preconditioner. It has turned out in this example that there was no clustering of the eigenvalues at the upper bound of the spectrum of the preconditioned matrix. Obviously the coarsening cannot end up in an efficient multilevel scheme.

6 Conclusions

We have derived new approaches for the construction of algebraic multilevel methods that automatically detects the coarse grid by choosing suitably chosen columns of the residual matrix. The basic feature is a combination of ideas from sparse approximate inverses and spares QR-factorizations. We have presented the mathematical theory to develop optimal preconditioners. The key feature of the new approach is the choice of an efficient pivoting strategy for the needed sparse QR-factorization and it turns out that an heuristic approach based on approximation of eigenvectors associated with the smallest eigenvalues of the preconditioned system combined with an eigenvector independent criterion yields very convincing numerical results. A more detailed analysis of methods to construct good pivoting strategies needs further research.

References

[1] MATLAB – The language of technical computing. The MathWorks Inc., 1996.

[2] O. Axelsson, M. Neytcheva, and B. Polman. An application of the boardering method to solve nearly singular systems. *Vestnik Moskovskogo Universiteta, Seria 15, Vychisl. Math. Cybern.*, 1:3–25, 1996.

[3] O. Axelsson, A. Padiy, and B. Polman. Generalized augmented matrix preconditioning approach and its application to iterative solution of ill-conditioned algebraic systems. Technical report, Katholieke Universiteit Nijmegen, Fakulteit der Wiskunde en Informatica, 1999.

[4] R. Barret, M. Berry, T. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkout, R. Pozo, C. Romine, and H. van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods.* SIAM Publications, 1995.

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

[6] W. Bunse and A. Bunse-Gerstner. *Numerische lineare Algebra.* B.G. Teubner Stuttgart, 1985.

[7] E. Chow and Y. Saad. Approximate inverse preconditioners for general sparse matrices. Research Report UMSI 94/101, University of Minnesota, Super Computing Institute, Minneapolis, Minnesota, 1994.

[8] I. Duff, A. Erisman, and J. Reid. *Direct Methods for Sparse Matrices.* Oxford University Press, 1986.

[9] I. Duff, R. Grimes, and J. Lewis. Sparse matrix test problems. *ACM Trans. Math. Software*, 15:1–14, 1989.

[10] R. Freund, G. Golub, and N. Nachtigal. Iterative solution of linear systems. *Acta Numerica*, pages 1–44, 1992.

[11] J. A. George and J. W. Liu. *Computer Solution of Large Sparse Positive Definite Systems.* Prentice-Hall, Englewood Cliffs, NJ, USA, 1981.

[12] N. Gibbs, W. Poole, and P. Stockmeyer. An algorithm for reducing the bandwidth and profile of a sparse matrix. *SIAM J. Numer. Anal.*, 13:236–250, 1976.

[13] G. Golub and C. V. Loan. *Matrix Computations*. The Johns Hopkins University Press, third edition, 1996.

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

[15] M. Grote and T. Huckle. Parallel preconditioning with sparse approximate inverses. *SIAM J. Sci. Comput.*, 18(3), 1997.

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

[17] W. Hackbusch. *Iterative Lösung großer schwachbesetzter Gleichungssysteme*. B.G. Teubner Stuttgart, second edition, 1993.

[18] T. Huckle. Matrix multilevel methods and preconditioning. Technical report SFB–Bericht Nr. 342/11/98 A, Technische Universität München, Fakulktät für Informatik, 1998.

[19] I. E. Kaporin. New convergence results and preconditioning strategies for conjugate gradient method. *Numer. Lin. Alg. w. Appl.*, 1(2):179–210, 1994.

[20] Y. Kolotilina and Y. Yeremin. Factorized sparse approximate inverse preconditionings I. theory. *SIAM J. Matrix Anal. Appl.*, 14:45–58, 1993.

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

[22] D. P. O'Leary. The block conjugate gradient algorithm and related methods. *Linear Algebra Appl.*, 29:293–322, 1980.

[23] B. Parlett. *The Symmetric Eigenvalue Problem*. Prentice-Hall, 1980.

[24] J. Ruge and K. Stüben. Algebraic multigrid. In S. McCormick, editor, *Multigrid Methods*. SIAM Publications, 1987.

[25] Y. Saad. *Iterative Methods for Sparse Linear Systems*. PWS Publishing Company, 1996.

[26] Y. Saad and M. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.*, 7:856–869, 1986.

[27] G. Stewart. Conjugate direction methods for solving systems of linear equations. *Numer. Math.*, 21:285–297, 1973.

[28] G. Stewart. Four algorithms for the the efficient computation of truncated pivoted QR approximations to a sparse matrix. Technical report UMIACS TR-98-12 CMSC TR-3875, University of Maryland, Department of Computer Science, 1998. to appear in Numerische Mathematik.

[29] R. S. Varga. *Matrix Iterative Analysis*. Prentice-Hall, Englewood Cliffs, New Jersey, 1962.

Other titles in the SFB393 series:

99-01 P. Kunkel, V. Mehrmann, W. Rath. Analysis and numerical solution of control problems in descriptor form. January 1999.

99-02 A. Meyer. Hierarchical preconditioners for higher order elements and applications in computational mechanics. January 1999.

99-03 T. Apel. Anisotropic finite elements: local estimates and applications (Habilitationsschrift). January 1999.

99-04 C. Villagonzalo, R. A. Römer, M. Schreiber. Thermoelectric transport properties in disordered systems near the Anderson transition. February 1999.

99-05 D. Michael. Notizen zu einer geometrisch motivierten Plastizitätstheorie. Februar 1999.

99-06 T. Apel, U. Reichel. SPC-PM Po 3D V 3.3, User's Manual. February 1999.

99-07 F. Tröltzsch, A. Unger. Fast solution of optimal control problems in the selective cooling of steel. March 1999.

99-08 W. Rehm, T. Ungerer (Eds.). Ausgewählte Beiträge zum 2. Workshop Cluster-Computing 25./26. März 1999, Universität Karlsruhe. März 1999.

99-09 M. Arav, D. Hershkowitz, V. Mehrmann, H. Schneider. The recursive inverse eigenvalue problem. March 1999.

99-10 T. Apel, S. Nicaise, J. Schöberl. Crouzeix-Raviart type finite elements on anisotropic meshes. May 1999.

99-11 M. Jung. Einige Klassen iterativer Auflösungsverfahren (Habilitationsschrift). Mai 1999.

99-12 V. Mehrmann, H. Xu. Numerical methods in control, from pole assignment via linear quadratic to H_∞ control. June 1999.

99-13 K. Bernert, A. Eppler. Two-stage testing of advanced dynamic subgrid-scale models for Large-Eddy Simulation on parallel computers. June 1999.

99-14 R. A. Römer, M. E. Raikh. The Aharonov-Bohm effect for an exciton. June 1999.

99-15 P. Benner, R. Byers, V. Mehrmann, H. Xu. Numerical computation of deflating subspaces of embedded Hamiltonian pencils. June 1999.

99-16 S. V. Nepomnyaschikh. Domain decomposition for isotropic and anisotropic elliptic problems. July 1999.

99-15 T. Stykel. On a criterion for asymptotic stability of differential-algebraic equations. August 1999.

The complete list of current and former preprints is available via
`http://www.tu-chemnitz.de/sfb393/preprints.html`.