SIAM J. SCI. COMPUT.
Vol. 15, No. 3, pp. 000–000, May 1994



# MULTILEVEL ALGORITHMS CONSIDERED AS ITERATIVE METHODS ON SEMIDEFINITE SYSTEMS*

MICHAEL GRIEBEL†

**Abstract.** For the representation of piecewise $d$-linear functions, instead of the usual finite element basis, we introduce a generating system that contains the nodal basis functions of the finest level and of all coarser levels of discretization. This approach enables us to work directly with multilevel decompositions of a function.

For a partial differential equation, the Galerkin scheme based on this generating system results in a semidefinite matrix equation that has in the 1D case only about twice, in the 2D case about 4/3 times, and in the 3D case about 8/7 times, as many unknowns as the usual system. Furthermore, the semidefinite system possesses not just one, but many solutions. However, the unique solution of the usual definite finite element problem can be easily computed from every solution of the semidefinite problem.

We show that modern multilevel algorithms can be considered as standard iterative methods over the semidefinite system. The conjugate gradient method (CG) for the semidefinite system is equivalent to the BPX- or MDS-preconditioned conjugate gradient method for the linear system that arises from the usual finite element basis. The Gauss-Seidel iteration applied to the semidefinite system is equivalent to the multigrid method (MG) applied to the standard basis system. Consequently, the Gauss-Seidel-preconditioned conjugate gradient method for the semidefinite system is equivalent to MG-CG for the standard basis system.

This paper includes results of numerical experiments regarding the condition number and the convergence rates of different iterative methods for the semidefinite system.

**Key words.** partial differential equations, multilevel methods, BPX-preconditioner, conjugate gradients, multigrid methods, Gauss-Seidel iteration, semidefinite system.

## 1. Introduction.

In recent years, different multilevel methods have been used extensively to obtain approximations to solutions of partial differential equations. Besides the multigrid method (see the references in [13], [10]) and the hierarchical basis multigrid method [3], multilevel methods for the preconditioning of the conjugate gradient algorithm have also been developed, including the BPX-preconditioner of Bramble, Pasciak and Xu (see [21], [5]), its generalization MDS (see [25], [20]), and the hierarchical basis preconditioner of Yserentant (see [22]). For a comparison of these preconditioners, see [23]. Another interesting type of multilevel preconditioner has been developed in [2]. Furthermore, preconditioners that make use of the multigrid method have been studied in [11].

Under rigorous assumptions on regularity and approximation properties, a convergence rate that is independent of the number of levels can be proved for multigrid methods. In contrast to this, two-level schemes have been proved to converge under weaker assumptions. The same is true for the preconditioner of Xu and, in the 2D case, of Yserentant. Here, no regularity assumptions are needed to prove a condition num-

* Received by the editors April 30, 1992; accepted for publication (in revised form) December 8, 1992. This research was partially supported by Deutsch Forschunsgemeinschaft-Sonderforschungsbereich SFB 0342 and Deutsch Forschunsgemeinschaft-AZ 477/766/92

† Institut für Informatik, Technische Universität München Arcisstraße 21, D-8000 München 2, Germany (griebel@informatik.tu-muenchen.de)

ber of $O(k^2)$ for the preconditioned system, where $k$ denotes the number of employed levels. Under strong regularity assumptions, a condition number estimate of $O(k)$ can be given. See the appendix in [23].

Recently, Oswald [16], [17], [18], Xu [20], and Bornemann and Yserentant [4] showed that (even under weak assumptions) the condition number of the BPX-preconditioner is $O(1)$. Also, Zhang [25] gave an elegant proof for the condition number of the MDS operator, which is a slight generalization of BPX. It grows at most linearly with the number of levels in the general case, and is bounded by a constant independent of the mesh sizes and number of levels under the $H^2$-regularity assumption. In this sense, multilevel methods are optimal.

We present in this paper a new approach to multilevel methods that abandons the usual basis approach in favor of generating functions that span the approximation space but are generally linearly dependent. The idea is to use the standard basis functions on **all** levels of discretization. The resulting Galerkin scheme therefore results in a redundant semidefinite system of linear equations that has multiple solutions, but each is equivalent to the unique solution of the standard basis approach. Fortunately, in many cases, the generalized condition number (i.e. the condition number of the matrix restricted to the orthogonal complement of its null space) of the semidefinite system matrix is of the order $O(1)$, and we will see that its singularity does not disrupt the iterative process we will use.

More specifically, there is no need for a basis of the approximation space if energy minimizing iteration methods like the conjugate gradient method or Gauss-Seidel type relaxations are used. In this case the use of a basis is even an obstacle because smooth errors cannot be adequately attenuated by the standard basis functions. Acceleration of these methods can be gained only by quite complex techniques like BPX-preconditioning or, in the multigrid context, by coarse grid corrections.

In contrast, the use of the generating system allows us to work directly with multilevel decompositions of functions. This gives additional freedom that is beyond the scope of the traditional basis approach and leads in a natural manner to accelerated algorithms. It is of special interest that standard iteration methods for the semidefinite system are equivalent to modern multilevel methods applied to the linear system that arises from the usual finite element basis.

We show that the BPX-preconditioner and its generalization MDS together with the conjugate gradient method are just special implementations of the (diagonally scaled) CG-iteration on the semidefinite system. Similarly, the HB-preconditioner can be interpreted in terms of the semidefinite system. Furthermore, we will show that the Gauss-Seidel iteration for the semidefinite system is equivalent to the multigrid method for the standard basis system. For Aitken's double sweep algorithm, which is also known as symmetric Gauss-Seidel (SGS) we obtain the standard (1,1)-MG-V-cycle. In this sense, our method and McCormick's approach of Unigrid and PML (multilevel projection method) are in similar spirit, see [14], [15]. Additionally, the HB-MG-method can be explained easily in terms of the semidefinite system. If we use the Gauss-Seidel iteration as a CG-preconditioner for the semidefinite system, we obtain an algorithm that is equivalent to MG-CG for the standard basis system.

The outline of the remainder of this paper is as follows. In Section 2, we introduce the generating system, derive the semidefinite system of linear equations, and discuss its properties. In Section 3, we consider iterative methods on the semidefinite system. First, we study the diagonally preconditioned conjugate gradient method. We show that it is equivalent to the BPX- or MDS-preconditioned CG-method for the standard basis system. Then we demonstrate that the Gauss-Seidel iteration for the semidefinite system is equivalent to the multigrid method for the standard basis system. Furthermore, we show that the use of Gauss-Seidel relaxations as a preconditioner for the semidefinite system results in MG-CG for the standard basis system. Finally, the results of numerical experiments illustrating the theory of the earlier sections are given in Section 4. An extended version of this paper can be found in [7].

**2. The semidefinite system.** In this section, we introduce a generating system that replaces the usual finite element basis in the discretization of a boundary value problem of a partial differential equation. We then derive the associated semidefinite linear system and discuss its properties.

Consider a partial differential equation in $d$ dimensions with a linear, second-order operator on the domain $\Omega = (0,1)^d$, $d$=1,2,...,

$$Lu = f \; on \; \Omega, \tag{1}$$

with appropriate boundary conditions and solution $u$. For reasons of simplicity, we restrict ourselves to homogeneous Dirichlet boundary conditions. Given an appropriate function space $V$, the problem can be expressed equivalently in its variational form: Find a function $u \in V$ with

$$a(u,v) = (f,v) \quad \forall \; v \in V. \tag{2}$$

(In the case of homogeneous Dirichlet boundary conditions, $V$ would be the Sobolev space $H_0^1(\Omega)$.) Here, $a$: $V \times V \to V$ is a bounded, positive-definite, symmetric bilinear form and $(.,.)$ is the usual linear form for the right-hand side. Let $||.||_a := \sqrt{a(.,.)}$ denote the induced energy norm. We assume that $V$ is complete with respect to $||.||_a$, which is true if $a(.,.)$ is $H_0^1$-elliptic. The Lax-Milgram lemma then guarantees the existence and uniqueness of the solution of (2). If we consider directly the energy functional $1/2 \cdot a(u,u) - (f,u)$, the problem can be stated alternatively as minimization of the energy in $V$.

**2.1. Spaces, bases, and the generating system.** Assume we are given a sequence of uniform, equidistant, nested grids,

$$\Omega_1 \subset \Omega_2 \subset ... \subset \Omega_{k-1} \subset \Omega_k, \tag{3}$$

on $\Omega$ with respective mesh sizes $h_l$=$2^{-l}$, $l$=1,..,$k$, and an associated sequence of spaces of piecewise $d$-linear functions,

$$V_1 \subset V_2 \subset .... \subset V_{k-1} \subset V_k, \tag{4}$$

with dimensions

$$n_l := dim(V_l) = (2^l - 1)^d, l = 1,..,k. \tag{5}$$

Here, 1 denotes the coarsest and $k$ the finest level of discretization. Consider also the sequence

$$N_1 \subset N_2 \subset ... \subset N_{k-1} \subset N_k \tag{6}$$

of sets of grid points $N_l = \{x_1, ..., x_{n_l}\}$ in the interior $\Omega_l$, $l=1,..,k$.

The standard finite element basis which spans $V_l$ on the equidistant grid $\Omega_l$ is denoted by $B_l$. It contains the nodal basis functions $\phi_i^{(l)}$, $i=1,..,n_l$, which are defined by

$$\phi_i^{(l)}(x_j) = \delta_{i,j}, \quad x_j \in N_l. \tag{7}$$

Note that we use rectangular grids. Therefore, the 2D basis functions can be written as the product of two 1D basis functions for the different coordinate directions. The analogue is true for the 3D case, where the basis functions are written as the product of three 1D basis functions.

Any function $u \in V_k$ can be expressed uniquely by

$$u = \sum_{i=1}^{n_k} u_i^{(k)} \phi_i^{(k)}, \tag{8}$$

with the vector $u_k := (u_1^{(k)}, u_2^{(k)}, ..., u_{n_k}^{(k)})^T$ of nodal values.

In contrast to the basis approach expressed by the expansion in (8), we instead consider the set of functions $E_k$ defined as the union of all the different nodal bases $B_l$ for the levels $l=1,..,k$,

$$E_k = B_1 \cup B_2 \cup .... \cup B_{k-1} \cup B_k. \tag{9}$$

Clearly, because $E_k$ is a linearly dependent set of functions, it is no longer a basis for $V_k$, but merely a generating system. See figure 1 for a simple 1D example.

FIG. 1. *Functions of the generating system $E_3$ in the 1D case.*

In any case, an arbitrary function $u \in V_k$ can be expressed in terms of the generating system by

$$u = \sum_{l=1}^{k} \sum_{i=1}^{n_l} w_i^{(l)} \phi_i^{(l)} \tag{10}$$

with the block vector $w_k^E := (w^{(1)}, w^{(2)}, ..., w^{(k)})^T$, were $w^{(l)} = (w_1^{(l)}, w_2^{(l)}, ..., w_{n_l}^{(l)})$, $l = 1,..,k$. This represents a **level-wise decomposition** of $u$ into functions $g^{(l)} \in V_l$, $l = 1,..,k$, where

$$u = \sum_{l=1}^{k} g^{(l)}, \quad g^{(l)} = \sum_{i=1}^{n_l} w_i^{(l)} \phi_i^{(i)}. \tag{11}$$

To simplify the notation, we will not distinguish explicitly between the functions $u$ or $g^{(l)}$ and its vector representations $u_k$ or $w^{(l)}$, but denote them by their vector representations only. Here and in the following, we denote representations in terms of the generating system $E_k$ by the superscript $E$. The length of $w_k^E$ is

$$n_k^E = \sum_{l=1}^{k} n_l, \tag{12}$$

which is in the 1D case about twice, in the 2D case about 4/3 times, and in the 3D case about 8/7 times larger than the length of the vectors for the basis representation above. This is due to the geometric rate of decrease of the number of grid points from fine to coarse levels, with factors of 1/2, 1/4, and 1/8 for 1D, 2D, and 3D, respectively. The generalization of this concept to higher dimensions is straightforward.

Note that the representation of $u$ in terms of $E_k$ is not unique. In general, there exists an $n_{k-1}^E$-dimensional variety of level-wise decompositions of $u \in V_k$. However, for a given representation $w_k^E$ of $u$ in $E_k$, we can easily compute its unique representation $u_k$ with respect to $B_k$. This transformation from $w_k^E$ to $u_k$ resembles a mapping $S_k^E : \mathbb{R}^{n_k^E} \to \mathbb{R}^{n_k}$, which can be described by the product

$$S_k^E = S_k^{k-1,E} \cdot S_{k-1}^{k-2,E} \cdot S_{k-2}^{k-3,E} \cdot \ldots \cdot S_3^{2,E} \cdot S_2^{1,E} = \prod_{l=k}^{2} S_l^{l-1,E} \tag{13}$$

of sparse triangular $(n_k^E - n_{l-1}^E) \times (n_k^E - n_{l-2}^E)$-matrices $S_l^{l-1,E}$. We then have

$$u_k = S_k^E w_k^E. \tag{14}$$

Let $P_{l-1}^l$ be the $d$-linear interpolation (prolongation) operator that maps $V_{l-1}$ to $V_l$, $l = k,..,2$. Let $R_l^{l-1}$ be the restriction operator from $V_l$ to $V_{l-1}$ defined as the adjoint of $P_{l-1}^l$, $R_l^{l-1}=(P_{l-1}^l)^T$, $l = k,..,2$. The transformation matrices $S_l^{l-1,E}$, $l = k,..,2$, can be expressed by means of these intergrid transfer operators as

$$S_l^{l-1,E} = \begin{pmatrix} P_{l-1}^l & I_l & 0 \\ 0 & 0 & I_{k,l+1} \end{pmatrix}. \tag{15}$$

$I_{k,l+1}$ is the identity operator of dimension $\sum_{i=k}^{l+1} n_i$. Due to the generating system approach, these transformation matrices $S_l^{l-1,E}$ cannot be inverted uniquely. Nevertheless, the transposed matrices $(S_l^{l-1,E})^T$, $l = k,..,2$, are defined by

$$(S_l^{l-1,E})^T = \begin{pmatrix} R_l^{l-1} & 0 \\ I_l & 0 \\ 0 & I_{k,l+1} \end{pmatrix}. \tag{16}$$

Note that the identity relation in the basis representation $B_k$ defines an equivalence relation $\equiv$ in $E_k$ that is different from the simple vector equivalence. Thus, we can identify two different but equivalent representations in $E_k$ of a function in $V_k$:

$$w_k^{E,1} \equiv w_k^{E,2} \;\; \Leftrightarrow_{def} \;\; S_k^E w_k^{E,1} = S_k^E w_k^{E,2}. \tag{17}$$

We see that two different representations $w_k^{E,1}$ and $w_k^{E,2}$ of $u_k$ differ only by a vector $v_k^E = w_k^{E,1} - w_k^{E,2}$ that is in $Ker(S_k^E)$, the kernel of $S_k^E$, i.e. $S_k^E v_k^E = 0$.

**2.2. Linear systems.** Using the nodal basis $B_k$, the Galerkin approach leads to the linear system of equations $L_k u_k = f_k$, where $(L_k)_{(i,j)} = a(\phi_i^{(k)}, \phi_j^{(k)})$ and $(f_k)_{(j)} = (f, \phi_j^{(k)})$, $i, j = 1, .., n_k$. Alternatively, we can use the discrete energy functional $1/2 \cdot u_k^T L_k u_k - u_k^T f_k$ to describe the problem as energy minimization in the space $V_k$ or $\mathbb{R}^{n_k}$, respectively. For the generating system $E_k$, the Galerkin approach leads to the system of equations

$$L_k^E w_k^E = f_k^E, \tag{18}$$

where

$$\begin{array}{l} (L_k^E)_{(i_l,j_l)}^{(l)} = a(\phi_{i_l}^{(l)}, \phi_{j_l}^{(l)}), \\ (f_n^E)_{(j_l)}^{(l)} = (f, \phi_{j_l}^{(l)}), \end{array} \quad l = 1, .., k, \; i_l, j_l = 1, .., n_l. \tag{19}$$

Alternatively, we can use the discrete energy functional $1/2 \cdot (w_k^E)^T L_k^E w_k^E - (w_k^E)^T f_k^E$ to describe the problem as (nonunique) minimization in $\mathbb{R}^{n_k^E}$.

The system $L_k^E w_k^E = f_k^E$ has the following properties. The matrix $L_k^E$ is semidefinite. It has the same rank as $L_k$. Thus $n_{k-1}^E = n_k^E - rank(L_k)$ eigenvalues are zero. The system is solvable because the right-hand side is constructed in a consistent manner. However, it has not just one unique solution, but rather numerous different solutions. For two different solutions $w_k^{E,1}$ and $w_k^{E,2}$,

$$S_k^E w_k^{E,1} = S_k^E w_k^{E,2} = u_k \tag{20}$$

holds, where $u_k$ is the unique solution of the system $L_k u_k = f_k$. This is a direct consequence of (21). Therefore, it is sufficient to compute just one solution of the enlarged semidefinite system to obtain, via $S_k^E$, the unique solution of the system $L_k u_k = f_k$.

Note that the enlarged matrix $L_k^E$ contains the submatrices $L_l$ that arise from the use of the standard basis $B_l$, $l = 1, .., k$. A similar property holds for the right-hand sides. The matrix that stems from the hierarchical basis discretization is also contained in $L_k^E$. For the definition of the hierarchical basis, the resulting linear system and further explanations, see [22].

Instead of using the bilinear form $a(.,.)$ and the linear form $(f,.)$, which involves explicit integration on all levels $l = 1, .., k$, we can express $L_k^E$ and $f_k^E$ in a simple product type representation that involves only the intergrid transfer operators $P_{l-1}^l$ and $R_l^{l-1}$, $l = k, .., 2$ and the discretizations of $L$ and $f$ on the finest level. Since the transformation from $E_k$ to $B_k$ is expressed by $u_k = S_k^E w_k^E$, we get

$$L_k^E = (S_k^E)^T L_k S_k^E, \qquad f_k^E = (S_k^E)^T f_k. \tag{21}$$

This can be interpreted as "semidefinite" preconditioning of the system $L_k u_k = f_k$. It is then easy to see that $1/2 \cdot (w_k^E)^T L_k^E w_k^E - (w_k^E)^T f_k^E = 1/2 \cdot u_k^T L_k u_k - u_k^T f_k$.

Note that the matrix $L_k^E$ is relatively densely populated. In practical applications, however, it is not necessary to know the matrix explicitly. Often only a matrix vector multiplication $L_k^E w_k^E$ is needed, which allows the product type representation (21) of $L_k^E$ to be exploited, so that this matrix vector multiplication can be computed in $O(n_k^E) = O(n_k)$ operations.

Note further that the restriction operator $R_k^{k-1}$ in the mapping $(S_k^E)^T$ is consistent with the Galerkin approach. If, for example, $f_k$ in space $V_k$ is given as $f_k = \sum_{i=1}^{n_k}(f,\phi_i^{(k)})\phi_i^{(k)}$, then

$$R_k^{k-1} f_k = \sum_{i=1}^{n_{k-1}} (f, \phi_i^{(k-1)})\phi_i^{(k-1)}. \tag{22}$$

This is an important fact that should be exploited in practical computations. The explicit calculation of the integral $(f,.)$ can be avoided on the coarser levels $l < k$. The linear combination of the coefficient values of $f_k$ according to the restriction stencil is sufficient to reproduce the integral on the next coarser level. Consequently, if $f_k$ on level $k$ has been computed, then $f_k^E$ can be gained easily using $(S_k^E)^T$.

Consider, as an example for the product type representation (21), the case of only two spaces $V_{k-1}$, $V_k$. Since the standard basis representation is computed from $w_k^E{=}(w^{(k-1)}, w^{(k)})^T$ by

$$u_k = \begin{pmatrix} P_{k-1}^k & I_k \end{pmatrix} \begin{pmatrix} w^{(k-1)} \\ w^{(k)} \end{pmatrix}, \tag{23}$$

then the system $L_k^E w_k^E = f_k^E$ can be rewritten as

$$\begin{pmatrix} R_k^{k-1} \\ I_k \end{pmatrix} L_k \begin{pmatrix} P_{k-1}^k & I_k \end{pmatrix} w_k^E = \begin{pmatrix} R_k^{k-1} \\ I_k \end{pmatrix} f_k, \tag{24}$$

and we see that

$$L_k^E = \begin{pmatrix} R_k^{k-1} L_k P_{k-1}^k & R_k^{k-1} L_k \\ L_k P_{k-1}^k & L_k \end{pmatrix}, \qquad f_k^E = \begin{pmatrix} R_k^{k-1} f_k \\ f_k \end{pmatrix}. \tag{25}$$

Note that $R_k^{k-1} L_k P_{k-1}^k$ is just the Galerkin coarse grid matrix $L_{k-1}$.

¿From this two-level example, we see that the system $L_k u_k = f_k$ arising from the finite element discretization using the basis $B_k$ can instead be written in terms of the generating system $E_k$ by the following steps:

Apply $\begin{pmatrix} R_k^{k-1} \\ I_k \end{pmatrix}$ to both sides of the equation and set $u_k = \begin{pmatrix} P_{k-1}^k & I_k \end{pmatrix} w_k^E$.

In the first step, new equations are added to the system that are constructed from the original equations by forming linear combinations with the weights of the MG-restriction stencil. The second relation defines the new unknowns and their relation to the original unknowns. A natural two-level decomposition $w_k^E$ of $u_k$ is defined implicitly, which is not unique but nevertheless consistent with the Galerkin approach.

**3. Iterative methods for the semidefinite system.** In this section, we study iterative methods for the solution of the semidefinite system. We show that the diagonally preconditioned conjugate gradient method is equivalent to the BPX- or MDS-preconditioned CG-approach for the standard system using $B_k$. The generalized condition number of $L_k^E$ is the same as for the BPX- or MDS-preconditioned matrix $L_k$. Therefore, proofs for the condition of BPX and MDS are directly valid for the semidefinite system. Results of Oswald, Zhang, Xu, Bornemann and Yserentant for BPX establish a condition number of the order $O(1)$ for $L_k^E$. Furthermore, we show that the Gauss-Seidel method (GS) and its symmetric counterpart (SGS) are equivalent to multigrid methods for the standard system. A Gauss-Seidel type relaxation itself can be used as preconditioner for the semidefinite system. This results in MG-CG for the standard basis system.

We want to achieve a solution of the semidefinite system $L_k^E w_k^E = f_k^E$. Because $L_k^E$ is not invertible, we cannot use a conventional direct solver. Instead, we can consider basic iterations $w_k^E := w_k^E + C_k^E(f_k^E - L_k^E w_k^E)$ with positive-definite (or semidefinite) matrices $C_k^E$. We focus on the conjugate gradient type accelerations of such iterations where $C_k^E$ is a diagonal matrix preconditioner. Furthermore, we study Gauss-Seidel type relaxations, both as an iteration itself (with $C_k^E$ as inverse of the lower triangular part of $L_k^E$) and as a preconditioner for the conjugate gradient method.

It is well known that these methods iteratively reduce the energy of the system. They differ in the employed descent directions. While for the conjugate gradient method the descent directions are based on the residual, the Gauss-Seidel iteration descends along the coordinate axes. It is well known that CG is sensitive to scaling, and that its convergence rate depends on the condition number of the system. On the other hand, Gauss-Seidel relaxations are invariant under scaling of the system. See [12] for further explanations on descent methods.

**3.1. Conjugate gradient iteration for the semidefinite system.** Here we consider the (diagonally preconditioned) conjugate gradient method for the semidefinite system. The BPX-preconditioner of Bramble, Pasciak and Xu and its generalization MDS applied to the standard basis system can be described directly in terms of $E_k$ with simple diagonal matrices $C_k^E$ of full rank $n_k^E$. This is also true of the HB-preconditioner introduced by Yserentant. It can be described with a simple diagonal matrix $C_k^E$ of deficient rank $n_k$. If the functions of the generating system are properly scaled, then the $C_k^E$ for BPX or MDS becomes the identity, and the conjugate gradient method for the semidefinite system $L_k^E w_k^E = f_k^E$ is directly equivalent to BPX-CG or MDS-CG for the definite system $L_k u_k = f_k$. Otherwise, this scaling is done explicitly by the respective matrix $C_k^E$.

**3.1.1. The residual and the conjugate gradient iteration .** In the CG method, a major task is to compute the residual

$$r_k^E = f_k^E - L_k^E w_k^E \tag{26}$$

of the semidefinite system for a given $w_k^E$. Here, the product type representation (21) of $L_k^E$ and $f_k^E$ can be exploited. We get

$$r_k^E = (S_k^E)^T f_k - (S_k^E)^T L_k S_k^E w_k^E, \tag{27}$$

and, with $u_k = S_k^E w_k^E$ and $r_k = f_k - L_k u_k$, we have

$$r_k^E = (S_k^E)^T (f_k - L_k u_k) = (S_k^E)^T r_k. \tag{28}$$

Consider as an example once more the two-level case (23)-(25). The residual of the semidefinite system now reads

$$r_k^E = \begin{pmatrix} R_k^{k-1} f_k \\ f_k \end{pmatrix} - \begin{pmatrix} R_k^{k-1} L_k P_{k-1}^k w^{(k-1)} + R_k^{k-1} L_k w^{(k)} \\ L_k P_{k-1}^k w^{(k-1)} + L_k w^{(k)} \end{pmatrix}. \tag{29}$$

If we use the product type representation (21) of $L_k^E$ and $f_k^E$, we are able to implement the computation of the residual efficiently. We have

$$r_k^E = \begin{pmatrix} R_k^{k-1} \\ I_k \end{pmatrix} f_k - \begin{pmatrix} R_k^{k-1} \\ I_k \end{pmatrix} L_k \begin{pmatrix} P_{k-1}^k & I_k \end{pmatrix} \begin{pmatrix} w^{(k-1)} \\ w^{(k)} \end{pmatrix}, \tag{30}$$

and the same result is obtained if we compute first $u_k = P_{k-1}^k w^{(k-1)} + w^{(k)}$, then the fine level residual $r_k = f_k - L_k u_k$, then finally

$$r_k^E = \begin{pmatrix} R_k^{k-1} r_k \\ r_k \end{pmatrix}. \tag{31}$$

This avoids computing the expressions individually on the coarse and fine levels, which differ only by the coarse grid term $R_k^{k-1}$.

Note that all representations $w_k^E$ in $E_k$ of a given function $u_k$ have exactly the **same residual vector** $r_k^E$. Therefore, even if our generating system allows many nonunique representations for a function $u_k$, it is unique with respect to the residual. In fact,

$$w_k^{E,1} \equiv w_k^{E,2} \Rightarrow L_k^E w_k^{E,1} = L_k^E w_k^{E,2}. \tag{32}$$

In the following, let $C_k^E$ be a diagonal matrix. The central step in the CG method for solving $L_k^E w_k^E = f_k^E$ is the computation of the residual and its scalar product. With $C_k^E$-preconditioning, we must compute $(r_k^E)^T C_k^E r_k^E$. Since $r_k^E = (S_k^E)^T r_k$ (see (28)), then

$$(r_k^E)^T C_k^E r_k^E = r_k^T S_k^E C_k^E (S_k^E)^T r_k. \tag{33}$$

Hence, the $C_k^E$-preconditioned CG algorithm for solving $L_k^E w_k^E = f_k^E$ is equivalent to the conjugate gradient algorithm with preconditioner $S_k^E C_k^E (S_k^E)^T$ for solving $L_k u_k = f_k$.

**3.1.2. BPX- and related preconditioners.** In the following, we show that the BPX-, MDS-, and HB-preconditioners can be expressed by the operator $S_k^E C_k^E (S_k^E)^T$ for appropriate diagonal matrices $C_k^E$. In [23], the finest level system $L_k u_k = f_k$ is treated by CG-accelerated iterations with a general self-adjoint, positive-definite preconditioning operator $C_k : V_k \to V_k$. Specific choices for $C_k$ for the BPX- and HB-preconditioners were given by the respective matrices $C_k^X$ and $C_k^Y$ defined as follows:

$$C_k^X r_k = L_0^{-1} \sum_{i=1}^{n_0} (r_k, \phi_i^{(0)}) \phi_i^{(0)} + \sum_{l=1}^{k} \sum_{i=1}^{n_l} \frac{1}{d_i^{(l)}} (r_k, \phi_i^{(l)}) \phi_i^{(l)} \qquad (BPX) \tag{34}$$

$$C_k^Y r_k = L_0^{-1} \sum_{i=1}^{n_0} (r_k, \phi_i^{(0)}) \phi_i^{(0)} + \sum_{l=1}^{k} \sum_{i=n_{l-1}+1}^{n_l} \frac{1}{d_i^{(l)}} (r_k, \phi_i^{(l)}) \phi_i^{(l)} \qquad (HB) \tag{35}$$

with the scaling factors $d_i^{(l)} = 4^l (1, \phi_i^{(l)})$. Yserentant suggested replacing the scaling factors $d_i^{(l)}$ by $\hat{d}_i^{(l)} = a(\phi_i^{(l)}, \phi_i^{(l)})$, which leads to the preconditioner $C_k^Z$ defined as

$$C_k^Z r_k = L_0^{-1} \sum_{i=1}^{n_0} (r_k, \phi_i^{(0)}) \phi_i^{(0)} + \sum_{l=1}^{k} \sum_{i=1}^{n_l} \frac{1}{\hat{d}_i^{(l)}} (r_k, \phi_i^{(l)}) \phi_i^{(l)}. \qquad (MDS) \tag{36}$$

$C_k^Z$ is further analyzed by Zhang in [25] as a multilevel diagonal scaling variant of the multilevel additive Schwarz method. This preconditioner can be expected to work in practice better than BPX for non-model problems, since with $\hat{d}_i^{(l)}$ it takes information into account that reflects more closely the properties of the problem. However, for elliptic problems with constant coefficients, MDS is equivalent to BPX up to a constant multiple.

Note that, for our simplified case of Dirichlet boundary conditions, $V_0 = \{0\}$ and the term $L_0^{-1} \sum_{i=1}^{n_0} (r_k, \phi_i^{(0)}) \phi_i^{(0)}$ vanishes. For more general boundary conditions, a system with the level 0 discretization matrix $L_0$ must be solved here.

These preconditioners can now be expressed easily in terms of our semidefinite system $L_k^E w_k^E = f_k^E$. Computation of the residual $r_k^E = f_k^E - L_k^E w_k^E$ produces

$$r_k^E = (S_k^E)^T r_k = \begin{pmatrix} r^{(1)} & r^{(2)} & . & . & r^{(k-1)} & r^{(k)} \end{pmatrix}^T, \tag{37}$$

where

$$(r^{(l)})_i = (r_k, \phi_i^{(l)}) \phi_i^{(l)}, \quad l = k, .., 1, \quad i = 1, .., n_l. \tag{38}$$

The BPX- and HB- preconditioners resemble just a multiplication of the residual $r_k^E$ of the enlarged system with the matrix $C_k^E$. For all preconditioners introduced above, this matrix $C_k^E$ has diagonal form. The preconditioners differ only in the diagonal entries. Therefore, in terms of the enlarged system, BPX-, MDS- or HB-preconditioning amounts to nothing but diagonal scaling of the residual.

For the BPX-preconditioner $C_k^X$, its analogue $C_k^{E,X}$ has full rank. In fact

$$C_k^{E,X} = diag\left(1/d_i^{(l)}\right). \tag{39}$$

If the functions of our generating system use a special scaling, then $C_k^{E,X} = I_k^E$, and the BPX-preconditioner yields the usual conjugate gradient method on the semidefinite system. This special scaling is expressed by

$$\bar{\phi}_i^{(l)} = h_l^{(2-d)/2} \phi_i^{(l)}, \quad l = k, .., 1, \;\; d = 1, 2, ... \tag{40}$$

Note that, in the 2D case, no scaling is actually necessary.

Since the modified scaling factors $\hat{d}_i^{(l)} = a(\phi_i^{(l)}, \phi_i^{(l)})$ are used in the MDS-scheme, we see that

$$C_k^{E,Z} = diag\left(1/(L_k^E)_{(i,i)}^{(l)}\right). \tag{41}$$

Here, the MDS-preconditioner reduces to just a Jacobi-preconditioner for the semidefinite system (Jacobi-CG). If the functions of our generating system use a special scaling, then $C_k^{E,Z} = I_k^E$, and the MDS-preconditioner yields the usual conjugate gradient method on the semidefinite system. This special scaling is expressed by

$$\hat{\phi}_i^{(l)} = a(\phi_i^{(l)}, \phi_i^{(l)})^{-1/2} \phi_i^{(l)}, \quad l = k, .., 1. \tag{42}$$

Note that Jacobi-iteration on the semidefinite system alone does not converge in general, nor do the simple iterations based on BPX- or MDS-preconditioners. It is the combination of CG and preconditioning that makes the method properly (and quickly) converge.

For the HB-preconditioner $C_k^Y$, its analogue $C_k^{E,Y}$ has deficient rank. Only the $n_k$ diagonal entries associated with the functions appearing in the hierarchical basis are non-zero. (They contain the inverse of the scaling factors $d_i^{(l)}$.) For further details, see [22] and [7].

If we want to express the resulting vector $C_k^{E,J} r_k^E$, $J \in \{X, Y, Z\}$, in terms of the basis $B_k$, we must multiply it by $S_k^E$, which amounts to the explicit level-wise summation of the BXP-, MDS-, or HB-preconditioner. However, using the generating system approach, this summation is avoided, because of the direct, multilevel representation of the functions of $V_k$, implying that the summation is really done implicitly.

It is now possible to characterize the BPX-, MDS-, and HB-preconditioning in terms of $B_k$ by the matrix notation

$$C_k^J = S_k^E C_k^{E,J} (S_k^E)^T, \quad J \in \{X, Y, Z\}. \tag{43}$$

It is easy to see that the scalar products involved in the preconditioned CG-methods for $L_k u_k = f_k$ and for $L_k^E w_k^E = f_k^E$ are the same. Since $r_k^E = (S_k^E)^T r_k$, then

$$(r_k)^T C_k^J r_k = (r_k^E)^T C_k^{E,J} r_k^E, \quad J \in \{X, Y, Z\}. \tag{44}$$

We have seen that the use of the generating system, instead of the standard basis approach for the representation of the functions of $V_k$, allows an easy and simple description of the BPX-, MDS-, and HB-preconditioned conjugate gradient method. The somewhat complicated form of the BPX-, MDS-, and HB-preconditioning reduces in $E_k$ to simple diagonal scaling of the residual $r_k^E$. If we choose specially scaled generating functions in $E_k$, then $C_k^E$ reduces to the identity matrix, which means that only the standard CG-algorithm is performed on the associated semidefinite system.

**3.1.3. Condition number considerations.** Here we consider the condition number of the $C_k^E$-preconditioned semidefinite system. Of course, in general, the conventional condition number is infinity because $L_k^E$ is singular. However, we will show here that the condition numbers of the preconditioned enlarged and original systems are the same, provided we use a generalized condition number that considers the preconditioned matrix restricted to the orthogonal complement of its null space. In fact, each eigenvalue of $C_k L_k$ is also an eigenvalue of $C_k^E L_k^E$, and the rest of the eigenvalues of $C_k^E L_k^E$ are zero. This holds for all symmetric, positive-definite preconditioning matrices $C_k$ with $C_k^E = (S_k^E)^T C_k S_k^E$. It follows from considering the cases $e_k^E \in Ker(S_k^E)$ and $e_k^E \notin Ker(S_k^E)$ and from observing that the relation

$$C_k^E (S_k^E)^T L_k S_k^E e_k^E = \lambda_k^E e_k^E \tag{45}$$

implies the relation

$$S_k^E C_k^E (S_k^E)^T L_k e_k = \lambda_k^E e_k, \tag{46}$$

where $e_k = S_k^E e_k^E$. The generalized condition number that we use is given by

$$\kappa(C_k^E L_k^E) = \lambda_{max}/\lambda_{min}, \tag{47}$$

where $\lambda_{max}$ denotes the largest eigenvalue of $C_k^E L_k^E$ and $\lambda_{min}$ its smallest nonzero eigenvalue. We thus have

$$\kappa(C_k L_k) = \kappa(C_k^E L_k^E). \tag{48}$$

Note that $\kappa(C_k L_k)$ reduces to the usual definition of condition number.

Any results on the condition number of $C_k L_k$ therefore extends directly to the semidefinite, preconditioned matrix $C_k^E L_k^E$, provided $C_k$ is symmetric, positive-definite, as is the case for the BPX-, and MDS-preconditioners. It has been shown by Oswald [16], [17], [18], Xu [20], Zhang [25], [26], and Bornemann and Yserentant [4] that the condition number of the BPX-preconditioner is of the order $O(1)$. So we may conclude that

$$\kappa(C_k^{E,J} L_k^E) = O(1), \quad J \in \{X, Z\}. \tag{49}$$

Thus, if we use specially scaled functions in $E_k$, then $C_k^{E,J} = I_k^E$ and the semidefinite system matrix that now results from the Galerkin approach has a generalized condition number that is independent of $k$.

**3.2. Gauss-Seidel-iteration for the semidefinite system.** Here we consider relaxation type iterations for the semidefinite system. We focus on the Gauss-Seidel method (GS) and its symmetric counterpart (SGS), and show that the multigrid method for the standard system can be interpreted as Gauss-Seidel type iteration for the semidefinite system. The levels used in the multigrid scheme induce a block partitioning of the system, and switching from level to level corresponds to an outer block

Gauss-Seidel iteration for the semidefinite system. The MG-smoothing steps resemble inner Gauss-Seidel iterations for each block. The multigrid V-cycle with one pre- and post-smoothing step by Gauss-Seidel iterations reduces to the SGS method, which is also known as Aitken's double sweep scheme. Other MG-cycle types can also be modeled by different orderings of the block GS-traversal. The case of multiple smoothing steps corresponds to multiple inner iterations. Furthermore, other smoothers can be incorporated in the block GS-method as inner iterations, and the SGS-iteration for the semidefinite system can be used itself as a preconditioner of the CG-method.

As usual, we decompose the semidefinite matrix $L_k^E$ by $L_k^E = F_k^E + G_k^E + (G_k^E)^T$, where $F_k^E$ and $G_k^E$ denote the diagonal and strictly lower triangular parts of $L_k^E$, respectively. Then GS is expressed by using $C_k^{E,GS} := (F_k^E + G_k^E)^{-1}$ and SGS is expressed by using $C_k^{E,SGS} := (F_k^E + G_k^E)^{-T} F_k^E (F_k^E + G_k^E)^{-1}$. Note that $F_k^E + G_k^E$ is generally not symmetric but it is positive definite and invertible. In what follows, we order the unknowns level-wise starting with the coarsest level and the unknowns of each level are ordered lexicographically.

**3.2.1. Gauss-Seidel and multigrid.** Here we demonstrate that Gauss-Seidel type relaxations for the semidefinite system are equivalent to the multigrid method for the standard system. Without loss of generality, we restrict ourselves for simplicity to the two-level case and consider the block system

$$\begin{pmatrix} R_k^{k-1} L_k P_{k-1}^k & R_k^{k-1} L_k \\ L_k P_{k-1}^k & L_k \end{pmatrix} \begin{pmatrix} w^{(k-1)} \\ w^{(k)} \end{pmatrix} = \begin{pmatrix} R_k^{k-1} f_k \\ f_k \end{pmatrix}. \tag{50}$$

The two-level block partitioned Gauss-Seidel iteration (with some inner relaxation scheme like point-Gauss-Seidel) for the semidefinite system then proceeds as follows:

0. Choose some $w_k^{E,0} = (w^{(k),0}, w^{(k-1),0})^T$ as a starting value.

   For $i = 1$ to convergence:

   1. Relax on $L_k w^{(k),i-1} = f_k - L_k P_{k-1}^k w^{(k-1),i-1}$ to obtain $w^{(k),i}$.
   2. Relax on $L_{k-1} w^{(k-1),i-1} = R_k^{k-1} f_k - R_k^{k-1} L_k w^{(k),i}$ to obtain $w^{(k-1),i}$.

Consider now the well-known correction scheme form of the multigrid method. Here, instead of exact solving the coarse grid equation, we only iterate it. It consists of the following steps:

0. Choose some $u_k^0$ as a starting value.

   For $i = 1$ to convergence:

   1. Relax on $L_k u_k^{i-1} = f_k$ to obtain $u_k^i$.
   2. Set $u_{k-1}^{i-1} = 0$ and relax on $L_{k-1} u_{k-1}^{i-1} = R_k^{k-1}(f_k - L_k u_k^i)$ to obtain $u_{k-1}^i$.
   3. Form the correction $u_k^i = u_k^i + P_{k-1}^k u_{k-1}^i$.

It is easy to see that principally the same equations arise in both of these schemes. The coarse grid equations are exactly the same, and although the fine grid equations are not, their difference is compensated for in Step 3 of the CS-scheme. This correction step performs just the analogue of the transformation $S_k^E$ from the nonunique representation of the approximation of the solution in $E_k$ to its unique representation in the standard basis $B_k$. Therefore, in the CS-scheme, the approximation to the solution is represented after each coarse grid correction step in standard basis. This is not necessary for the block Gauss-Seidel scheme because it works directly with the nonunique representation based on the generating system.

Thus, the only difference between these two schemes is their representation. They are otherwise equivalent. The interpretation of the block entries of $w_k^E$ is therefore clarified if the correction scheme of the multigrid-method is kept in mind: In our two-level example we can see $w^{(k)}$ as some sort of implicit approximation of the solution on level $k$ and $w^{(k-1)}$ as an implicit coarse grid correction to it. In this sense, our method and McCormick's approach of Unigrid and PML (multilevel projection method) are in similar spirit, see [14], [15].

The V-cycle with one pre- and one post-smoothing step is equivalent to two Gauss-Seidel iterations for the semidefinite system, where the second step is performed in reversed ordering. This is just the well known Aitken double sweep or symmetric Gauss-Seidel method (SGS). Multiple pre- and post-smoothing iterations can now be modeled by block sweeps over the semidefinite system with multiple inner iterations. Of course, other smoothers can also be applied in the inner iteration and the outer block iteration can be reordered so that other MG-cycling strategies are implemented.

The Gauss-Seidel- or SGS-iteration can be considered as a method for the minimization of the energy functional, where the search directions are just the coordinate axes. Using the enlarged generating system, we can interpret the unknowns of the coarse grid equations of the multigrid method as additional coordinate axes which are given via $(S_k^E)^T$ as linear combinations of the fine grid coordinate axes. The coarse grid correction now gives additional search directions for the minimization of the energy, which makes multigrid fast to converge.

Furthermore, it is easy to see that the HB-MG-Method of [3] can be interpreted in the extended system as well: Iteration on the unknowns of $w^{(l)}, l = k, .., 2$, which belong to grid points that are contained in the next coarser grid $\Omega_{l-1}$, is simply omitted, and its values are frozen there.

Note that we obtain the usual Gauss-Seidel iteration on the fine grid system by omitting iteration over the blocks that are associated with the coarser levels. This would mean that the associated values of $w_k^E$ values are never changed. But if we transform the resulting solution to its standard basis representation, we obtain exactly the same result as for an iteration on the fine grid system only. For remarks on an estimate concerning the condition number of the SGS-preconditioned semidefinite matrix, we refer to [7].

**4. Numerical experiments.** Consider the simple model problem

$$\Delta u = f \; on \; \Omega = (0,1)^d, \; d = 1, 2, \tag{51}$$

with Dirichlet boundary conditions, where $\Delta = \sum_{i=1}^d \partial^2/\partial x_i^2$.

**4.1. Eigenvalues and condition numbers.** The numerically computed eigenvalues and generalized condition numbers of the semidefinite matrix $L_k^E$ and the MDS-preconditioned semidefinite matrix $C_k^{E,Z} L_k^E$ are shown for the 1D case in Table 1. (The BPX-preconditioned matrix $C_k^{E,X} L_k^E$ has the same condition numbers and, up to the factor 2, the same eigenvalues.) Here, $s$ and $r$ denote the respective size and rank of the matrix.

TABLE 1
*Eigenvalues and generalized condition numbers in the 1D case.*

| | | | $L_k^E$ | | | | $C_k^{E,Z} L_k^E$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| k | $s$ | $r$ | $\lambda_{min}$ | $\lambda_{max}$ | $\kappa$ | $\frac{\kappa(k-1)}{\kappa(k)}$ | $\lambda_{min}$ | $\lambda_{max}$ | $\kappa$ | $\frac{\kappa(k-1)}{\kappa(k)}$ |
| 1 | 1 | 1 | 4.0000 | 4.0000 | 1.0000 | - | 1.0000 | 1.0000 | 1.0000 | - |
| 2 | 4 | 3 | 5.5279 | 14.472 | 2.6180 | 2.618 | 1.0000 | 2.0000 | 2.0000 | 2.0000 |
| 3 | 11 | 7 | 6.6046 | 31.206 | 4.7249 | 1.8048 | 1.0000 | 2.8666 | 2.8666 | 1.4333 |
| 4 | 26 | 15 | 7.1989 | 63.575 | 8.8312 | 1.8690 | 1.0000 | 3.4839 | 3.4839 | 1.2156 |
| 5 | 57 | 31 | 7.5073 | 127.77 | 17.019 | 1.9272 | 1.0000 | 3.9834 | 3.9834 | 1.1434 |
| 6 | 120 | 63 | 7.6642 | 255.88 | 33.386 | 1.9616 | 1.0000 | 4.3891 | 4.3891 | 1.1018 |
| 7 | 147 | 127 | 7.7433 | 511.94 | 66.114 | 1.9803 | 1.0000 | 4.7239 | 4.7239 | 1.0763 |
| 8 | 502 | 255 | 7.7831 | 1024.00 | 137.57 | 1.9900 | 1.0000 | 5.0029 | 5.0029 | 1.0591 |

Note that $L_k^E$ has a generalized condition number of $O(h_k^{-1})$, which is by a factor $h_k^{-1}$ better than the condition number of the standard basis matrix $L_k$. The diagonally preconditioned matrix $C_k^{E,Z} L_k^E$, however, has the constant 1.0 as the smallest eigenvalue and a very slowly growing largest eigenvalue. Its condition number actually behaves like $O(1)$.

Using Fourier analysis [19], it is easy to see that the eigenvalue

$$\lambda_{N/2} = 2/h_k \tag{52}$$

of $L_k$ is also an eigenvalue of $L_k^E$. Since $C_k^{E,Z}$ has the diagonal entries $h_l/2,\ l = 1, .., k$ (c.f. (41)), it is clear that 1.0 is an eigenvalue of $C_k^{E,Z} L_k^E$. This is in fact the smallest nonzero eigenvalue of $C_k^{E,Z} L_k^E$, although we have yet to prove it. Applying Gerschgorin's theorem, it is easy to show that the largest eigenvalue $\lambda_{max}$ of $C_k^{E,Z} L_k^E$ is bounded from above by a constant, which can be estimated to be 6.8284. This is due to the fact that, at least in the 1D case, the entries of the spectrally equivalent matrix $(C_k^{E,Z})^{1/2} L_k^E (C_k^{E,Z})^{1/2}$ decrease in a certain geometric progression.

For the 2D case, the condition number and the eigenvalues of the MDS preconditioned matrix $C_k^{E,Z} L_k^E$ are shown in Table 2. (The BPX-preconditioner again results in the same condition numbers, although the eigenvalues are now multiplied by the factor 8/3.)

TABLE 2
*Eigenvalues and generalized condition numbers for $C_k^{E,Z} L_k^E$ in the 2D case.*

| k | $s$ | $r$ | $\lambda_{min}$ | $\lambda_{max}$ | $\kappa$ | $\frac{\kappa(k-1)}{\kappa(k)}$ |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1.0000 | 1.0000 | 1.0000 | - |
| 2 | 10 | 9 | 0.8232 | 1.6971 | 2.0615 | 2.0615 |
| 3 | 59 | 49 | 0.7690 | 2.2785 | 2.9629 | 1.4372 |
| 4 | 284 | 225 | 0.7548 | 2.7102 | 3.5906 | 1.2118 |
| 5 | 1245 | 961 | 0.7512 | 3.0576 | 4.0702 | 1.1336 |

Once again, using Fourier analysis, it can be shown that the eigenvalues

$$\lambda_{n_1,n_2} = \frac{2}{3}(4 - \cos(\pi n_1 h_k) - \cos(\pi n_2 h_k) - 2\cos(\pi n_1 h_k)\cos(\pi n_2 h_k)), \tag{53}$$

of $L_k$, $n_1 = 1,..,N-1$, $n_2 = N/2$, and $n_2 = 1,..,N-1$, $n_1 = N/2$, where $N = 2^k$, $h_k = 1/N$, are also eigenvalues of $L_k^E$. Since $C_k^{E,Z}$ is a diagonal matrix with the constant diagonal entries 3/8, it follows that $3/8 \cdot \lambda_{n_1,N/2}$ and $3/8 \cdot \lambda_{N/2,n_2}$ are eigenvalues of $C_k^{E,Z} L_k^E$. In fact (compare Table 2), the smallest eigenvalue of $C_k^{E,Z} L_k^E$ appears to be $3/8 \cdot \lambda_{(1,N/2)} = 3/8 \cdot \lambda_{(N/2,1)}$. (Again we have yet to prove this.) Hence,

$$\begin{aligned} \lambda_{min}(C_k^{E,Z} L_k^E) &= \frac{1}{4}(4 - \cos(\pi h_k) - \cos(\pi h_k N/2) - 2\cos(\pi h_k)\cos(\pi h_k N/2)) \\ &= 1 - \frac{1}{4}\cos(\frac{\pi}{2^k}). \end{aligned} \tag{54}$$

The largest eigenvalue grows very moderately. However, Gerschgorin's theorem can no longer be used since the entries of $(C_k^{E,Z})^{1/2} L_k^E (C_k^{E,Z})^{1/2}$ no longer decrease geometrically for all rows. Note that in the 2D case the standard generating functions in $E_k$ are already specially scaled with respect to $h_l$. $C_k^{E,Z}$-preconditioning results only in a scaling by the constant factor 3/8.

We now consider the SGS-CG approach. We computed the eigenvalues and generalized condition numbers of the preconditioned matrix $C_k^{E,SGS} L_k^E$. The results for the 1D case are shown in Table 3. Compared to the BPX- and MDS-preconditioner, the

TABLE 3
*Eigenvalues and generalized condition numbers for $C_k^{E,SGS} L_k^E$ in the 1D case.*

| k | $s$ | $r$ | $\lambda_{min}$ | $\lambda_{max}$ | $\kappa$ | $\frac{\kappa(k-1)}{\kappa(k)}$ |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1.0000 | 1.0000 | 1.0000 | - |
| 2 | 4 | 3 | 0.8125 | 1.0000 | 1.2308 | 1.2308 |
| 3 | 11 | 7 | 0.8002 | 1.0000 | 1.2496 | 1.0153 |
| 4 | 26 | 15 | 0.8003 | 1.0000 | 1.2495 | 0.9999 |
| 5 | 57 | 31 | 0.8000 | 1.0000 | 1.2500 | 1.0000 |
| 6 | 120 | 63 | 0.8000 | 1.0000 | 1.2500 | 1.0000 |
| 7 | 147 | 127 | 0.8000 | 1.0000 | 1.2500 | 1.0000 |
| 8 | 402 | 255 | 0.8000 | 1.0000 | 1.2500 | 1.0000 |

SGS-preconditioner improves the condition number roughly by a factor 4.

For the 2D case the condition number and eigenvalues of $C_k^{E,SGS} L_k^E$ are shown in table 4.

TABLE 4
*Eigenvalues and generalized condition numbers for $L_k^{E,SGS}$ in the 2D case.*

| k | $s$ | $r$ | $\lambda_{min}$ | $\lambda_{max}$ | $\kappa$ | $\frac{\kappa(k-1)}{\kappa(k)}$ |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1.0000 | 1.0000 | 1.0000 | - |
| 2 | 10 | 9 | 0.9007 | 1.0000 | 1.1102 | 1.1102 |
| 3 | 59 | 49 | 0.8472 | 1.0000 | 1.1804 | 1.0632 |
| 4 | 284 | 225 | 0.8325 | 1.0000 | 1.2012 | 1.0176 |
| 5 | 1245 | 961 | 0.8288 | 1.0000 | 1.2065 | 1.0044 |

Once more, the SGS-preconditioner improves the condition number by about a factor 4 and the formula $\kappa(C_k^{E,SGS} L_k^E) \approx 1 + 1/(4\lambda_{min}(C_k^{E,Z} L_k^E))$ gives a good estimate of its upper bound.

**4.2. Convergence factors and iteration steps.** Here we study the convergence properties of the different preconditioned CG and GS methods. As shown in [1], the CG-method needs at most $s = int\,[1/2\sqrt{\kappa}\ |ln(2/\varepsilon)| + 1]$ steps to reduce the energy of the initial error by a factor $\varepsilon \in (0,1)$, where $\kappa$ is the (generalized) condition number of the preconditioned matrix to be considered. The reduction factor after $s$ steps can be estimated by the quantity $2 \cdot \rho_{cg}^s$, where $\rho_{cg} := \left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)$ is the (worst-case) convergence factor. Note that, due to the sophisticated convergence properties of the conjugate gradient method taking full advantage of the eigenvalue distribution of the preconditioned matrix, the upper bound $2 \cdot \rho_{cg}^s$ may be pessimistic.

We are able to derive estimates of the reduction factors from the computed condition numbers of the above tables. For example, in the case $k$=5, the generalized condition number of the MDS-preconditioner is close to 4.0 (compare Tables 1 and 2), so, we can expect a theoretical convergence factor of about 0.333. Also, since the generalized condition number of the SGS-preconditioner is about 1.25 (compare Tables 3 and 4), we can expect a convergence factor of about 0.0557. If we invest the additional work necessary for the GS-iterations on each level, we get an improvement of the convergence factor by a multiple of about 6.0.

For comparison, we performed numerical experiments to measure the numbers of CG steps necessary to reduce the $L_2$ norm of the residual $r_k^E$ by the factors $10^{-2}$, $10^{-4}$, $10^{-6}$, $10^{-8}$, and $10^{-10}$ for the MDS-preconditioned and the SGS-preconditioned semidefinite system. The results are given in Tables 5 and 6. For the measurements, we directly computed $r_k^E$, the residual of the semidefinite system, which is related to the residual $r_k$ by $(S_k^E)^T$, as in (28). Note that the SGS-preconditioner reduces the number of iteration steps roughly by a factor of 2.5 in comparison to the MDS-preconditioner.

TABLE 5
*1D case: Required number of CG steps for various residual reduction factors.*

| | $C_k^{E,Z} L_k^E$ | | | | | $C_k^{E,SGS} L_k^E$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| k | $10^{-2}$ | $10^{-4}$ | $10^{-6}$ | $10^{-8}$ | $10^{-10}$ | $10^{-2}$ | $10^{-4}$ | $10^{-6}$ | $10^{-8}$ | $10^{-10}$ |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 3 | 4 | 4 | 4 | 4 | 4 | 2 | 3 | 4 | 5 | 5 |
| 4 | 4 | 6 | 8 | 8 | 8 | 2 | 3 | 5 | 7 | 8 |
| 5 | 5 | 8 | 11 | 13 | 14 | 2 | 3 | 5 | 7 | 8 |
| 6 | 5 | 9 | 13 | 16 | 18 | 2 | 3 | 5 | 7 | 8 |
| 7 | 5 | 9 | 13 | 17 | 21 | 2 | 3 | 5 | 7 | 8 |
| 8 | 5 | 10 | 14 | 18 | 22 | 2 | 3 | 5 | 7 | 8 |

TABLE 6
*2D case: Required number of CG steps for various residual reduction factors.*

| | $C_k^{E,Z} L_k^E$ | | | | | $C_k^{E,SGS} L_k^E$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| k | $10^{-2}$ | $10^{-4}$ | $10^{-6}$ | $10^{-8}$ | $10^{-10}$ | $10^{-2}$ | $10^{-4}$ | $10^{-6}$ | $10^{-8}$ | $10^{-10}$ |
| 2 | 2 | 3 | 3 | 3 | 3 | 2 | 3 | 4 | 5 | 6 |
| 3 | 4 | 6 | 8 | 9 | 10 | 2 | 3 | 5 | 6 | 7 |
| 4 | 5 | 8 | 12 | 15 | 18 | 2 | 3 | 5 | 6 | 8 |
| 5 | 5 | 9 | 13 | 16 | 20 | 2 | 3 | 5 | 6 | 8 |

These results should be compared to the performance of the simple iterative method associated with the respective preconditioner, for which the worst-case convergence factor is

$$\rho(I_K^E - C_k^{E,J} L_k^E) = \max_i |1 - \lambda_i(C_k^{E,J} L_k^E)|, \quad J \in \{Z, SGS\}, \tag{55}$$

where $\lambda_i$ denotes the respective eigenvalues. We see from the Tables 1 and 2 directly that the Jacobi iteration for the semidefinite system is divergent, with spectral radius is greater 1. Note that a damping parameter could be introduced to enforce convergence.

For the symmetric Gauss-Seidel iteration on the semidefinite system, which is equivalent to the (1,1)-MG-V-cycle for the standard system, we get

$$\rho_{sgs} = 1 - \lambda_{min}(C_k^{E,SGS} L_k^E) = 1 - 1/\kappa(C_k^{E,SGS} L_k^E). \tag{56}$$

Since the generalized condition number is close to 1.25 (compare Tables 3 and 4), we can expect a theoretical convergence factor of about 0.2. If we invest the additional work necessary for the CG-iteration to gain SGS-CG, we see that the convergence factor is improved by a multiple of about 3.588.

Tables 7 and 8 show the number of steps that were needed by the GS- and SGS-iteration in numerical experiments to reduce the residual $r_k^E$ by the factors $10^{-2}$, $10^{-4}$, $10^{-6}$, $10^{-8}$, and $10^{-10}$. These iterations resemble directly a V-cycle with one post-smoothing step by (lexicographical) GS and zero or one pre-smoothing steps, respectively.

TABLE 7
*1D case: Required number of GS and SGS steps for various residual reduction factors.*

| | $(C^{E,Z})^{1/2} L_k^E (C^{E,Z})^{1/2}$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | GS | | | | | SGS | | | | |
| k | $10^{-2}$ | $10^{-4}$ | $10^{-6}$ | $10^{-8}$ | $10^{-10}$ | $10^{-2}$ | $10^{-4}$ | $10^{-6}$ | $10^{-8}$ | $10^{-10}$ |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 5 | 8 | 11 | 13 |
| 3 | 4 | 6 | 8 | 9 | 11 | 2 | 5 | 8 | 11 | 13 |
| 4 | 4 | 8 | 11 | 14 | 16 | 2 | 5 | 8 | 10 | 13 |
| 5 | 4 | 8 | 12 | 16 | 20 | 2 | 5 | 8 | 10 | 13 |
| 6 | 4 | 8 | 12 | 16 | 20 | 2 | 5 | 8 | 10 | 13 |
| 7 | 4 | 8 | 12 | 16 | 21 | 2 | 5 | 8 | 10 | 13 |
| 8 | 4 | 8 | 12 | 17 | 21 | 2 | 5 | 7 | 10 | 13 |

TABLE 8
*2D case: Required number of GS and SGS steps for various residual reduction factors.*

| | $(C^{E,Z})^{1/2} L_k^E (C^{E,Z})^{1/2}$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | GS | | | | | SGS | | | | |
| k | $10^{-2}$ | $10^{-4}$ | $10^{-6}$ | $10^{-8}$ | $10^{-10}$ | $10^{-2}$ | $10^{-4}$ | $10^{-6}$ | $10^{-8}$ | $10^{-10}$ |
| 2 | 3 | 4 | 6 | 8 | 10 | 2 | 4 | 6 | 8 | 10 |
| 3 | 3 | 6 | 8 | 10 | 12 | 2 | 4 | 7 | 9 | 12 |
| 4 | 4 | 7 | 9 | 11 | 14 | 2 | 4 | 7 | 9 | 12 |
| 5 | 4 | 7 | 10 | 14 | 18 | 2 | 5 | 7 | 10 | 12 |

**4.3. Efficiency considerations.** To compare the different methods with respect to their efficiency, we restrict ourselves to the 2D model problem. BPX/MDS-CG is realized by a multigrid V-cycle followed by a CG step on the fine grid equations, where explicit smoothing iterations are omitted in the V-cycle. Note that smoothing now only takes place implicitly by simply (weighted) summing of the residuals on all levels. SSOR-CG is implemented by a multigrid V-cycle with one pre- and one post-smoothing iteration (here we use lexicographical Gauss-Seidel) followed by a CG-step on the fine grid equations.

Table 9 shows the number of operations per fine grid point needed by the major processes of the methods we consider. The operations $+, -, \times, /$, are all counted equal. Here, $C_{gs}$ denotes the number of operations per grid point of one Gauss-Seidel smoothing

TABLE 9
*2D case: Number of operations per fine grid point for the major algorithmic processes.*

| $C_{gs}$ | $C_t$ | $C_{cg}$ |
|---|---|---|
| 12 | 12 | 15 |

iteration in the V-cycle, $C_t$ that necessary for the intergrid transfer operators in the V-cycle, and $C_{cg}$ that necessary for the CG-method on the fine grid equations. Compare also [6] for the operation count of the different MG-components and [1] for the CG-operation count. The number, *op*, of operations per grid point for the different methods itself is then given in Table 10.

TABLE 10
*2D case: Number of operations per fine grid point for the different algorithms.*

| | BPX-CG | SGS | SGS-CG |
|---|---|---|---|
| | $C_t + C_{cg}$ | $2 \cdot C_{gs} + C_t$ | $2 \cdot C_{gs} + C_t + C_{cg}$ |
| op | 27 | 36 | 51 |

The efficiency of these schemes is defined as the value

$$eff = -op/ln(\rho), \tag{57}$$

which gives the number of operations per fine grid point necessary to reduce the norm of the residual at least by a factor $e$. For the case $k = 5$, Table 11 shows the theoretical convergence factors $\rho_{th}$ and the resulting efficiences derived from the condition numbers in Tables 2 and 4.

For comparison, we ran numerical tests and measured the average reduction factors after enough steps, $s$, to reduce the residual by about $\varepsilon = 10^{-10}$. This average factor is defined to be

$$\rho_{ave} = \left( \frac{|r_k^{E,s}|_2}{|r_k^{E,0}|_2} \right)^{1/s} \approx \varepsilon^{1/s}. \tag{58}$$

For the results, see also Table 11.

All three considered methods perform comparably with respect to efficiency even though their convergence rates differ strongly. BPX-CG is slightly (by about a multiple of 1.15) less efficient than SGS, which in turn is slightly (by about a multiple of 1.15) less efficient than SGS-CG. This shows that it is possible to speed up multigrid somewhat by

TABLE 11
*2D case: Convergence factors and efficiencies for the different algorithms with $k = 5$.*

| | BPX-CG | SGS | SGS-CG |
|---|---|---|---|
| $\rho_{th}$ | 0.333 | 0.200 | 0.0557 |
| $eff$ | 24.58 | 22.37 | 17.66 |
| $\rho_{ave}$ | 0.355 | 0.203 | 0.074 |
| $eff$ | 26.06 | 22.58 | 19.68 |

CG. Note, however, that multigrid allows other relaxation orderings and other choices for the number of pre- and post-smoothing steps, which might allow for improvement in its efficiency. BPX-CG may also be improved by additional Gauss-Seidel iterations in the BPX-preconditioning process.

**5. Concluding remarks.** In this paper we have introduced the idea of using a generating system instead of the usual basis approach, which allows for the direct use of multilevel decompositions of a function. A Galerkin scheme based on the generating system results in a semidefinite matrix equation.

We showed that modern multilevel methods can be interpreted as standard iterative methods for the semidefinite system. The BPX- or MDS-preconditioner can be seen as simple diagonal scaling of the semidefinite system that resembles a Jacobi iteration step. Multigrid algorithms are equivalent to Gauss-Seidel iterations and can be used as a preconditioners for CG as well, resulting in a substantial improvement of the convergence rate.

Of course, an efficient implementation of these methods must follow the traditional way that is already employed in the multigrid method. ¿From this point of view, we have presented nothing really new. However, the interpretation of different multilevel methods in term of a generating system as standard iteration techniques on a semidefinite system fully clarifies the relations and differences between them and provides a simple and natural approach to their general treatment. We believe that the development of new multilevel methods will be substantially simplified by this perspective.

Generalizations of the semidefinite system can be derived using coarse grid spaces other than the standard coarse grid spaces used here. For example, $E_k$ can be enlarged to contain all possible coarse grid basis functions with respect to standard coarsening and semi-coarsening in all coordinate directions, allowing multilevel methods to be derived for sparse grids [24], [9], [8].

## REFERENCES

[1] O. Axelsson and V. Barker, *Finite element solutions of boundary value problems*, Academic Press, 1984.
[2] O. Axelsson and P. Vassilevski, *Algebraic multilevel preconditioning methods*, Numerische Mathematik, 56 (1989), pp. 157–177.
[3] R. Bank, T. Dupont, and H. Yserentant, *The hierarchical basis multigrid method*, Num. Math., 52 (1988), pp. 427–458.
[4] F. Bornemann and H. Yserentant, *A basic norm equivalence for the theory of multilevel methods*, ZIB, Berlin, preprint 92-1, 1992.

[5] J. Bramble, J. Pasciak, and J. Xu, *Parallel multilevel preconditioners*, Math. Comp., 31 (1990), pp. 333–390.

[6] M. Griebel, *Zur Lösung von Finite-Differenzen- und Finite-Element-Gleichungen mittels der Hierarchischen-Transformations-Mehrgitter-Methode*, TU München, Institut f. Informatik, TUM-I9007; SFB-Report 342/4/90 A, 1990.

[7] ——, *Multilevel algorithms considered as iterative methods on indefinite systems*, TU München, Institut f. Informatik, TUM-I9143; SFB-Report 342/29/91 A, 1991.

[8] ——, *Parallel multigrid methods on sparse grids*, in Int. Series of Numer. Mathematics, Vol. 98, Birkhäuser Verlag, Basel, 1991.

[9] ——, *A parallelizable and vectorizable multi-level algorithm on sparse grids*, in Parallel Algorithms for Partial Differential Equations, Proceedings of the Sixth GAMM-Seminar, Kiel, January 19-21, 1990, W. Hackbusch, ed., Vieweg-Verlag, 1991.

[10] W. Hackbusch, *Multigrid Methods and Applications*, Springer-Verlag, Berlin, Heidelberg, New York, 1985.

[11] R. Kettler, *Analysis and comparison of relaxation schemes in robust multigrid and preconditioned conjugate gradient methods*, in Multigrid Methods, Lecture Notes in Mathematics, Springer-Verlag, 960 (1982).

[12] D. Luenberger, *Introduction to linear and nonlinear programming*, Addison-Wesley, Massachusetts, 1973.

[13] S. McCormick, *Multigrid Methods*, SIAM, Philadelphia, PA, 1987.

[14] ——, *Multilevel adaptive methods for partial differential equations*, SIAM Frontiers in Applied Mathematics 6, Philadelphia, PA, 1989.

[15] ——, *Multilevel projection methods for partial differential equations*, CBMS-NSF Regional conference series in applied mathematics 62, SIAM, Philadelphia, PA, 1992.

[16] P. Oswald, *Approximationsräume und Anwendungen in der Theorie der Multilevel-Verfahren*, Manuskript, FSU Jena, 1991.

[17] ——, *Two remarks on multilevel preconditioners*, FSU Jena, Mathematische Fakultät, Bericht Nr. Math/91/1, 1991.

[18] ——, *Norm equivalencies and multilevel Schwarz preconditioning for variational problems*, FSU Jena, Mathematische Fakultät, Bericht Nr. Math/92/1, 1992.

[19] K. Stüben and U. Trottenberg, *Multigrid methods: fundamental algorithms, model problem analysis and applications*, in Multigrid methods, Lecture Notes in Mathematics, Springer-Verlag, 960 (1982).

[20] J. Xu, *Iterative Methods by Space Decomposition and Subspace Correction: A Unifying Approach*, Report No. AM67, Department of Mathematics, Penn State University, PA, 1990, revised 1992.

[21] ——, *Theory of multilevel methods*, Report No. AM48, Department of Mathematics, Penn State University, 1989.

[22] H. Yserentant, *On the multi-level splitting of finite element spaces*, Numer. Math., 49 (1986), pp. 379–412.

[23] ——, *Two preconditioners based on the multi-level splitting of finite element spaces*, Report No. SC89-9, Konrad-Zuse-Zentrum für Informationstechnik, Berlin, 1990.

[24] C. Zenger, *Sparse Grids*, in Parallel Algorithms for Partial Differential Equations, Proceedings of the Sixth GAMM-Seminar, Kiel, January 19-21, 1990, W. Hackbusch, ed., Vieweg-Verlag, 1991.

[25] X. Zhang, *Multilevel additive Schwarz methods*, New York University, Courant Institute, Department of Computer Science, Tech. Report 582, 1991.

[26] ——, *Multilevel Schwarz methods*, University of Maryland, Department of Computer Science, College Park, MD, Tech. Report CS-TR-2894, UMIACS-TR-92-52, 1992.