Technische Universität Chemnitz-Zwickau

Sonderforschungsbereich 393

Numerische Simulation auf massiv parallelen Rechnern

M. Jung and S.V. Nepomnyaschikh*

Variable Preconditioning Procedures for Elliptic Problems

Preprint SFB393/96-22

Abstract

For solving systems of grid equations approximating elliptic boundary value problems a method of constructing variable preconditioning procedures is presented. The main purpose is to discuss how an efficient preconditioning iterative procedure can be constructed in the case of elliptic problems with disproportional coefficients, e.g. equations with a large coefficient in the reaction term (or a small diffusion coefficient). The optimality of the suggested technique is based on fictitious space and multilevel decomposition methods. Using an additive form of the preconditioners, we introduce factors into the preconditioners to optimize the corresponding convergence rate. The optimization with respect to these factors is used at each step of the iterative process.

The application of this technique to two-level p-hierarchical preconditioners and domain decomposition methods is considered too.

Key words: Iterative methods, preconditioning operators, conjugate gradient methods, additive Schwarz methods, domain decomposition, finite elements

AMS(MOS) subject classification: 65F10, 65N55, 65N30

* This work was partially supported by the German Ministry for Education and Research (BMBF) under the project X223.5 within the scientific cooperation with Russia, by the Netherlands Organization for Scientific Research (NWO), dossiernr. 047.003.017, and by the Russian Basic Research Foundation (RBRF) under the grant 96-01-01665

Preprint-Reihe des Chemnitzer SFB 393

SFB393/96-22 **Dezember 1996**

Authors' address:

Dr. Michael Jung
Faculty for Mathematics
Technical University Chemnitz-Zwickau
D - 09107 Chemnitz, Germany

e-mail: `michael.jung@mathematik.tu-chemnitz.de`

http://www.tu-chemnitz.de/~jung/jung.html

Dr. Sergej V. Nepomnyaschikh
Computing Center
Siberian Branch of Russian Academy of Sciences
Novosibirsk, 630090, Russia

e-mail: `svnep@oapmg.sscc.ru`

1 Introduction

Finite element discretizations of boundary value problems (BVP) lead, in general, to large scale systems of algebraic finite element equations. Usually, these systems are solved by means of the preconditioned conjugate gradient (PCG) method. In recent years, very efficient preconditioners were developed such that the resulting PCG algorithms have a convergence rate which is independent of the discretization parameter, and the cost of arithmetical work per iteration step is proportional to the number of unknowns (see, e.g., [5, 6, 10, 14, 18, 22, 31, 32, 35]). All these preconditioners make use of a sequence of discretizations of the BVP or at least of a sequence of triangulations of the domain in which the BVP is considered. For some practical problems, as e.g. BVP's in domains with a complicated geometry, it is impossible to construct such a sequence of triangulations with a sufficiently coarse grid. Then, the known preconditioners lose their efficiency. To overcome this problem, in [28, 29] preconditioners based on fictitious space and multilevel decomposition methods were proposed. There, the original domain is embedded into a rectangular domain for which a sequence of nested triangulations can be constructed easily. These preconditioners require some mappings between the triangulation of the original domain and the auxiliary meshes, as well as the application of BPX-like preconditioners on the auxiliary meshes. Other techniques for constructing preconditioners on unstructured meshes were proposed in [7, 11, 24, 26, 27, 33]. The construction of preconditioning operators on non-hierarchical grids was considered in [21].

In this paper, we apply the technique described in [28, 29] to the construction of preconditioners for the BVP

$$- \operatorname{div}(p \operatorname{grad} u) + qu = f \quad \text{in } \Omega\,, \qquad u = 0 \quad \text{on } \partial\Omega\,. \tag{1}$$

The BPX-like preconditioners can be written in the additive form

$$C^{-1} = \delta_1 C_1 + \ldots + \delta_l C_l\,, \tag{2}$$

where l denotes the number of triangulations (auxiliary meshes) used. For the Poisson equation $-\Delta u = f$ the parameters δ_k, $k = 1, 2, \ldots, l$, are well-known [10, 14, 31]. In the present paper, we define these parameters for problem (1) analytically. For more general elliptic problems it would be helpful if one had a possibility to compute the parameters δ_k numerically. To do this, we propose four CG-like procedures with preconditioners of the type (2), where the parameters δ_k are variable, i.e. they are computed within each iteration step automatically. Therefore, in each iteration step the preconditioner is changed. We show that these CG-like methods have a convergence rate which is independent of the discretization parameter h as well as of the parameters p and q.

Within each iteration step of the presented methods a set of search directions is computed. This is similar as in the parallel conjugate gradient method analysed in [17]. The difference to our algorithms is that in [17] a fixed preconditioner is applied, and that starting with a set of initial guesses in each iteration step a set of new iterates is computed.

In [4] also algorithms with variable preconditioners were considered. There, the preconditioner is changed when one encounters a stagnation of the convergence rate in generalized conjugate gradient minimum residual methods. However, the technique used there is different from our approach and thus, we will not discuss it further.

The paper is organized as follows: In Section 2, we describe the construction of the additive preconditioner (2) with fixed parameters by means of the fictitious space lemma and multilevel techniques, and we show the optimality of this preconditioner. Section 3 is devoted to the description and analysis of different iterative processes with variable preconditioners. In Section 4, implementation aspects of the proposed algorithms are discussed, and the convergence properties of these algorithms are demonstrated by numerical examples. Here, we apply the new algorithms to the systems of algebraic finite element equations which result from the discretization of problem (1). Furthermore, we use the idea of variable preconditioners in connection with two-level p-hierarchical preconditioners and preconditioners based on domain decomposition techniques. Finally, some conclusions are given.

2 Multilevel preconditioning operators for elliptic problems with parameters

Let $\Omega \subset \mathbb{R}^2$ be a bounded domain with a piecewise smooth boundary Γ which satisfies the Lipschitz condition [34]. In the domain Ω we consider the boundary value problem:

$$\begin{aligned} -\mathrm{div}(p\,\mathrm{grad}u) + qu &= f(x)\,, \qquad x \in \Omega\,, \\ u(x) &= 0\,, \qquad x \in \Gamma\,, \end{aligned} \tag{3}$$

where

$$p = const \geq 0\,, \;\; q = const \geq 0\,, \;\; p + q > 0\,.$$

We introduce a bilinear form $a(u,v)$ and a linear functional $l(v)$ as follows:

$$\begin{aligned} a(u,v) &= \int_\Omega (p\,(\nabla u, \nabla v) + quv)\,d\Omega\,, \qquad \forall u,v \in \overset{\circ}{H}{}^1(\Omega) \\ l(v) &= \int_\Omega fv\,d\Omega\,, \qquad \forall v \in \overset{\circ}{H}{}^1(\Omega)\,. \end{aligned}$$

Here, $f \in H^{-1}(\Omega)$ is assumed.

The generalized solution $u \in \overset{\circ}{H}{}^1(\Omega)$ of the problem (3) is, by definition, a solution to the projection problem

$$u \in \overset{\circ}{H}{}^1(\Omega)\;:\;\; a(u,v) = l(v)\,, \quad \forall v \in \overset{\circ}{H}{}^1(\Omega)\,. \tag{4}$$

Let a positive parameter h be fixed (we always suppose that h is sufficiently small). Let

$$\Omega^h = \bigcup_{i=1}^{m} \tau_i$$

be a triangulation of the domain Ω. We suppose that Ω^h is a quasi-uniform triangulation [12], i.e. there exist positive constants $\underline{c}$, $\overline{c}$, and σ which are independent of h and satisfy the conditions

$$\underline{c}\,h \leq r_i \leq \overline{c}\,h\,, \quad \frac{r_i}{\varrho_i} \leq \sigma\,, \quad i = 1,2,\ldots,m\,,$$

where r_i and ϱ_i are radii of the circumscribed and the inscribed circles for the triangle τ_i, respectively. We also assume that the triangulation boundary Γ^h approximates Γ with an error $\mathcal{O}(h^2)$ and $\Gamma \subset \mathbb{R}^2 \setminus \Omega^h$. For the triangulation Ω^h we define the space $H_h(\Omega^h)$ of real continuous functions which are linear on each triangle of Ω^h and vanish at Γ^h. We extend these functions on $\Omega \setminus \Omega^h$ by zero.

The solution of the projection problem

$$u^h \in H_h(\Omega^h) : \; a(u^h, v^h) = l(v^h), \quad \forall v^h \in H_h(\Omega^h) \tag{5}$$

will be called an approximate solution of problem (4). Each function $u^h \in H_h(\Omega^h)$ is put in standard correspondence with a real column vector $u \in \mathbb{R}^N$ whose components are values of the function u^h at the corresponding nodes of the triangulation Ω^h. Then, problem (5) is equivalent to the system of linear algebraic equations

$$\begin{gathered} Au = f \\ (Au, v) = a(u^h, v^h), \; \forall u^h, v^h \in H_h(\Omega^h), \quad (f, v) = l(v^h), \; \forall v^h \in H_h(\Omega^h), \end{gathered} \tag{6}$$

where u^h and v^h are the respective prolongations of the vectors u and v; (f, v) is the Euclidean scalar product in $\mathbb{R}^N$.

The main goal of this section is to construct a symmetric positive definite preconditioning operator B for problem (6) such that the inequalities

$$c_1(Bu, u) \leq (Au, u) \leq c_2(Bu, u), \quad \forall u \in \mathbb{R}^N, \tag{7}$$

are fulfilled with positive constants c_1 and c_2 which are independent of the parameters h, p, and q. The multiplication of a vector by B^{-1} should be easy to implement. To do it, we completely follow [28]. The preconditioning operator B in (7) is constructed on the basis of the fictitious space lemma [27]. We introduce a fictitious (auxiliary) space and the corresponding operators. To end this, we embed the domain Ω in a square Π.

Let K_i denote the union of triangles in the triangulation Ω^h which have a common vertex z_i and let d_i be the maximal radius of the circle inscribed in K_i. In the square Π we introduce an auxiliary grid Π^h with a step size $\bar{h}$ such that

$$\bar{h} < \frac{1}{\sqrt{2}} \min_i d_i . \tag{8}$$

Let us assume that $\bar{h} = 2^{-l}s$ holds, where s is the length of the sides of Π and l is a positive integer. We denote the nodes of the grid Π^h by $Z_{i,j}$

$$Z_{i,j} = (x_i, y_j), \quad i, j = 0, 1, \ldots, l,$$

and the cells of Π^h by D_{ij}

$$D_{ij} = \{(x, y) \mid x_i \leq x < x_{i+1}, \; y_j \leq y < y_{j+1}\}, \quad \Pi^h = \bigcup_{i,j=0}^{l-1} D_{ij} .$$

Let Q^h denote the minimal figure that consists of cells D_{ij} and contains Ω^h : $\Omega^h \subset Q^h$, $D_{ij} \cap \Omega^h \neq \emptyset$; let S^h be the set of boundary nodes of Q^h. Using cell diagonals, we triangulate Q^h and Π^h; hereafter, the designations Q^h and Π^h refer to triangulations too. Let $H_h(Q^h)$ be the space of real continuous functions which are linear on the triangles of Q^h and vanish at the nodes of S^h. The space $H_h(Q^h)$ will be used as the fictitious space [28]. Furthermore, we introduce

$$a_Q(U, V) = \int_{Q^h} (p\,(\nabla U, \nabla V) + qUV)\, dQ^h, \quad \forall U, V \in \overset{\circ}{H}{}^1(Q^h). \tag{9}$$

We define now the restriction operator R

$$R : \; H_h(Q^h) \to H_h(\Omega^h),$$

the extension operator T
$$T\,:\;H_h(\Omega^h)\rightarrow H_h(Q^h)\,,$$
and an easily invertible operator in the space $H_h(Q^h)$. Let us begin with the operator R. For a given mesh function
$$U^h(Z_{i,j})\in H_h(Q^h)$$
we define a function $u^h\in H_h(\Omega^h)$ as follows. Let z_l be a vertex in the triangulation Ω^h; assume that $z_l\in D_{ij}$. We put
$$u^h(z_l)=(RU^h)(z_l)=U^h(Z_{i,j})\,.$$
The function u^h is equal to zero at nodes $z_l\in\Gamma^h$.

Then, let us define the operator T. For a given function $u^h\in H_h(\Omega^h)$ we define a function $U^h\in H_h(Q^h)$. The function U^h is equal to zero at nodes $Z_{i,j}\in S^h$. At the other nodes U^h is defined as follows: If a cell D_{ij} contains a vertex z_l (according to (8) it can be only one vertex of the triangulation Ω^h) we put
$$U^h(Z_{i,j})=(Tu^h)(Z_{i,j})=u^h(z_l)\,.$$
For each of the remaining nodes $Z_{i,j}\in Q^h$ we find the closest vertex z_l of the triangulation Ω^h (if there are several closest vertices, we can choose any of them) and put
$$U^h(Z_{i,j})=(Tu^h)(Z_{i,j})=u^h(z_l)\,.$$
To define an easily invertible operator in the space $H_h(Q^h)$ which generates an equivalent norm to $a_Q(U^h,U^h)$ we consider in Π^h the sequence of grids
$$\Pi^h_1,\,\Pi^h_2,\ldots,\,\Pi^h_l\equiv\Pi^h$$
with the step sizes
$$h_1=2^{-1}s,\;h_2=2^{-2}s,\;\ldots,\;h_l=2^{-l}s\equiv\bar h\,.$$
We triangulate these grids and consider the corresponding finite element spaces
$$W^h_1\subset W^h_2\subset\cdots\subset W^h_l\equiv H_h(\Pi^h)\,.$$
Using the nodal basis $\{\Phi^{(k)}_i\}_{i=1}^{N_k}$ of the spaces W^h_k, $k=1,2,\ldots,l$, we define operators
$$\begin{gathered}C_kU^h=\sum_{\mathrm{supp}\Phi^{(k)}_i\subset Q^h}(U^h,\Phi^{(k)}_i)_{L_2(Q^h)}\Phi^{(k)}_i\,,\quad B_k=RC_kR^T\,,\quad k=1,2,\ldots,l\,,\\ C^{-1}=\delta_1C_1+\ldots+\delta_lC_l\,,\quad B^{-1}=RC^{-1}R^T=\delta_1B_1+\ldots+\delta_lB_l\end{gathered}\tag{10}$$
with
$$\delta_k=\Big(p+\frac{q}{2^{2k}}\Big)^{-1}\,,$$
and R^T is the transposed matrix of R.

Here and in the following we use the same notation for the operators (C_k, B_k, R) and their matrix representation.

Theorem 2.1 *There exist positive constants c_1 and c_2 independent of h, p, and q, such that*
$$c_1(A^{-1}u,u)\le(B^{-1}u,u)\le c_2(A^{-1}u,u)\,,\quad\forall u\in\mathbb{R}^N\,.$$

Proof: It is obvious that

$$RTu^h = u^h\,, \quad \forall u^h \in H_h(\Omega^h)\,.$$

Using the well-known equivalence of norms and seminorms in the spaces $H_h(\Omega^h)$ and $H_h(Q^h)$ [30], as well as the difference counterparts of these we get

$$\begin{aligned}
c_3\|u^h\|^2_{L_2(\Omega^h)} &\le \sum_{\tau_i\in\Omega^h} h^2((u^h(z_{i_1}))^2 + (u^h(z_{i_2}))^2 + (u^h(z_{i_3}))^2) \\
&\le c_4\|u^h\|^2_{L_2(\Omega^h)}\,, \quad \forall u^h \in H_h(\Omega^h)\,,
\end{aligned}$$

$$\begin{aligned}
c_3\|U^h\|^2_{L_2(Q^h)} &\le \sum_{D_{ij}\in Q^h} h^2((U^h(Z_{i,j}))^2 + (U^h(Z_{i+1,j}))^2 + (U^h(Z_{i,j+1}))^2 + (U^h(Z_{i+1,j+1}))^2) \\
&\le c_4\|U^h\|^2_{L_2(Q^h)}\,, \quad \forall U^h \in H_h(Q^h)\,,
\end{aligned}$$

$$\begin{aligned}
c_3\|\nabla u^h\|^2_{L_2(\Omega^h)} &\le \sum_{\tau_i\in\Omega^h} ((u^h(z_{i_1}) - u^h(z_{i_2}))^2 + (u^h(z_{i_2}) - u^h(z_{i_3}))^2 \\
&\qquad + (u^h(z_{i_3}) - u^h(z_{i_1}))^2) \\
&\le c_4\|\nabla u^h\|^2_{L_2(\Omega^h)}\,, \quad \forall u^h \in H_h(\Omega^h)\,,
\end{aligned}$$

$$\begin{aligned}
c_3\|\nabla U^h\|^2_{L_2(Q^h)} &\le \sum_{D_{ij}\in Q^h} ((U^h(Z_{i+1,j}) - U^h(Z_{i,j}))^2 + (U^h(Z_{i,j+1}) - U^h(Z_{i,j}))^2 \\
&\qquad + (U^h(Z_{i+1,j+1}) - U^h(Z_{i+1,j}))^2 + (U^h(Z_{i+1,j+1}) - U^h(Z_{i,j+1}))^2) \\
&\le c_4\|\nabla U^h\|^2_{L_2(Q^h)}\,, \quad \forall U^h \in H_h(Q^h)\,,
\end{aligned}$$

where z_{i_1}, z_{i_2}, z_{i_3} are the vertices of the triangle $\tau_i \in \Omega^h$. Furthermore, we have the following estimates

$$\begin{aligned}
a(RU^h, RU^h) &\le c_R\, a_Q(U^h, U^h)\,, &\quad \forall U^h \in H_h(Q^h)\,, \\
c_T\, a_Q(Tu^h, Tu^h) &\le a(u^h, u^h)\,, &\quad \forall u^h \in H_h(\Omega^h)\,.
\end{aligned}$$

Here c_3, c_4, c_R, and c_T are independent of h, p, and q; $a_Q(.,.)$ is defined in (9). Using the fictitious space lemma, multilevel techniques [8, 28, 31, 32], and the obvious estimate

$$a_Q(U^h, U^h) \le p\,\|U^h\|^2_{H^1(Q^h)} + q\|U^h\|^2_{L_2(Q^h)} \le c_5\, a_Q(U^h, U^h) \quad \forall U^h \in H_h(Q^h)\,,$$

then we get the assertion of Theorem 2.1. □

Remark 2.1 *The cost of arithmetical work for the action of R or R^T on a vector is proportional to the number of nodes N_l in the mesh domain. The arithmetical cost for the action C^{-1} on a vector depends on the kind of implementation: If we apply each operator C_k, $k = 1, 2, \ldots, l$, to a vector r individually, the arithmetical cost is of the order $\mathcal{O}(N_l \log N_l)$. If we need only the vector $w = C^{-1}r$ and not the vectors $w_k = C_k r$, $k = 1, 2, \ldots, l$, we can implement the action C^{-1} on a vector in such a way that the arithmetical cost is of the order $\mathcal{O}(N_l)$.*

Remark 2.2 *In the case of Neumann boundary conditions in (3) and positive q, we can define corresponding preconditioning operators C_N^{-1} and B_N^{-1} as follows:*

$$C_N^{-1} = \delta_0 C_0 + C^{-1}\,, \quad B_N^{-1} = RC_N^{-1}R^T\,, \quad \delta_0 = q^{-1}\,,$$

$$C_0 U^h = \frac{(U^h, \Phi^{(0)})_{L_2(Q^h)}}{(\Phi^{(0)}, \Phi^{(0)})_{L_2(Q^h)}}\,\Phi^{(0)}\,, \quad \Phi^{(0)}(x) \equiv 1\,,\ x \in Q^h\,,$$

where C^{-1} is defined in (10). Then, Theorem 2.1 holds for this case too.

Remark 2.3 *Optimal preconditioning operators can also be constructed using the technique from [29] for locally refined grids.*

3 Iterative processes with variable preconditioners

In the previous section, the additive preconditioning operator B for iterative solvers was constructed. For example, we can use the following preconditioned gradient method [23] for solving the system of equations (6):

$$u^{(k+1)} = u^{(k)} - \tau^{(k+1)}(\delta_1 B_1 + \ldots + \delta_l B_l)(Au^{(k)} - f)\,, \quad k = 0, 1, \ldots\,, \quad u^{(0)} \in \mathbb{R}^N\,, \quad (11)$$

where $\tau^{(k+1)}$ is defined from the minimization problem:

$$\tau^{(k+1)} = \arg\min_{\tau^{(k+1)} \in \mathbb{R}} \|x^{(k+1)}\|_2$$

with

$$x^{(k)} = A^{1/2} z^{(k)}\,, \quad z^{(k)} = u^{(k)} - u\,, \quad \|x^{(k)}\|_2 = \|z^{(k)}\|_A.$$

Then, we get with $D = A^{1/2}B^{-1}A^{1/2}$

$$\tau^{(k+1)} = \frac{(Dx^{(k)}, x^{(k)})}{(Dx^{(k)}, Dx^{(k)})} = \frac{(B^{-1}Az^{(k)}, Az^{(k)})}{(AB^{-1}Az^{(k)}, B^{-1}Az^{(k)})}\,.$$

It follows from Theorem 2.1 that the convergence rate of the iterative process (11) is independent of the parameters h, p, and q. To get this optimal convergence rate we used the explicit form of the elliptic problem (3), and we defined the factors δ_k, $k = 1, 2, \ldots, l$, analytically. For more general elliptic problems the optimal choice of the factors δ_k in preconditioners of additive form is not obvious. In this situation, it is quite natural to consider, instead of (11), the following gradient-like iterative procedure with a variable preconditioner.

Algorithm 3.1 *Define a sequence $\{u^{(k+1)}\}$ as follows:*

$$u^{(k+1)} = u^{(k)} - (\tau_1^{(k+1)} B_1 + \ldots + \tau_l^{(k+1)} B_l)(Au^{(k)} - f)\,, \quad k = 0, 1, \ldots\,, \quad u^{(0)} \in \mathbb{R}^N\,, \ (12)$$

where $\tau_1^{(k+1)}$, ..., $\tau_l^{(k+1)}$ are defined from the minimization problem

$$\{\tau_1^{(k+1)}, \ldots, \tau_l^{(k+1)}\} = \arg\min_{\tau_i^{(k+1)} \in \mathbb{R},\, i=1,\ldots,l} \|x^{(k+1)}\|_2\,.$$

Then, $\tau_1^{(k+1)}, \ldots, \tau_l^{(k+1)}$ satisfy the following system of equations

$$Q \begin{pmatrix} \tau_1^{(k+1)} \\ \tau_2^{(k+1)} \\ \vdots \\ \tau_l^{(k+1)} \end{pmatrix} = \begin{pmatrix} (D_1 x^{(k)}, x^{(k)}) \\ (D_2 x^{(k)}, x^{(k)}) \\ \vdots \\ (D_l x^{(k)}, x^{(k)}) \end{pmatrix}, \tag{13}$$

where the matrix Q is the Gram-Schmidt matrix

$$\begin{pmatrix} (D_1x^{(k)}, D_1x^{(k)}) & (D_2x^{(k)}, D_1x^{(k)}) & \cdots & (D_lx^{(k)}, D_1x^{(k)}) \\ (D_1x^{(k)}, D_2x^{(k)}) & (D_2x^{(k)}, D_2x^{(k)}) & \cdots & (D_lx^{(k)}, D_2x^{(k)}) \\ \vdots & \vdots & \ddots & \vdots \\ (D_1x^{(k)}, D_lx^{(k)}) & (D_2x^{(k)}, D_lx^{(k)}) & \cdots & (D_lx^{(k)}, D_lx^{(k)}) \end{pmatrix}, \qquad \begin{array}{l} D_i = A^{1/2} B_i A^{1/2}, \\ i = 1, 2, \ldots, l. \end{array}$$

Remark 3.1 *In general, the matrix Q can be singular, but the system (13) is consistent and any solution of (13) gives a solution of the minimization problem.*

Using the well-known motivation [23], we have the following Theorem.

Theorem 3.1 *Let A, B_i, $\tau_i^{(k+1)}$, $i = 1, 2, \ldots, l$, be from (6), (10), (12), and (13), respectively. Then, there exists a constant $\rho < 1$ independent of h, p, and q such that*

$$\|u^{(k)} - u\|_A \le \rho^k \|u^{(0)} - u\|_A, \quad k = 0, 1, 2, \ldots .$$

Here, $u^{(k)}$ is the k-th iterate in the iterative process (12).

Proof: Let $u^{(k)}$ be the k-th iterate from (12). If we put in (11)

$$\tau = \tau^{(k+1)} \equiv \frac{2}{c_1 + c_2},$$

where c_1, c_2 are from Theorem 2.1, and define

$$\tilde{u}^{(k+1)} = u^{(k)} - \tau B^{-1}(Au^{(k)} - f),$$

then we get from the minimization property and from the convergence estimate for the gradient method (11) the following inequalities

$$\|u^{(k+1)} - u\|_A \le \|\tilde{u}^{(k+1)} - u\|_A \le \frac{c_2 - c_1}{c_2 + c_1} \|u^{(k)} - u\|_A$$

which give the assertion of Theorem 3.1. □

Now we consider generalizations of a conjugate gradient method with variable preconditioners. For the sequence $\{x^{(k)}\}$,

$$x^{(k+1)} = x^{(k)} - (\tau_1^{(k+1)} D_1 + \ldots + \tau_l^{(k+1)} D_l) x^{(k)},$$

we have the representation

$$x^{(n)} = (I - (\tau_1^{(n)} D_1 + \ldots + \tau_l^{(n)} D_l)) \cdots (I - (\tau_1^{(1)} D_1 + \ldots + \tau_l^{(1)} D_l)) x^{(0)}. \tag{14}$$

By analogy to the classical conjugate gradient method with a fixed preconditioner we can define nl parameters $\tau_i^{(k)}$, $i = 1, 2, \ldots, l$, $k = 1, 2, \ldots, n$, from the global minimization problem through n iterations:

$$\{\tau_i^{(k)}\}_{i=1,2,\ldots,l,\, k=1,2,\ldots,n} = \mathop{\arg\min}_{\tau_i^{(k)} \in \mathbb{R},\, 1\le k\le n,\, 1\le i\le l} \|x^{(n)}\|_2 \,, \tag{15}$$

where $x^{(n)}$ is defined in (14).

Problem (15) is strongly nonlinear. To simplify it, we define the following formalism. Let α be a multi-index

$$\alpha = (\alpha_1 \alpha_2 \ldots \alpha_j)\,, \quad \alpha_i \in \{1, 2, \ldots, l\}\,.$$

We denote the length of the multi-index α by $|\alpha|$:

$$|\alpha| = j$$

and define

$$D_\alpha = D_{\alpha_1} D_{\alpha_2} \cdots D_{\alpha_j}\,.$$

Obviously, we have

$$D_\alpha D_\beta = D_{\alpha\beta}\,, \quad \alpha\beta = (\alpha_1\alpha_2\ldots\alpha_j\beta_1\beta_2\ldots\beta_m)\,, \quad |\alpha| = j\,,\ |\beta| = m\,,$$

$$D_\alpha^T = D_{\bar\alpha}\,, \quad \bar\alpha = (\alpha_j\alpha_{j-1}\ldots\alpha_1)\,.$$

Instead of (14), let us consider the relation

$$x^{(n)} = \Big(I + \sum_{j=1}^{n} \sum_{|\alpha|=j} a_\alpha^{(n)} D_\alpha\Big) x^{(0)} \tag{16}$$

with arbitrary numbers $a_\alpha^{(n)} \in \mathbb{R}$. We define the $(l^{n+1} - 1)/(l-1) - 1$ parameters $a_\alpha^{(n)}$ from the following minimization problem

$$\{a_\alpha^{(n)}\}_{|\alpha|=1,2,\ldots,n} = \mathop{\arg\min}_{a_\alpha^{(n)} \in \mathbb{R},\, 1\le|\alpha|\le n} \|x^{(n)}\|_2 \,, \tag{17}$$

where $x^{(n)}$ is given by the relation (16). It is obvious that we have in (17) an extended set of parameters compared to problem (15). Consequently, the parameters determined by (15) give a $\|x^{(n)}\|_2$ which is not smaller than in the minimization problem (17). Problem (17) is equivalent to the system of equations

$$\frac{\partial\|x^{(n)}\|_2^2}{\partial a_\alpha^{(n)}} = 2\,\Big(\sum_{i=1}^{n}\sum_{|\beta|=i} a_\beta^{(n)} (D_\beta x^{(0)}, D_\alpha x^{(0)}) + (x^{(0)}, D_\alpha x^{(0)})\Big) = 0 \quad \forall\alpha\,. \tag{18}$$

Lemma 3.1 *Let $x^{(n)}$ be defined by relation (16). Then, the numbers $a_\alpha^{(n)}$, $1 \le |\alpha| \le n$, give a solution of problem (17) if and only if*

$$(D_\alpha x^{(n)}, x^{(k)}) = 0\,, \quad 1 \le |\alpha| \le n - k\,, \quad k = 0, 1, \ldots, n-1\,. \tag{19}$$

Proof: From (18) it is easy to see that condition (17) is equivalent to

$$(x^{(n)}, D_\alpha x^{(0)}) = 0\,, \quad 1 \le |\alpha| \le n\,. \tag{20}$$

Obviously, relations (20) follow from (19). Since

$$
\begin{aligned}
(D_\alpha x^{(n)}, x^{(k)}) &= \Big(x^{(n)}, D_{\bar\alpha}\Big(x^{(0)} + \sum_{j=1}^{k}\sum_{|\beta|=j} a_\beta^{(k)} D_\beta x^{(0)}\Big)\Big) \\
&= \Big(x^{(n)}, D_{\bar\alpha} x^{(0)} + \sum_{j=1}^{k}\sum_{|\beta|=j} a_\beta^{(k)} D_{\bar\alpha} D_\beta x^{(0)}\Big) \\
&= (x^{(n)}, D_{\bar\alpha} x^{(0)}) + \sum_{j=1}^{k}\sum_{|\beta|=j} a_\beta^{(k)} (x^{(n)}, D_{\bar\alpha\beta} x^{(0)})
\end{aligned}
$$

and $|\bar\alpha\beta| = |\alpha| + |\beta|$ we get from (20) the relations (19). □

To perform the iterative scheme (16), we can solve system (18) and compute $x^{(n)}$ or the iterate $u^{(n)}$ by

$$
u^{(n)} = u^{(0)} + \sum_{j=1}^{n}\sum_{|\alpha|=j} a_\alpha^{(n)} B_\alpha (Au^{(0)} - f)\,, \quad B_\alpha = B_{\alpha_1} A B_{\alpha_2} A \cdots A B_{\alpha_j}\,. \tag{21}
$$

Some numerical results for the application of (21) are presented in Subsection 4.3. But in practice the solution of system (18) and the computation of $u^{(n)}$ by the formula (21) is very expensive. We do not know a low cost implementation of (21) which solves the optimization problem (17). This is an open question for us. Below we give CG-like iterative procedures with variable additive preconditioners. These procedures do not satisfy all conditions (19) and, consequently, do not solve the minimization problem (17), but these procedures can be useful in practice.

Algorithm 3.2 *We define sequences $\{x^{(k)}\}$, $\{p_1^{(k)}\}$, ... ,$\{p_l^{(k)}\}$ as follows:*

$$
\begin{aligned}
&x^{(0)} \in \mathbb{R}^N\,, \quad p_i^{(1)} = D_i x^{(0)}\,, \quad i = 1, 2, \ldots l\,, \\
&x^{(1)} = x^{(0)} - (\tau_1^{(1)} p_1^{(1)} + \ldots + \tau_l^{(1)} p_l^{(1)})\,, \\
&\{\tau_i^{(1)}\}_{i=1,2,\ldots,l} = \arg \min_{\tau_i^{(1)} \in \mathbb{R},\, 1 \le i \le l} \|x^{(1)}\|_2\,,
\end{aligned} \tag{22}
$$

$$
\begin{aligned}
p_i^{(k+1)} &= D_i x^{(k)} + \beta_i^{(k+1)} p_i^{(k)}\,, \quad i = 1, 2, \ldots, l\,, \\
x^{(k+1)} &= x^{(k)} - (\tau_1^{(k+1)} p_1^{(k+1)} + \ldots + \tau_l^{(k+1)} p_l^{(k+1)})\,, \quad k = 1, 2, \ldots \\
\{\tau_i^{(k+1)}, \beta_i^{(k+1)}\}_{i=1,2,\ldots,l} &= \arg \min_{\tau_i^{(k+1)} \in \mathbb{R},\, \beta_i^{(k+1)} \in \mathbb{R},\, 1 \le i \le l} \|x^{(k+1)}\|_2\,.
\end{aligned} \tag{23}
$$

The minimization problem (23) is equivalent to the system of linear equations

$$
\begin{pmatrix} Q_x & Q_{xp} \\ Q_{px} & Q_p \end{pmatrix} \begin{pmatrix} a_x \\ a_p \end{pmatrix} = \begin{pmatrix} b_x \\ b_p \end{pmatrix} \tag{24}
$$

of the order $2l$, where

$$
Q_x = [(D_i x^{(k)}, D_j x^{(k)})]_{i,j=1}^{l}\,,\ Q_p = [(p_i^{(k)}, p_j^{(k)})]_{i,j=1}^{l}\,,\ Q_{xp} = [(p_i^{(k)}, D_j x^{(k)})]_{i,j=1}^{l}\,,\ Q_{px} = Q_{xp}^T
$$

$$
a_x = [\tau_i^{(k+1)}]_{i=1}^{l}\,, \quad a_p = [\tau_i^{(k+1)} \beta_i^{(k+1)}]_{i=1}^{l}\,,
$$

$$b_x = [(x^{(k)}, D_i x^{(k)})]_{i=1}^l \,, \quad b_p = [(x^{(k)}, p_i^{(k)})]_{i=1}^l \,.$$

Using the same idea as in the proof of Theorem 3.1, i.e. considering

$$\tau_i^{(k+1)} = 2/(c_1 + c_2)\, \delta_i \,, \quad \beta_i^{(k+1)} = 0 \,, \quad i = 1, 2, \ldots, l, \quad k = 0, 1, \ldots$$

Theorem 3.2 can be proven.

Theorem 3.2 *Let A, B_i, $\tau_i^{(k+1)}$, $\beta_i^{(k+1)}$, $i = 1, 2, \ldots, l$, be from (6), (10), (22), and (23), respectively. Then, there exists a constant $\rho < 1$ independent of h, p, and q such that*

$$\|x^{(k)}\|_2 \le \rho^k \, \|x^{(0)}\|_2 \,, \quad k = 0, 1, \ldots \,.$$

Here $x^{(k)}$ is defined by Algorithm 3.2.

Next, we modify Algorithm 3.2 slightly, i.e. we use fewer parameters, and obtain Algorithm 3.3.

Algorithm 3.3 *We define sequences $\{x^{(k)}\}$ and $\{p^{(k)}\}$ as follows:*

$$\begin{gathered} x^{(0)} \in \mathbb{R}^N \,, \quad p^{(1)} = \tau_1^{(1)} D_1 x^{(0)} + \ldots + \tau_l^{(1)} D_l x^{(0)} \,, \\ x^{(1)} = x^{(0)} - p^{(1)} \,, \\ \{\tau_i^{(1)}\}_{i=1,2,\ldots,l} = \mathop{\arg\min}_{\tau_i^{(1)} \in \mathbb{R},\, 1 \le i \le l} \|x^{(1)}\|_2 \,, \end{gathered} \tag{25}$$

$$\begin{gathered} p^{(k+1)} = (\tau_1^{(k+1)} D_1 x^{(k)} + \ldots + \tau_l^{(k+1)} D_l x^{(k)}) + \beta^{(k+1)} p^{(k)} \,, \quad i = 1, 2, \ldots, l \,, \\ x^{(k+1)} = x^{(k)} - p^{(k+1)} \,, \quad k = 1, 2, \ldots \,, \\ \{\tau_i^{(k+1)}, \beta^{(k+1)}\}_{i=1,2,\ldots,l} = \mathop{\arg\min}_{\tau_i^{(k+1)} \in \mathbb{R},\, \beta^{(k+1)} \in \mathbb{R},\, 1 \le i \le l} \|x^{(k+1)}\|_2 \,. \end{gathered} \tag{26}$$

This minimization problem is equivalent to the system of linear equations

$$\begin{pmatrix} Q_x & Q_{xp} \\ Q_{px} & Q_p \end{pmatrix} \begin{pmatrix} a_x \\ a_p \end{pmatrix} = \begin{pmatrix} b_x \\ b_p \end{pmatrix}$$

of the order $l + 1$, where

$$Q_x = [(D_i x^{(k)}, D_j x^{(k)})]_{i,j=1}^l \,, \; Q_p = (p^{(k)}, p^{(k)}) \,, \; Q_{xp} = [(p^{(k)}, D_i x^{(k)})]_{i=1}^l \,, \; Q_{px} = Q_{xp}^T$$

$$\begin{gathered} a_x = [\tau_i^{(k+1)}]_{i=1}^l \,, \quad a_p = \beta^{(k+1)} \,, \\ b_x = [(x^{(k)}, D_i x^{(k)})]_{i=1}^l \,, \quad b_p = (x^{(k)}, p^{(k)}) \,. \end{gathered}$$

Using the same ideas as in the proof of Theorem 3.1 we get the following Theorem.

Theorem 3.3 *Let A, B_i, $\tau_i^{(k+1)}$, $\beta^{(k+1)}$, $i = 1, 2, \ldots, l$, be from (6), (10), (25), and (26), respectively. Then, there exists a constant $\rho < 1$ independent of h, p, and q such that*

$$\|x^{(k)}\|_2 \le \rho^k \, \|x^{(0)}\|_2 \,, \quad k = 0, 1, \ldots$$

with $x^{(k)}$ from Algorithm 3.3.

By introducing additionally some orthogonality conditions we obtain Algorithm 3.4.

Algorithm 3.4 *We define sequences* $\{x^{(k)}\}$ *and* $\{p_1^{(k)}\}, \ldots, \{p_l^{(k)}\}$ *as follows:*

$$x^{(0)} \in \mathbb{R}^N, \quad \tilde{p}_i^{(1)} = D_i x^{(0)}, \quad i = 1, 2, \ldots, l.$$

Then, we orthogonalize the vectors $\tilde{p}_i^{(1)}$*:*

$$\begin{aligned} p_1^{(1)} &= \tilde{p}_1^{(1)} \\ p_i^{(1)} &= \sum_{j=1}^{i-1} \alpha_j^{(i)} p_j^{(1)} + \tilde{p}_i^{(1)}, \quad \alpha_j^{(i)} = -\frac{(\tilde{p}_i^{(1)}, p_j^{(1)})}{(p_j^{(1)}, p_j^{(1)})}, \quad i = 1, 2, \ldots, l, \end{aligned}$$

and we put

$$x^{(1)} = x^{(0)} - (\tau_1^{(1)} p_1^{(1)} + \ldots + \tau_l^{(1)} p_l^{(1)}), \quad \tau_i^{(1)} = \frac{(x^{(0)}, p_i^{(1)})}{(p_i^{(1)}, p_i^{(1)})}. \tag{27}$$

For $k = 1, 2, \ldots$ *we define*

$$\begin{aligned} \tilde{p}_i^{(k+1)} = D_i x^{(k)} &+ \left(\beta_{i_1}^{\left(k+\frac{k}{k}\right)} p_1^{(k)} + \ldots + \beta_{i_l}^{\left(k+\frac{k}{k}\right)} p_l^{(k)}\right) \\ &+ \left(\beta_{i_1}^{\left(k+\frac{k-1}{k}\right)} p_1^{(k-1)} + \ldots + \beta_{i_l}^{\left(k+\frac{k-1}{k}\right)} p_l^{(k-1)}\right) \\ &+ \ldots + \left(\beta_{i_1}^{\left(k+\frac{1}{k}\right)} p_1^{(1)} + \ldots + \beta_{i_l}^{\left(k+\frac{1}{k}\right)} p_l^{(1)}\right), \end{aligned}$$

with

$$\beta_{i_j}^{\left(k+\frac{s}{k}\right)} = -\frac{(D_i x^{(k)}, p_j^{(s)})}{(p_j^{(s)}, p_j^{(s)})}, \quad s = k, k-1, \ldots, 1, \quad i, j = 1, 2, \ldots, l. \tag{28}$$

Then, using the Gram-Schmidt procedure we define

$$\begin{aligned} p_1^{(k+1)} &= \tilde{p}_1^{(k+1)} \\ p_i^{(k+1)} &= \sum_{j=1}^{i-1} \alpha_j^{(i)} p_j^{(k+1)} + \tilde{p}_i^{(k+1)}, \quad \alpha_j^{(i)} = -\frac{(\tilde{p}_i^{(k+1)}, p_j^{(k+1)})}{(p_j^{(k+1)}, p_j^{(k+1)})}, \quad i = 1, 2, \ldots, l. \end{aligned}$$

We put

$$x^{(k+1)} = x^{(k)} - (\tau_1^{(k+1)} p_1^{(k+1)} + \ldots + \tau_l^{(k+1)} p_l^{(k+1)}), \; \tau_i^{(k+1)} = \frac{(x^{(k)}, p_i^{(k+1)})}{(p_i^{(k+1)}, p_i^{(k+1)})}, \; i = 1, 2, \ldots, l. \tag{29}$$

Within the Algorithm 3.4 we have the following orthogonalities

$$\begin{aligned} (x^{(k+1)}, p_i^{(s)}) = 0, \quad i = 1, 2, \ldots, l, \quad s = 1, 2, \ldots, k+1 \\ (p_i^{(s)}, p_j^{(t)}) = 0, \quad |s-t| + |i-j| \neq 0, \quad s, t = 1, 2, \ldots, k+1, \quad i, j = 1, 2, \ldots, l. \end{aligned} \tag{30}$$

These orthogonalities are of the same type as in the usual conjugate gradient method with a fixed preconditioner.

Theorem 3.4 *Let A, B_i, $\beta_i^{(k+1)}$, $\tau_i^{(k+1)}$, $i = 1, 2, \ldots, l$, be from (6), (10), (28), (27) and (29), respectively. Then, there exists a constant $\rho < 1$ independent of h, p, and q such that*

$$\|x^{(k)}\|_2 \leq \rho^k \|x^{(0)}\|_2 \,, \quad k = 0, 1, 2, \ldots \,.$$

Here, $x^{(k)}$ is the k-th iterate in Algorithm 3.4.

Proof: The orthogonality conditions (30) lead to the following minimization property

$$\begin{aligned}
\|x^{(k+1)}\|_2^2 &= \Big\| x^{(k)} - \sum_{i=1}^{l} \tau_i^{(k+1)} p_i^{(k+1)} \Big\|_2^2 = \min_{\eta_i^{(k+1)},\, 1 \leq i \leq l} \Big\| x^{(k)} + \sum_{i=1}^{l} \eta_i^{(k+1)} p_i^{(k+1)} \Big\|_2^2 \\
&= \min_{\eta_i^{(s)},\, 1 \leq i \leq l,\, 1 \leq s \leq k+1} \Big\{ \Big\| x^{(k)} + \sum_{i=1}^{l} \eta_i^{(k+1)} p_i^{(k+1)} \Big\|_2^2 + \sum_{s=1}^{k} \sum_{i=1}^{l} (\eta_i^{(s)} p_i^{(s)}, \eta_i^{(s)} p_i^{(s)}) \Big\} \qquad (31) \\
&= \min_{\eta_i^{(s)},\, 1 \leq i \leq l,\, 1 \leq s \leq k+1} \Big\| x^{(k)} + \sum_{s=1}^{k+1} \sum_{i=1}^{l} \eta_i^{(s)} p_i^{(s)} \Big\|_2^2 .
\end{aligned}$$

Let $\zeta_i^{(k+1)} \in \mathbb{R}$, $i = 1, 2, \ldots, l$, be arbitrary parameters. Using the representation

$$p_i^{(k+1)} = D_i x^{(k)} + \sum_{j=1}^{i-1} \gamma_j^{(i)} D_j x^{(k)} + \sum_{s=1}^{k} \sum_{i=1}^{l} \xi_i^{(s)} p_i^{(s)}$$

for $p_i^{(k+1)}$, we can define numbers $\tilde{\eta}_i^{(s)}$ such that

$$\sum_{s=1}^{k+1} \sum_{i=1}^{l} \tilde{\eta}_i^{(s)} p_i^{(s)} = \sum_{i=1}^{l} \zeta_i^{(k+1)} D_i x^{(k)} .$$

Therefore, the relations

$$\begin{aligned}
\|x^{(k+1)}\|_2^2 &= \min_{\eta_i^{(k+1)},\, 1 \leq i \leq l,\, 1 \leq s \leq k+1} \Big\| x^{(k)} + \sum_{s=1}^{k+1} \sum_{i=1}^{l} \eta_i^{(s)} p_i^{(s)} \Big\|_2^2 \\
&\leq \Big\| x^{(k)} + \sum_{s=1}^{k+1} \sum_{i=1}^{l} \tilde{\eta}_i^{(s)} p_i^{(s)} \Big\|_2^2 = \min_{\zeta_i^{(k+1)},\, 1 \leq i \leq l} \Big\| x^{(k)} + \sum_{i=1}^{l} \zeta_i^{(k+1)} D_i x^{(k)} \Big\|_2^2
\end{aligned} \qquad (32)$$

hold. Then, using the proof of Theorem 3.1, we get the assertion of this theorem. □

Simplifying Algorithm 3.4, i.e. using fewer parameters, we get the next iterative scheme.

Algorithm 3.5 *We define sequences $\{x^{(k)}\}$ and $\{p_1^{(k)}\}$, …, $\{p_l^{(k)}\}$ as follows:*

$$x^{(0)} \in \mathbb{R}^N \,, \quad \tilde{p}_i^{(1)} = D_i x^{(0)} \,, \quad i = 1, 2, \ldots, l \,.$$

Then, using the Gram-Schmidt orthogonalization procedure, we define

$$\begin{aligned}
p_1^{(1)} &= \tilde{p}_1^{(1)} \\
p_i^{(1)} &= \sum_{j=1}^{i-1} \alpha_j^{(i)} p_j^{(1)} + \tilde{p}_i^{(1)} \,, \quad \alpha_j^{(i)} = -\frac{(\tilde{p}_i^{(1)}, p_j^{(1)})}{(p_j^{(1)}, p_j^{(1)})} \,, \quad i = 1, 2, \ldots, l \,,
\end{aligned}$$

and we put

$$x^{(1)} = x^{(0)} - (\tau_1^{(1)} p_1^{(1)} + \ldots + \tau_l^{(1)} p_l^{(1)}), \quad \tau_i^{(1)} = \frac{(x^{(0)}, p_i^{(1)})}{(p_i^{(1)}, p_i^{(1)})}. \tag{33}$$

For $k = 1, 2, \ldots$ we define

$$\tilde{p}_i^{(k+1)} = D_i x^{(k)} + \beta_{i_1}^{(k+1)} p_1^{(k)} + \ldots + \beta_{i_l}^{(k+1)} p_l^{(k)}, \quad \beta_{i_j}^{(k+1)} = -\frac{(D_i x^{(k)}, p_j^{(k)})}{(p_j^{(k)}, p_j^{(k)})}, \tag{34}$$

$$j = 1, 2, \ldots, l, \quad i = 1, 2, \ldots, l.$$

Using again the Gram-Schmidt procedure we define

$$\begin{aligned} p_1^{(k+1)} &= \tilde{p}_1^{(k+1)} \\ p_i^{(k+1)} &= \sum_{j=1}^{i-1} \alpha_j^{(i)} p_j^{(k+1)} + \tilde{p}_i^{(k+1)}, \quad \alpha_j^{(i)} = -\frac{(\tilde{p}_i^{(k+1)}, p_j^{(k+1)})}{(p_j^{(k+1)}, p_j^{(k+1)})}, \quad i = 1, 2, \ldots, l, \end{aligned}$$

and we put

$$x^{(k+1)} = x^{(k)} - (\tau_1^{(k+1)} p_1^{(k+1)} + \ldots + \tau_l^{(k+1)} p_l^{(k+1)}), \quad \tau_i^{(k+1)} = \frac{(x^{(k)}, p_i^{(k+1)})}{(p_i^{(k+1)}, p_i^{(k+1)})}. \tag{35}$$

A simple calculation shows that the following orthogonality conditions

$$(x^{(k+1)}, p_i^{(k+1)}) = 0, \quad (p_i^{(k+1)}, p_j^{(k)}) = 0, \quad i, j = 1, 2, \ldots, l, \tag{36}$$

and the relations

$$(p_i^{(k+1)}, p_j^{(k+1)}) = \delta_{ij} (p_i^{(k+1)}, p_i^{(k+1)}), \quad i, j = 1, 2, \ldots, l \tag{37}$$

hold.

Theorem 3.5 *Let A, B_i, $\beta_i^{(k+1)}$, $\tau_i^{(k+1)}$, $i = 1, 2, \ldots, l$, be from (6), (10), (34), (33), and (35), respectively. Then, there exists a constant $\rho < 1$ independent of h, p, and q such that*

$$\|x^{(k)}\|_2 \leq \rho^k \|x^{(0)}\|_2, \quad k = 0, 1, 2, \ldots$$

holds, where $x^{(k)}$ is the k-th iterate of Algorithm 3.5.

Proof: From the orthogonality conditions (36) and the relations (37) we have the following minimization property

$$\begin{aligned} \|x^{(k+1)}\|_2^2 &= \left\| x^{(k)} - \sum_{i=1}^{l} \tau_i^{(k+1)} p_i^{(k+1)} \right\|_2^2 = \min_{\eta_i^{(k+1)}, 1 \leq i \leq l} \left\| x^{(k)} + \sum_{i=1}^{l} \eta_i^{(k+1)} p_i^{(k+1)} \right\|_2^2 \\ &= \min_{\eta_i^{(s)}, 1 \leq i \leq l, k \leq s \leq k+1} \left\{ \left\| x^{(k)} + \sum_{i=1}^{l} \eta_i^{(k+1)} p_i^{(k+1)} \right\|_2^2 + \sum_{i=1}^{l} (\eta_i^{(k)} p_i^{(k)}, \eta_i^{(k)} p_i^{(k)}) \right\} \\ &= \min_{\eta_i^{(s)}, 1 \leq i \leq l, k \leq s \leq k+1} \left\| x^{(k)} + \sum_{s=k}^{k+1} \sum_{i=1}^{l} \eta_i^{(s)} p_i^{(s)} \right\|_2^2. \end{aligned} \tag{38}$$

Let $\zeta_i^{(k+1)} \in \mathbb{R}$, $i = 1, 2, \ldots, l$, be arbitrary parameters. Using the following representation for $p_i^{(k+1)}$:

$$p_i^{(k+1)} = D_i x^{(k)} + \sum_{j=1}^{i-1} \gamma_j^{(i)} D_j x^{(k)} + \sum_{i=1}^{l} \xi_i^{(k)} p_i^{(k)} ,$$

we can define numbers $\tilde{\eta}_i^{(s)}$ such that

$$\sum_{s=k}^{k+1} \sum_{i=1}^{l} \tilde{\eta}_i^{(s)} p_i^{(s)} = \sum_{i=1}^{l} \zeta_i^{(k+1)} D_i x^{(k)} .$$

The rest of the proof is analogous to the proof of Theorem 3.4. □

Remark 3.2 *The suggested methods can be used for an arbitrary symmetric positive definite matrix A when B^{-1} has an additive form. In the next section we examine numerically some two-level preconditioners for elliptic problems and domain decomposition methods.*

4 Numerical experiments

In this section, we demonstrate the convergence properties of the iterative methods proposed in the previous section on numerical examples. We concentrate our experiments on the Algorithms 3.1 – 3.5. We assume that the triangulations Ω^h and the auxiliary grids Π^h coincide. In this case the operator R in (10) is the identity operator.

We present numerical results for the iterative schemes applied to problem (3). Furthermore, we use our schemes for additive preconditioners which arise from domain decomposition methods and finite element discretizations with two-level p-hierarchical basis functions.

Let us first discuss some implementation details of the new methods with variable preconditioners.

4.1 Some implementation aspects

In Section 3, the Algorithms 3.2 – 3.5 are formulated in terms of sequences $\{x^{(k)}\}$ and $\{p_1^{(k)}\}$, $\ldots$,$\{p_l^{(k)}\}$. This formulation is convenient for the theoretical investigations but not for an implementation. For that reason, we write down these algorithms in terms of sequences $\{u^{(k)}\}$, $\{r^{(k)}\}$, and $\{s_1^{(k)}\}$, $\ldots$,$\{s_l^{(k)}\}$. Then, Algorithm 3.2 can be implemented in the following way:

We define sequences $\{u^{(k)}\}$, $\{r^{(k)}\}$, and $\{s_1^{(k)}\}$, $\ldots$,$\{s_l^{(k)}\}$ as follows:

$$u^{(0)} \in \mathbb{R}^N , \quad r^{(0)} = Au^{(0)} - f , \quad s_i^{(1)} = B_i r^{(0)} , \quad i = 1, 2, \ldots l ,$$

$$\begin{aligned} u^{(1)} &= u^{(0)} - (\tau_1^{(1)} s_1^{(1)} + \ldots + \tau_l^{(1)} s_l^{(1)}) , \\ r^{(1)} &= r^{(0)} - (\tau_1^{(1)} A s_1^{(1)} + \ldots + \tau_l^{(1)} A s_l^{(1)}) , \end{aligned}$$

$$\{\tau_i^{(1)}\}_{i=1,2,\ldots,l} = \underset{\tau_i^{(1)} \in \mathbb{R},\, 1 \le i \le l}{\arg\min} \|z^{(1)}\|_A ,$$

$$
\begin{aligned}
s_i^{(k+1)} &= B_i r^{(k)} + \beta_i^{(k+1)} s_i^{(k)}, \quad i = 1, 2, \ldots, l, \\
u^{(k+1)} &= u^{(k)} - (\tau_1^{(k+1)} s_1^{(k+1)} + \ldots + \tau_l^{(k+1)} s_l^{(k+1)}), \\
r^{(k+1)} &= r^{(k)} - (\tau_1^{(k+1)} A s_1^{(k+1)} + \ldots + \tau_l^{(k+1)} A s_l^{(k+1)}), \quad k = 1, 2, \ldots
\end{aligned}
$$

$$
\{\tau_i^{(k+1)}, \beta_i^{(k+1)}\}_{i=1,2,\ldots,l} = \underset{\tau_i^{(k+1)} \in \mathbb{R}, \beta_i^{(k+1)} \in \mathbb{R}, 1 \le i \le l}{\arg\min} \|z^{(k+1)}\|_A .
$$

From Section 3 it is known that the last minimization problem is equivalent to the system of linear equations (24). In terms of $r^{(k)}$ and $s^{(k)}$ the entries of the submatrices Q_x, Q_p, and Q_{xp} have the form:

$$
Q_x = [(AB_i r^{(k)}, B_j r^{(k)})]_{i,j=1}^l , \; Q_p = [(As_i^{(k)}, s_j^{(k)})]_{i,j=1}^l ,
$$

$$
Q_{xp} = [(As_i^{(k)}, B_j r^{(k)})]_{i,j=1}^l , \; Q_{px} = Q_{xp}^T ,
$$

and for the right-hand side $(b_x \;\; b_p)^T$ we get

$$
b_x = [(r^{(k)}, B_i r^{(k)})]_{i=1}^l , \quad b_p = [(r^{(k)}, s_i^{(k)})]_{i=1}^l .
$$

It is obvious that the other algorithms can be written in an analogous manner.

In Section 3, convergence estimates for the CG-like Algorithms 3.2 – 3.5 were given. These estimates are not the typical estimates for CG methods, they are more similar to the estimates which we get for the usual gradient method with a fixed preconditioner. However, our numerical experiments will show that in many cases the new CG-like algorithms have better convergence properties than the CG method with a fixed preconditioner. Taking into consideration the different cost of arithmetical work per iteration step of the methods presented and of the usual CG method, it arises the question whether the new methods or the usual CG method give the best algorithm with respect to the CPU-time needed. The answer depends on the problem (see the presented numerical results). If we use a fixed preconditioner with the right scaling of the summands in the additive form, the usual CG method will be in general faster than the new methods, since one iteration step of the new methods is more expensive than in the usual CG method.

Let us analyse the arithmetical cost for each iteration step more precisely. Within each step we have to compute scalar products, matrix by vector multiplications, vector operations of the type $y := \alpha x + y$ (called DAXPY operations), and the application of the preconditioner B. In Table 1 we summarize the amount of these operations. Note that in Algorithm 3.4 the amount of arithmetical operations per iteration step is growing with the number k of the iteration step. This is caused by the increasing cost of the orthogonalization process for the vectors $p_i^{(k)}$ or $s_i^{(k)}$, respectively.

Additionally, within Algorithm 3.2 and within Algorithm 3.3, the computation of the iteration parameters in each iteration step requires to solve a system of linear equations of the order $2l$ and of the order $l + 1$, respectively.

Table 1 shows that for large l the cost of arithmetical work of the new methods is much more higher than that of the usual CG method with fixed preconditioners. But we can also see that we have to perform in all algorithms the application of the preconditioner B only once. In some cases, as for example preconditioners on the basis of domain decomposition ideas (see Subsection 4.4), the application of the preconditioning operator B is relatively expensive. In such cases, the better convergence properties of the proposed methods in comparison to the usual CG method implies fewer applications of the operator B. This can lead to faster algorithms.

method	number of			
	scalar products	matrix by vector	DAXPY	actions of B
usual CG method	2	1	3	1
Algorithm 3.2	$l(2l+3)$	l	l^2+3l	1
Algorithm 3.3	$(l+1)(l+4)/2$	l	$l+3$	1
Algorithm 3.4	$l((2k+1)l-3)/2$	l	$kl^2+3l+l(l-1)/2$	1
Algorithm 3.5	$3l(l+1)/2$	l	$l^2+3l+l(l-1)/2$	1

Table 1: Arithmetical cost per iteration step

4.2 BPX preconditioner with variable parameters

As the first example we consider problem (3) with $p = 1$ and $q = s^2$, s being an arbitrary real number. We compare the algorithms with variable preconditioners and the CG method with the fixed preconditioner (10). Problem (3) is solved in the unit square $(0,1)\times(0,1)$. Starting from a coarse mesh with 32 congruent right isosceles triangles, a sequence of nested finite element triangulations is generated. Corresponding to each triangulation we define the finite element subspaces spanned by the usual piecewise linear functions. Using this discretization we get the systems of algebraic finite element equations (6).

Since we want to measure the norm $\|z^{(k)}\|_A$ in each iteration step we need the exact solution u of the system of algebraic equations (6). We choose in the BVP (3) the right-hand side $f \equiv 0$, and therefore, the exact solution of the system (6) is the zero-vector.

The initial guess for the iterative processes is the vector whose components correspond to the values of the function $x^3(1-x)y(1-y)^5$ in the nodes of the finite element meshes. The iterative methods are terminated when the relative error $\|z^{(k)}\|_A/\|z^{(0)}\|_A \le 10^{-4}$ is achieved. In Table 2 the number of iterations for the CG method preconditioned by the BPX method with the fixed factors $\delta_k = (p + 2^{-2k}q)^{-1}$, $k = 1, 2, \ldots, l$, (see Section 2) is presented. In all tables, N denotes the number of unknowns.

s		0	10	20	30	40	50	60	70	80	90	100
l	N	number of iterations										
2	81	11	6	6	8	9	10	11	12	13	13	14
3	289	13	9	7	7	8	9	10	11	11	12	12
4	1089	14	12	8	7	6	7	7	8	8	9	10
5	4225	15	15	11	9	8	7	7	6	6	6	7
6	16641	16	16	13	11	10	9	8	8	7	7	7

Table 2: CG (preconditioned by BPX with fixed factors)

In the Tables 3 – 7 we give the number of iterations for Algorithm 3.1, i.e. the gradient method with the variable preconditioner, and the Algorithms 3.2 – 3.5.

Tables 4 – 7 show that the CG-like algorithms with variable preconditioners have better convergence properties than the CG method with the fixed preconditioner. However, in all these experiments the CPU time needed for the new methods was higher than that for the old one.

s		0	10	20	30	40	50	60	70	80	90	100
l	N	number of iterations										
2	81	32	9	5	5	6	8	9	11	12	13	14
3	289	37	13	8	5	4	4	5	5	6	6	6
4	1089	40	18	12	9	7	5	3	3	3	4	4
5	4225	43	22	16	12	10	8	7	6	5	4	3
6	16641	44	25	19	15	13	11	10	8	7	7	6

Table 3: Gradient method with l parameters (Algorithm 3.1)

s		0	10	20	30	40	50	60	70	80	90	100
l	N	number of iterations										
2	81	12	6	4	5	5	6	7	7	8	8	9
3	289	13	8	6	4	4	4	4	4	4	5	5
4	1089	14	10	7	6	5	4	3	3	3	3	3
5	4225	14	11	9	7	6	6	5	4	4	4	3
6	16641	15	12	10	8	7	6	6	6	5	5	4

Table 4: Application of Algorithm 3.2

s		0	10	20	30	40	50	60	70	80	90	100
l	N	number of iterations										
2	81	12	6	4	5	5	6	7	7	8	8	9
3	289	14	8	6	4	4	4	4	4	5	5	5
4	1089	16	10	8	6	5	4	3	3	3	3	3
5	4225	16	11	9	7	6	6	5	5	4	4	3
6	16641	16	12	10	8	8	7	6	6	5	5	4

Table 5: Application of Algorithm 3.3

s		0	10	20	30	40	50	60	70	80	90	100
l	N	number of iterations										
2	81	10	6	4	5	5	6	7	7	8	8	9
3	289	11	8	6	4	4	4	4	4	4	5	5
4	1089	12	9	7	6	5	4	3	3	3	3	3
5	4225	13	10	8	7	6	5	5	4	4	3	3
6	16641	13	11	9	8	7	6	6	5	5	5	4

Table 6: Application of Algorithm 3.4

s		0	10	20	30	40	50	60	70	80	90	100
l	N	number of iterations										
2	81	11	6	4	5	5	6	7	7	8	8	9
3	289	13	8	6	4	4	4	4	4	4	5	5
4	1089	14	10	8	6	5	4	3	3	3	3	3
5	4225	15	11	9	7	6	6	5	4	4	3	3
6	16641	16	12	10	8	7	7	6	6	5	5	4

Table 7: Application of Algorithm 3.5

Finally, in Table 8, we present the number of iterations if we use the preconditioner B in (10) with the factors $\delta_k = 1$, $k = 1, 2, \ldots, l$. The bad convergence properties in this case show that it is important to know some a-priori information of the corresponding boundary value problem or to compute suitable factors within the iterative process automatically as it is made in the Algorithms 3.1 – 3.5.

s		0	10	20	30	40	50	60	70	80	90	100
l	N	number of iterations										
2	81	11	9	8	11	13	14	15	16	16	17	17
3	289	13	12	13	14	15	17	18	19	20	20	21
4	1089	14	13	15	17	19	20	20	21	22	24	24
5	4225	15	13	15	18	20	22	24	25	26	28	29
6	16641	16	14	16	19	20	22	25	27	28	30	32

Table 8: Application of the BPX preconditioner with factors equal to 1

4.3 Two-level p-hierarchical preconditioners

In this section, we apply the presented algorithms to systems of finite element equations arising from the discretization with two-level p-hierarchical ansatz functions.

We consider two problems, namely problem (3) (with $p = 1$, $q = 0$, and $f = 0$) in the domain $\Omega = (0,1) \times (0,1)$, and a plane linear elasticity problem (state of the plane stress) in the same domain but with the boundary conditions

$$u = (u_1, u_2)^T = 0 \text{ on } \Gamma_1 = \{(x,y) \, : 0 \le x \le 1 \, , \; y = 0\} \, ,$$
$$\sigma_{ij} n_j = 0 \text{ on } \partial\Omega \setminus \Gamma_1 \;\; (i,j = 1,2),$$

where $u = (u_1, u_2)^T$ denotes the displacement vector, σ_{ij} are the components of the stress tensor, and $n = (n_1, n_2)^T$ is the vector of the outer normal on the boundary $\partial\Omega$. Furthermore, we use a Poisson's ratio $\nu = 0.3$ in the elasticity problem.

As in Subsection 4.2, the discretization process is started with a coarse triangulation of the domain Ω which consists of 32 congruent right isosceles triangles. The finer triangulations are generated by a successive refinement process where we generate in each triangle four congruent subtriangles by connecting the midpoints of the edges. In the finest triangulation we define on each edge of the triangles the midpoint such that we get triangles with 6 nodes. The finite element subspaces on the coarse meshes are spanned by the usual piecewise linear finite element functions, and on the finest mesh the finite element subspace is spanned by so-called two-level p-hierarchical functions. These functions are the usual piecewise linear functions and piecewise quadratic functions which are equal to one in exactly one midpoint and are equal to zero in all other nodes. Numbering first the vertex nodes and then the midpoint nodes in the finest triangulation, the system of finite element equations has the following block structure

$$\begin{pmatrix} A_{vv} & A_{vm} \\ A_{mv} & A_{mm} \end{pmatrix} \begin{pmatrix} u_v \\ u_m \end{pmatrix} = \begin{pmatrix} f_v \\ f_m \end{pmatrix}, \tag{39}$$

where "v" and "m" correspond to the vertex nodes and the midpoint nodes, respectively. It is well-known [2, 3, 20] that the matrix

$$\tilde{B} = \begin{pmatrix} A_{vv} & 0 \\ 0 & A_{mm} \end{pmatrix}$$

is spectrally equivalent to the stiffness matrix in (39). From the matrix $\tilde{B}$ we derive the preconditioner

$$B = \begin{pmatrix} B_{vv} & 0 \\ 0 & B_{mm} \end{pmatrix},$$

where B_{vv} stands for the usual BPX preconditioner [10, 31], and for B_{mm} we choose the diagonal part of the matrix A_{mm}, i.e. $B_{mm} = \mathrm{diag}(A_{mm})$. In this way we get a preconditioner B which is spectrally equivalent to the stiffness matrix in (39) (see, e.g., [2, 3, 10, 19, 31]). It is obvious that the preconditioner B is of additive form, and therefore, we can apply the algorithms described in Section 3.

In Table 9 we present the number of iterations of the Algorithms 3.1 – 3.5 and of the usual CG method.

Furthermore, we made some experiments with a preconditioner $B_{mm} = \alpha\,\mathrm{diag}(A_{mm})$ with different values of α (The corresponding algorithms are denoted by CG-α.). The results show that a wrong scaling factor α influences the convergence rate of the usual CG-method essentially, but the convergence behaviour of the new algorithms is not disturbed by such a wrong factor. Using Algorithm 3.1, i.e. the gradient method with variable preconditioner, we can compute the optimal scaling factor α. This optimal parameter α_{opt} is also applied in the usual CG method.

In the case of problem (3), the initial guess for the iterative processes is the vector whose components correspond to the values of the function $x^3(1-x)y(1-y)^5$ in the nodes of the finite element meshes. For the elasticity problem we use that vector whose components correspond to the values of the function $x^3(1-x)y(1-y)^5$ for the u_1 displacement and to the values of the function $x(1-x)^5y^3(1-y)$ for the u_2 displacement. The iterative methods are terminated when the relative error $\|z^{(k)}\|_A/\|z^{(0)}\|_A \leq 10^{-4}$ is achieved.

From Table 9 we can see that the scaling factors 1.0, which are used in the usual CG method, are not so far from the right scaling. Consequently, the algorithms presented in Section 3 can not lead to an essential reduction of the number of iterations.

In this section, we also show an experiment with the iterative process (21). As mentioned in Section 3, this iterative process is very expensive. For that reason we solved only a very small problem, i.e. a problem with 81 and 162 unknowns in the cases of the Poisson's equation and the elasticity problem, respectively. In Table 10 we present for each iteration step k the relative error $\|z^{(k)}\|_A/|z^{(0)}\|_A$. As preconditioner we use the matrix

$$\begin{pmatrix} A_{vv} & 0 \\ 0 & \mathrm{diag}(A_{mm}) \end{pmatrix}.$$

	Poisson's equation				elasticity problem		
N	289	1089	4225	16641	578	2178	8450
solver	number of iterations				number of iterations		
usual CG	15	15	15	14	25	25	24
Algorithm 3.1	34	34	32	29	83	85	81
Algorithm 3.2	14	14	13	12	25	24	21
Algorithm 3.3	15	15	14	14	24	25	23
Algorithm 3.4	13	13	12	11	21	20	19
Algorithm 3.5	14	14	13	12	23	22	20
CG–α_{opt}	14	14	13	12	23	23	23
α_{opt}	0.65	0.55	0.47	0.39	0.72	0.62	0.69
CG–10	26	25	23	20	43	40	36
CG–100	48	53	51	47	100	103	98

Table 9: Number of iterations of the different methods

Table 10 shows that the iterative process (21) has better convergence properties than the other methods. But in the case of the elasticity problem we can see that this iterative process can be unstable or even not converge.

	Poisson's problem	elasticity problem
iteration step	relative error $\|z^{(k)}\|_A/\|z^{(0)}\|_A$	relative error $\|z^{(k)}\|_A/\|z^{(0)}\|_A$
1	0.4903	0.4571
2	0.1134	0.3288
3	0.4674e-01	0.1886
4	0.1848e-01	0.8071e-01
5	0.3446e-02	0.2762e-01
6	0.6385e-03	0.1089e-01
7	0.1345e-04	0.5054e-02
8		0.9596e-03
9		0.3404e-03
10		0.3404e-03

Table 10: Application of the iterative process (21)

4.4 Preconditioners in domain decomposition methods

In this section, we apply our algorithms to the Poisson's equation, i.e. to problem (3) with $p = 1$ and $q = 0$. Here, we want to study the convergence behaviour of the algorithms presented in Section 3, when we use a preconditioner based on a non-overlapping domain decomposition (DD) strategy.

The starting point for the DD method is a decomposition of the domain Ω into non-overlapping subdomains Ω_i, $i = 1, 2, \ldots, p$. A well-known way for defining DD preconditioners is the following: First, we consider the system of algebraic finite element equations in the exact discrete harmonical basis [15, 16]. In this basis, the stiffness matrix has, by a suitable numbering of the nodes, a block structure, where one block is the Schur complement matrix and the other blocks are that parts of the stiffness matrix in the nodal basis which correspond to the inner nodes of each subdomain. Then, the preconditioning matrix

$$B = \begin{pmatrix} \delta_C B_C & 0 \\ 0 & \delta_I B_I \end{pmatrix}, \quad B_I = \mathrm{blockdiag}\{B_{I,i}\}_{i=1,2,\ldots,p}, \tag{40}$$

is spectrally equivalent to the stiffness matrix in the exact discrete harmonical basis if B_C and B_I are spectrally equivalent to the Schur complement and to the stiffness matrix corresponding to the problems in the subdomains, respectively. Examples for such preconditioners are given in [15].

In general, transformations into the exact discrete harmonical basis are too expensive. Therefore, one utilizes an approximate discrete harmonical basis. In [15, 16] it is shown that also in this case the matrix (40) is a spectrally equivalent preconditioner supposed that the matrices B_C and B_I are chosen in an appropriate way. Because of the additive form of the preconditioning matrix (40) we can use this preconditioner within our iterative solvers.

Usually, one does not perform the iterative methods for the system of finite element equations in the approximate discrete harmonical basis. One applies the algorithms to the systems of finite element equations in the nodal basis and utilizes the transformation to the approximate discrete harmonical basis within the preconditioner. This results in the DD-preconditioner $V^T B V$ with a matrix V which describes the change from the nodal basis to the approximate discrete harmonical basis.

In the following, we consider problem (3) ($p = 1$, $q = 0$, $f = 0$) in the square $(0, 4) \times (0, 4)$. The domain Ω is decomposed into 16 subdomains. We generate in each subdomain a sequence of nested triangulations such that we get an admissible triangulation for the whole domain Ω. Corresponding to this sequence of triangulations, a sequence of systems of finite element equations is defined by using the usual piecewise linear functions.

As initial guess for the iterative solvers, a vector is used whose components correspond to the values of the function $x^3(4-x)y(4-y)^5$ in the nodes of the finite element meshes. The iterative methods are terminated when the relative error $\|z^{(k)}\|_A / \|z^{(0)}\|_A \le 10^{-4}$ is achieved.

As matrix B_C we apply a BPS-preconditioner [9] using ideas from [1, 13, 25] on the coupling boundary, i.e. on the boundary of the subdomains, and a global cross-point system. The preconditioner B_I is defined implicitly, i.e. for solving the problems corresponding to the subdomains a multigrid V-cycle with one pre- and one post-smoothing step of Gauss-Seidel type is employed. The basis transformation V makes use of a hierarchical extension technique described in [16].

The numerical experiments presented in Table 11 are performed on 16 processors of a multiprocessor system GC/PP-128.

Table 11 shows that in the most cases the Algorithms 3.2 – 3.5 are faster than the usual PCG method with the preconditioner (40) and $\delta_C = \delta_I = 1$. Within each iteration step of the algorithms implemented on a parallel computer we have to perform communication between the processors. On the multiprocessor system GC/PP-128 the communication power is relatively slow in comparison to the processing power. Therefore, the higher

N	497	2001	8081	32529	130577	523281
solver	CPU-time [sec] (number of iterations)					
CG	0.37 (14)	0.57 (20)	0.95 (26)	2.66 (34)	10.71 (42)	48.42 (50)
Algorithm 3.1	0.76 (38)	1.25 (55)	2.84 (89)	11.39 (142)	64.81 (225)	400.38 (359)
Algorithm 3.2	0.29 (14)	0.40 (17)	0.74 (21)	2.39 (27)	11.38 (35)	59.42 (47)
Algorithm 3.3	0.26 (13)	0.36 (15)	0.68 (20)	1.91 (23)	8.44 (28)	39.75 (34)
Algorithm 3.4	0.34 (13)	0.49 (16)	0.93 (20)	3.30 (23)		
Algorithm 3.5	0.33 (13)	0.42 (15)	0.79 (20)	2.23 (24)	9.54 (29)	45.36 (36)
δ_c/δ_I	0.86	0.71	0.47	0.37	0.28	0.26

Table 11: Iterative algorithms with DD preconditioner

arithmetical cost in each iteration step in the Algorithms 3.2 – 3.5 compared with the usual PCG method is less important.

The ratio δ_c/δ_I in Table 11 is computed by using Algorithm 3.1. This ratio indicates that for the problems with a small number of unknowns the scaling of B_C and B_I with $\delta_C = \delta_I = 1$ (which is usually used) is relatively good.

5 Conclusions

We have presented iterative schemes which are especially designed for the application of additive preconditioners. The algorithms allow to compute the scaling factors for each summand in the preconditioner automatically. Numerical examples show that the new algorithms can be faster than the usual CG method, in particular, when the scaling factors are not known a-priori.

References

[1] V. B. Andreev. Stability of the difference schemes for elliptic equations with regard Dirichlet's condition. *U.S.S.R. Comput. Math. and Math. Phys.*, 12:598–611, 1972.

[2] O. Axelsson. On multigrid methods of the two-level type. In W. Hackbusch and U. Trottenberg, editors, *Multigrid Methods, Proceedings of the Conference held at Köln–Porz, November 23–27, 1981*, volume 960 of *Lecture Notes in Mathematics*, pages 352–367, Berlin–Heidelberg–New York, 1982. Springer–Verlag.

[3] O. Axelsson and I. Gustafsson. Preconditioning and two–level multigrid methods of arbitrary degree of approximation. *Math. Comput.*, 40:219–242, 1983.

[4] O. Axelsson and M. Nikolova. A generalized conjugate gradient minimum residual method (GCG-MR) with variable preconditioners and a relation between residuals of the GCR-MR and GCR-OR methods. Technical report, University of Nijmegen, 1996. (private communication).

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

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

[7] R. E. Bank and J. Xu. The hierarchical basis multigrid method and incomplete LU decomposition. In D. E. Keyes and J. Xu, editors, *Domain decomposition for PDEs*, volume 180 of *Contemporary Mathematics*, pages 163–174, 1994.

[8] F. A. Bornemann and H. Yserentant. A basic norm equivalence for the theory of multilevel methods. *Numer. Math.*, 64:455–476, 1993.

[9] J. H. Bramble, J. E. Pasciak, and A. H. Schatz. The construction of preconditioners for elliptic problems by substructuring I – IV. *Math. Comput.*, 1986, 1987, 1988, 1989. 47, 103–134, 49, 1–16, 51, 415–430, 53, 1–24.

[10] J. H. Bramble, J. E. Pasciak, and J. Xu. Parallel multilevel preconditioners. *Math. Comput.*, 55(191):1–22, 1990.

[11] T. F. Chan and B. F. Smith. Domain decomposition and multigrid algorithms for elliptic problems on unstructured meshes. In D. E. Keyes and J. Xu, editors, *Domain decomposition for PDEs*, volume 180 of *Contemporary Mathematics*, pages 175–190, 1994.

[12] P. Ciarlet. *The finite element method for elliptic problems.* North–Holland, Amsterdam, 1978.

[13] M. Dryja. A capacitance matrix method for Dirichlet problems on polygonal regions. *Numer. Math.*, 39(1):51–64, 1982.

[14] M. Griebel. *Multilevelmethoden als Iterationsverfahren über Erzeugendensystemen.* Teubner Skripten zur Numerik. B. G. Teubner Stuttgart, 1994.

[15] G. Haase, U. Langer, and A. Meyer. The approximate Dirichlet domain decomposition method. Part I: An algebraic approach. Part II: Applications to 2nd-order elliptic boundary value problems. *Computing*, 47:137–151 (Part I), 153–167 (Part II), 1991.

[16] G. Haase, U. Langer, A. Meyer, and S. V. Nepomnyaschikh. Hierarchical extension and local multigrid methods in domain decomposition preconditioners. *East–West J. Numer. Math.*, 2(3):173–193, 1994.

[17] W. Hackbusch. A parallel conjugate gradient method. *Journal of Numerical Linear Algebra with Applications*, 1(2):133–147, 1992.

[18] M. Jung, U. Langer, A. Meyer, W. Queck, and M. Schneider. Multigrid preconditioners and their applications. In G. Telschow, editor, *Third Multigrid Seminar, Biesenthal 1988*, pages 11–52, Berlin, 1989. Karl–Weierstraß–Institut. Report R–MATH–03/89.

[19] M. Jung, U. Langer, and U. Semmler. Two–level hierarchically preconditioned conjugate gradient methods for solving linear elasticity finite element equations. *BIT*, 29:748–768, 1989.

[20] V. G. Korneev. Iterative methods of solving systems of equations of the finite element method. *U.S.S.R. Math. Math. Phys.*, 17(5):109–129, 1977.

[21] R. Kornhuber and H. Yserentant. Multilevel methods for elliptic problems on domains not resolved by the coarse grid. In D. E. Keyes and J. Xu, editors, *Domain decomposition for PDEs*, volume 180 of *Contemporary Mathematics*, pages 49–60, 1994.

[22] Yu. A. Kuznetsov. Multigrid domain decomposition methods for elliptic problems. In *Comput. Meth. for Applied Science and Engineering, VII*, pages 605–616. 1987.

[23] G. I. Marchuk and Yu. A. Kuznetsov. *Iterative Methods and Quadratic Functionals.* Nauka, Moscow, 1972. See also in: Sur les Méth. Numer. en Sci. Phys. et Com., Paris, Dunod, 1974, pp. 3–132.

[24] A. M. Matsokin. Solution of grid equations on non-regular grids. Preprint, Computing Centre, Siberian Branch of Acad. Sci. of USSR, Novosibirsk, 1987. (in Russian).

[25] A. M. Matsokin and S. V. Nepomnyaschikh. On the convergence of the non-overlapping Schwarz subdomain alternating method. *Sov. J. Numer. Anal. Math. Modelling*, 4(6):479–486, 1989. (English version of the original paper in Russian from 1981).

[26] A. M. Matsokin and S. V. Nepomnyaschikh. The fictitious domain method and explicit continuation operator. *ŽVMiMF*, 33:45–59, 1993.

[27] S. V. Nepomnyaschikh. Method of splitting into subspaces for solving elliptic boundary value problems in complex–form domains. *Sov. J. Numer. Anal. Math. Modelling*, 6:151–168, 1991.

[28] S. V. Nepomnyaschikh. Fictitious space method on unstructured meshes. *East–West J. Numer. Math.*, 3(1):71–79, 1995.

[29] S. V. Nepomnyaschikh. Preconditioning operators on unstructured grids. In N. D. Melson, T. A. Manteuffel, S. F. McCormick, and C. C. Douglas, editors, *Seventh Copper Mountain Conference on Multigrid Methods*, number 3339 in NASA Conference Publication, pages 607–621, 1996. (also available as Preprint 178, Weierstraß–Institut Berlin, 1995).

[30] L. A. Oganesyan and L. A. Rukhovets. *Variational-difference methods for the solution of elliptic equations.* Izd. Akad. Nauk Armianskoi SSR, Jerevan, 1979. (Russian).

[31] P. Oswald. *Multilevel Finite Element Approximation: Theory and Applications.* Teubner Skripten zur Numerik. B. G. Teubner Stuttgart, 1994.

[32] J. Xu. Iterative methods by space decomposition and subspace correction. *SIAM Review*, 34(4):581–613, 1992.

[33] J. Xu. The auxiliary space method and optimal multigrid preconditioning techniques for unstructured grids. *Computing*, 56:215–235, 1996.

[34] G. N. Yakovlev. On traces of piecewise-smooth surfaces of functions from the space W_p^l. *Mat. Sbornik*, 74:526–543, 1967.

[35] X. Zhang. Multilevel Schwarz methods. *Numer. Math.*, 63:521–539, 1992.

Other titles in the SFB393 series:

96-01 V. Mehrmann, H. Xu. Chosing poles so that the single-input pole placement problem is well-conditioned. Januar 1996.

96-02 T. Penzl. Numerical solution of generalized Lyapunov equations. January 1996.

96-03 M. Scherzer, A. Meyer. Zur Berechnung von Spannungs- und Deformationsfeldern an Interface-Ecken im nichtlinearen Deformationsbereich auf Parallelrechnern. March 1996.

96-04 Th. Frank, E. Wassen. Parallel solution algorithms for Lagrangian simulation of disperse multiphase flows. Proc. of 2nd Int. Symposium on Numerical Methods for Multiphase Flows, ASME Fluids Engineering Division Summer Meeting, July 7-11, 1996, San Diego, CA, USA. June 1996.

96-05 P. Benner, V. Mehrmann, H. Xu. A numerically stable, structure preserving method for computing the eigenvalues of real Hamiltonian or symplectic pencils. April 1996.

96-06 P. Benner, R. Byers, E. Barth. HAMEV and SQRED: Fortran 77 Subroutines for Computing the Eigenvalues of Hamiltonian Matrices Using Van Loans's Square Reduced Method. May 1996.

96-07 W. Rehm (Ed.). Portierbare numerische Simulation auf parallelen Architekturen. April 1996.

96-08 J. Weickert. Navier-Stokes equations as a differential-algebraic system. August 1996.

96-09 R. Byers, C. He, V. Mehrmann. Where is the nearest non-regular pencil? August 1996.

96-10 Th. Apel. A note on anisotropic interpolation error estimates for isoparametric quadrilateral finite elements. November 1996.

96-11 Th. Apel, G. Lube. Anisotropic mesh refinement for singularly perturbed reaction diffusion problems. November 1996.

96-12 B. Heise, M. Jung. Scalability, efficiency, and robustness of parallel multilevel solvers for nonlinear equations. September 1996.

96-13 F. Milde, R. A. Römer, M. Schreiber. Multifractal analysis of the metal-insulator transition in anisotropic systems. October 1996.

96-14 R. Schneider, P. L. Levin, M. Spasojević. Multiscale compression of BEM equations for electrostatic systems. October 1996.

96-15 M. Spasojević, R. Schneider, P. L. Levin. On the creation of sparse Boundary Element matrices for two dimensional electrostatics problems using the orthogonal Haar wavelet. October 1996.

96-16 S. Dahlke, W. Dahmen, R. Hochmuth, R. Schneider. Stable multiscale bases and local error estimation for elliptic problems. October 1996.

96-17 B. H. Kleemann, A. Rathsfeld, R. Schneider. Multiscale methods for Boundary Integral Equations and their application to boundary value problems in scattering theory and geodesy. October 1996.

96-18 U. Reichel. Partitionierung von Finite-Elemente-Netzen. November 1996.

96-19 W. Dahmen, R. Schneider. Composite wavelet bases for operator equations. November 1996.

96-20 R. A. Römer, M. Schreiber. No enhancement of the localization length for two interacting particles in a random potential. December 1996. to appear in: Phys. Rev. Lett., March 1997

96-21 G. Windisch. Two-point boundary value problems with piecewise constant coefficients: weak solution and exact discretization. December 1996.

96-22 M. Jung, S. V. Nepomnyaschikh. Variable preconditioning procedures for elliptic problems. December 1996.

The complete list of current and former preprints is available via `http://www.tu-chemnitz.de/sfb393/sfb97pr.html`.