J Sci Comput (2009) 39: 340–370
DOI 10.1007/s10915-009-9272-6

Comparison of Two-Level Preconditioners Derived from Deflation, Domain Decomposition and Multigrid Methods

J.M. Tang · R. Nabben · C. Vuik · Y.A. Erlangga

Received: 6 December 2008 / Revised: 7 January 2009 / Accepted: 12 January 2009 /
Published online: 27 January 2009

Abstract For various applications, it is well-known that a multi-level, in particular two-level, preconditioned CG (PCG) method is an efficient method for solving large and sparse linear systems with a coefficient matrix that is symmetric positive definite. The corresponding two-level preconditioner combines traditional and projection-type preconditioners to get rid of the effect of both small and large eigenvalues of the coefficient matrix. In the literature, various two-level PCG methods are known, coming from the fields of deflation, domain decomposition and multigrid. Even though these two-level methods differ a lot in their specific components, it can be shown that from an abstract point of view they are closely related to each other. We investigate their equivalences, robustness, spectral and convergence properties, by accounting for their implementation, the effect of roundoff errors and their sensitivity to inexact coarse solves, severe termination criteria and perturbed starting vectors.

Part of this work has been done during the visit of the first, third and fourth author at Technische Universität Berlin. The research is partially funded by the Dutch BSIK/BRICKS project and the Deutsche Forschungsgemeinschaft (DFG), Project NA248/2-2.

J.M. Tang · C. Vuik (✉)
Faculty of Electrical Engineering, Mathematics and Computer Science, Delft Institute of Applied Mathematics, Delft University of Technology, J.M. Burgerscentrum, Mekelweg 4, 2628 CD Delft, The Netherlands
e-mail: c.vuik@tudelft.nl

J.M. Tang
e-mail: jok.tang@gmail.com

R. Nabben
Institut für Mathematik, Technische Universität Berlin, MA 3-3, Straße des 17. Juni 136, 10623 Berlin, Germany
e-mail: nabben@math.tu-berlin.de

Y.A. Erlangga
Department of Earth and Ocean Sciences, The University of British Columbia, 6339 Stores Road, Vancouver, British Columbia, V6T 1Z4, Canada
e-mail: yerlangga@eos.ubc.ca

Keywords Deflation · Domain decomposition · Multigrid · Conjugate gradients · Two-grid schemes · Two-level preconditioning · SPD matrices · Two-level PCG methods

1 Introduction

The Conjugate Gradient (CG) method [17] is a very popular iterative method for solving large linear systems of equations,

$$Ax = b, \quad A = [a_{ij}] \in \mathbb{R}^{n\times n}, \tag{1}$$

whose coefficient matrix, A, is sparse and symmetric positive definite (SPD). The convergence rate of CG is bounded in terms of the condition number of A, i.e., after j iterations of CG, we have

$$\|x - x_j\|_A \leq 2\|x - x_0\|_A \left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^j, \tag{2}$$

where x_0 is the starting vector, $\kappa = \kappa(A)$ denotes the spectral condition number of A (which in this case is the ratio of the largest to the smallest eigenvalue), and $\|x\|_A$ is the A-norm of x, defined as $\|x\|_A = \sqrt{x^T Ax}$. If κ is large, it is more favorable to solve a preconditioned system instead of (1), where the preconditioned system is defined as $\widehat{A}\hat{x} = \hat{b}$, with $\widehat{A} = M^{-1/2}AM^{-1/2}$, $\hat{x} = M^{1/2}x$, $\hat{b} = M^{-1/2}b$ and $M^{-1} \in \mathbb{R}^{n\times n}$ is an SPD matrix. This system can be transformed into the system

$$M^{-1}Ax = M^{-1}b,$$

see e.g. [14, Sect. 10.3.1]. The preconditioner, M, should be chosen such that $M^{-1}A$ has a more clustered spectrum than A. Furthermore, systems of the form $My = z$ must be cheap to solve, relative to the improvement that they provide in the convergence rate. The resulting method is called the preconditioned CG (PCG) method.

Since so far there exists no universal preconditioner, which works for any type of problems, the design and analysis of preconditioners for CG remain of great interest. The diagonal scaling methods, basic iterative methods, approximate inverse and incomplete Cholesky preconditioners are examples of *traditional* preconditioners. In many cases, a traditional preconditioner is often combined with a two-level component or a second level correction, which leads to a two-level PCG method. This two-level correction is also known as a coarse-grid correction or subspace correction. Examples of the two-level preconditioners are multigrid methods (MG), domain decomposition methods (DDM), and deflated or augmented Krylov subspace methods. For an overview paper we refer to [48].

Two-grid or multigrid preconditioning has been known for a long time, dating back at least to the 1930s. Its potential was first exploited by Fedorenko and Bakhalov in the 1960s, and later by Brandt [3] and Hackbusch [16], which paved the way to the birth of multigrid methods. We refer to [44, 47] and references therein for more details. Also for domain decomposition methods it was shown in [2] that adding a coarse-grid correction or a second level correction can lead to a significant improvement in the convergence rate. We refer to [36, 43] for more details. In MG and DDM, there exist several ways of incorporating the coarse-grid correction to the traditional preconditioner, which include the additive coarse-grid correction [2, 5, 6] and the multiplicative or balancing version of DDM as proposed in [21].

Another example of two-level preconditioners for Krylov methods is deflation. First used by Nicolaides to accelerate the convergence of CG [31], several contributions have been made since then, including [8, 19, 34, 45]. Following [31], the convergence of CG can be improved if the components of the residual or error associated with the smallest eigenvalues are no longer present during the iteration. To achieve this, these smallest eigenvalues need to be deflated explicitly (shift to zero) by augmenting the Krylov subspace with the corresponding eigenvectors. It can be shown that the convergence of CG is then bounded in terms of the effective condition number of the deflated linear system, which is the ratio of the largest eigenvalue to the smallest, nonzero eigenvalue.

From an implementation point of view, two-level PCG methods from deflation, DDM and MG seem to be different. For example, in deflation, eigenvectors or eigenvector approximations associated with unfavorable eigenvalues are often used as projection vectors. These projection vectors then form the operator that is acting between the subspace and the original space. In contrast, MG or DDM use interpolation operators between the fine-grid (original) and coarse-grid subspace. Generally speaking, each field has developed not only specific components but also some specific ways and implementations for combining the traditional preconditioner with the second level correction. Interestingly, however, from algebraic or abstract point of view, the two-level PCG methods from the three fields are quite comparable or even equivalent, as we will see throughout our discussion. This observation motivates us to compare these two-level methods and their implementations in detail. In this paper, we try to bridge the gap between these different fields by theoretically comparing them and investigating some specific variants and implementations of these methods.

Different from [20], where a comparison between the original deflation, MG and DDM methods (using their specific ingredients) is given, in this paper, we approach the two-level PCG methods from an abstract point of view. We will generally consider the methods from their algebraic components, and will not particularly focus on their specific components. This means that we do not use, e.g., the fact that the deflation method uses specific eigenvectors or approximate eigenvectors and the fact that the MG and DDM methods use specific interpolation matrices, since either choice leads to algebraically the same component. To establish this algebraic comparison, we will introduce a generalized formulation for deflation, MG and DDM, which allows us to derive a unified theory. The comparison given here, has led to a new method introduced in [9]. There, two of the authors combined advantages of these three types of methods.

We note that in [28–30], theoretical comparisons have been given for the deflation, abstract balancing and additive coarse-grid correction methods. It has been proven, using spectral analysis among other techniques, that the deflation method is expected to yield faster convergence compared to the other two methods, provided that the second level correction is solved very accurately. It is not uncommon, however, that such a solve is impractical due to the large size of the problem. This implementation aspect among others was not particularly discussed in those papers. In this paper, in addition to theoretical comparisons, we will also investigate some important aspects related to numerical implementations. We will see via numerical experiments, e.g., that two algebraically equivalent methods do not necessarily lead to the same convergence behavior. In particular, we will address the following issues:

- what are the relations and equivalences between the two-level PCG methods?
- which two-level PCG methods can be applied, if one uses inaccurate coarse solvers, severe termination criteria or perturbed starting vectors?
- is there a two-level preconditioner, that is robust and cheap, or in other words, is there a way to add the coarse grid correction, that is robust and cheap?

This paper is organized as follows. In Sect. 2, we introduce and discuss two-level PCG methods in a unified way. Section 3 is devoted to the theoretical comparison of these methods. Subsequently, the numerical comparison of the two-level PCG methods is carried out in Sect. 4. Finally, conclusions are given in Sect. 5.

2 Two-Level PCG methods

In this section, two-level PCG methods will be defined and justified, but we start with some terminology and a preliminary result, which are commonly used in the analysis of two-level preconditioners.

Definition 2.1 Suppose that an SPD coefficient matrix, $A \in \mathbb{R}^{n\times n}$, and a projection subspace matrix, $Z \in \mathbb{R}^{n\times k}$, with full rank and $k < n$ are given. Then, we define the invertible matrix $E \in \mathbb{R}^{k\times k}$, the matrix $Q \in \mathbb{R}^{n\times n}$, and the projection matrix, $P \in \mathbb{R}^{n\times n}$, as follows:

$$P := I - AQ, \qquad Q := ZE^{-1}Z^T, \qquad E := Z^T AZ,$$

where I is the $n \times n$ identity matrix. In addition, $M \in \mathbb{R}^{n\times n}$ is an SPD matrix that is called the preconditioner.

Lemma 2.1 *Let A, Z, Q and P be as in Definition 2.1. Then, the following equalities hold*:

(a) $P = P^2$;
(b) $PA = AP^T$;
(c) $P^T Z = \mathbf{0},\ P^T Q = \mathbf{0}$;
(d) $PAZ = \mathbf{0},\ PAQ = \mathbf{0}$;
(e) $QA = I - P^T,\ QAZ = Z,\ QAQ = Q$;
(f) $Q^T = Q$.

Proof The proof of these standard results can be found in, e.g., [37, 45]. □

Note that E is SPD for any full-rank Z, since A is SPD. If $k \ll n$ holds, then E is a matrix with small dimensions, so that it can be easily computed and factored. Moreover, PA always has k zero eigenvalues according to Lemma 2.1(d).

2.1 Background of the Matrices in Domain Decomposition, Multigrid and Deflation

While Definition 2.1 seems to be very specific to deflation, algebraic/abstract formulation of two-level PCG methods will require the matrices as defined there, in one form or another. From an abstract point of view, all two-level preconditioners of the methods will consist of an arbitrary M, combined with one or more matrices P and Q. Below, we will give an explanation of the choices for these matrices in the different fields. Nevertheless, from our point of view, matrices M and Z are arbitrary (but fixed) for each two-level PCG method. In this way, the abstract setting allows us to compare the methods in terms of operators, although they have their roots in different fields.

A standard CG method based on deflation is obtained if CG is applied to the coefficient matrix PA. The matrix Z consists of so-called projection vectors, whose columns span the projection subspace. It often consists of eigenvectors, approximations of eigenvectors, or piecewise-constant vectors, which are strongly related to DDM. If one chooses

eigenvectors, the corresponding eigenvalues would be shifted to zero in the spectrum of the deflated matrix. This fact has motivated the name 'deflation method'. Usually, systems with E are solved directly, using, e.g., a Cholesky decomposition. Moreover, in the actual implementation, a traditional preconditioner, M, is often incorporated to further improve the convergence. In this case, CG should solve the system based on $M^{-1}PA$ or $P^TM^{-1}A$. We will detail this is Sect. 2.3.2. For deflation, M can be of the form, e.g., of an incomplete factorization of A. In the literature, the deflation two-level preconditioner is also known as the spectral preconditioner, see, e.g., [13].

In the two-level PCG methods used in DDM, such as the balancing Neumann-Neumann and (two-level) additive Schwarz methods, the preconditioner, M, consists of the local exact or inexact solves on subdomains. Moreover, Z describes a prolongation (or interpolation) operator, while Z^T is a restriction operator based on the subdomains. In this case, E is called the coarse-grid (or Galerkin) matrix. In order to speed up the convergence of the additive Schwarz method, a coarse-grid correction matrix, Q, can be added, which is a so-called additive coarse-grid correction. Finally, P can be interpreted as a subspace correction, in which each subdomain is agglomerated into a single cell. More details can be found in [36, 43].

In the MG approach, Z and Z^T are also the prolongation and restriction operators, respectively, where typical MG grid-transfer operators allow interpolation between neighboring subdomains. E and Q are again the coarse-grid (or Galerkin) and coarse-grid correction matrices, respectively, corresponding to the Galerkin approach. The matrix P can be interpreted as the algebraic form of the coarse-grid correction step in MG, where linear systems with E are usually solved recursively. In the context of MG, M^{-1} should work as a smoother that eliminates the high-frequency errors in the residuals and often corresponds to Jacobi or Gauss-Seidel iterations. Before or after the smoothing step(s), a coarse-grid correction, P, is applied to remove the slow-frequencies in the residuals. We refer to [16, 44, 47] for more details.

2.2 General Linear Systems

The general linear system, that is the basis for two-level PCG methods, is

$$\mathcal{P}\mathcal{A}\mathbf{x} = \mathbf{b}, \quad \mathcal{P}, \mathcal{A} \in \mathbb{R}^{n\times n}. \tag{3}$$

In the standard preconditioned CG method, $\mathbf{x} = x$ is the solution of the original linear system, $Ax = b$, $\mathcal{A} = A$ is the SPD coefficient matrix, $\mathcal{P} = M^{-1}_{\text{PREC}}$ represents a traditional SPD preconditioner, and $\mathbf{b} = M^{-1}_{\text{PREC}}b$ is the right-hand side. We will call this method 'Traditional Preconditioned CG' (PREC), see also [14, 26].

Next, $\mathcal{A}$ may also be a combination of A and P, such that $\mathcal{A}$ is symmetric positive (semi-) definite (SP(S)D), while $\mathcal{P}$ remains a traditional preconditioner. Note that this does not cause difficulties for CG, since it is robust for SPSD matrices, as long as the linear system is consistent [18]. Furthermore, instead of choosing one traditional preconditioner for $\mathcal{P}$, we can combine different traditional preconditioners and projection matrices, P and Q, in an additive or multiplicative way, which will be illustrated below.

The additive combination of two SPD preconditioners, C_1 and C_2, leads to $\mathcal{P}_{a_2}$, given by

$$\mathcal{P}_{a_2} := C_1 + C_2, \tag{4}$$

which is also SPD. Of course, the summation of the preconditioners can be done with different weights for C_1 and C_2. Moreover, (4) can be easily generalized to $\mathcal{P}_{a_i}$ for more SPD preconditioners, $C_1, C_2, \ldots, C_i$.

The multiplicative combination of preconditioners can be explained by considering the stationary iterative methods induced by the preconditioner. Assuming that C_1 and C_2 are two SPD preconditioners, we can combine $x^{i+\frac{1}{2}} := x^i + C_1(b - Ax^i)$ and $x^{i+1} := x^{i+\frac{1}{2}} + C_2(b - Ax^{i+\frac{1}{2}})$ to obtain $x^{i+1} = x^i + \mathcal{P}_{m_2}(b - Ax^i)$, with

$$\mathcal{P}_{m_2} := C_1 + C_2 - C_2AC_1, \tag{5}$$

which can be interpreted as the multiplicative operator consisting of two preconditioners. Subsequently, C_1 and C_2 could again be combined with another SPD preconditioner, C_3, in a multiplicative way, yielding

$$\mathcal{P}_{m_3} = C_1 + C_2 + C_3 - C_2AC_1 - C_3AC_2 - C_3AC_1 + C_3AC_2AC_1. \tag{6}$$

This can also be generalized to $\mathcal{P}_{m_i}$ for $C_1, C_2, \ldots, C_i$.

2.3 Definition of the Two-Level PCG Methods

The two-level PCG methods that will be considered in this paper are given and justified below.

2.3.1 Additive Method

If one substitutes a traditional preconditioner, $C_1 = M^{-1}$, and a coarse-grid correction matrix, $C_2 = Q$, into the additive combination given in (4), this gives

$$\mathcal{P}_{\text{AD}} = M^{-1} + Q. \tag{7}$$

Using the additive Schwarz preconditioner for M, the abstract form (7) includes the additive coarse-grid correction preconditioner [2]. The BPS preconditioner, introduced by Bramble, Pasciak and Schatz in [2], can be written as (7). This has further been analyzed in, e.g., [5, 6, 32]. If the multiplicative Schwarz preconditioner is taken as M, we obtain the Hybrid-2 preconditioner [43, p. 47]. In the MG language, $\mathcal{P}_{\text{AD}}$ is sometimes called an additive multigrid preconditioner, see [1]. In this paper, the resulting method, associated with $\mathcal{P}_{\text{AD}}$, will be called 'Additive Coarse-Grid Correction' (AD).

2.3.2 Deflation Methods

The deflation technique has been exploited by several authors [11, 12, 19, 24, 25, 27–29, 31, 34, 45]. Below, we first describe the deflation method following [45] and, thereafter, [19, 31, 34].

First note that, from Lemma 2.1, $Q = Q^T$, $(I - P^T)x = Qb$ and $AP^T = PA$ hold. Then, in order to solve $Ax = b$, we write $x = (I - P^T)x + P^Tx$ where $(I - P^T)x$ can be computed immediately. For P^Tx, we solve the deflated system,

$$PA\tilde{x} = Pb. \tag{8}$$

Obviously, (8) is singular, and it can only be solved by CG if it is consistent, see also [18]. Since matrix A is nonsingular and $Ax = b$ is consistent, this is certainly true for (8), where the same projection is applied to both sides of the nonsingular system. If A was singular, this projection could also be applied in many cases, see [39–42]. Then, because $P^T\tilde{x} = P^Tx$,

the unique solution, x, can be obtained via (8), by premultiplying $\tilde{x}$ by P^T, and adding it to Qb, i.e., $x = Qb + P^T\tilde{x}$. Subsequently, the deflated system can also be solved, using a preconditioner, M, which gives

$$M^{-1}PA\tilde{x} = M^{-1}Pb, \tag{9}$$

see [45] for details. Linear system (9) can also be written in the form of (3), by taking $\mathcal{P} = M^{-1}$, $\mathcal{A} = PA$ and $\mathbf{b} = M^{-1}Pb$. Note that this is well-defined, since PA is an SPSD matrix. The resulting method will be called 'Deflation Variant 1' (DEF1).

An alternative way to describe the deflation technique is to start with an arbitrary vector, $\bar{x}$, and choose $x_0 := Qb + P^T\bar{x}$. Then, the solution of $Ax = b$ can be constructed from the deflated system

$$AP^T y = r_0, \quad r_0 := b - Ax_0. \tag{10}$$

The non-unique solution, y, is then used to obtain $\bar{y} = P^T y$. It can be shown that $x = x_0 + \bar{y}$ is the unique solution of $Ax = b$. Similarly, the deflated system (10) can also be solved with preconditioner M, leading to

$$M^{-1}AP^T y = M^{-1}r_0, \quad r_0 := b - Ax_0. \tag{11}$$

Similar to the procedure for the unpreconditioned case, x can be found from the non-uniquely determined solution, y, of (11). This leads to an algorithm that is based on the projection operator $P^T M^{-1}$, rather than $M^{-1}P$ as obtained in the first deflation variant above, see also [19, 31, 34] for more details. Hence, we solve

$$P^T M^{-1}Ax = P^T M^{-1}b, \tag{12}$$

where the iterates x_k within the algorithm are uniquely determined as long as $x_0 := Qb + P^T\bar{x}$ is used. We will treat this in more detail in Sect. 3.2. The resulting method will be denoted as 'Deflation Variant 2' (DEF2). Observe that (12) cannot be written in the form of (3), with an SPD operator $\mathcal{P}$ and an SPSD matrix $\mathcal{A}$. Fortunately, in Sect. 3.2, it will be shown that (12) is equivalent to a linear system, that is in the form of (3).

The main difference between DEF1 and DEF2 is their flipped two-level preconditioner. In addition, if we define the 'uniqueness'-operation as computing $v = Qb + P^T\tilde{v}$, for a given vector $\tilde{v}$, this operation is carried out at the end of the iteration process in DEF1, so that an arbitrarily chosen starting vector, x_0, can be used. On the other hand, this operation has been applied prior to the iteration process in DEF2, which can be interpreted as adopting a special starting vector. As a consequence, they have different robustness properties with respect to starting vectors, see Sect. 4.6.

2.3.3 Adapted Deflation Methods

If one applies $C_1 = Q$ and $C_2 = M^{-1}$ in a multiplicative combination, as given in (5), then this yields

$$\mathcal{P}_{\text{A-DEF1}} = M^{-1}P + Q, \tag{13}$$

see [37] for more details. In the MG language, this operator results from a non-symmetric multigrid V(1, 0)-cycle iteration scheme, where one first applies a coarse-grid correction, followed by a smoothing step. Note that, although Q and M are SPD preconditioners, (13) is a non-symmetric operator and, even more, it is not symmetric with respect to the inner

product induced by A. In addition, $\mathcal{P}_{\text{A-DEF1}}$ can also be interpreted as an adapted deflation preconditioner, since $M^{-1}P$ from DEF1 is combined in an additive way with a coarse-grid correction, Q. Hence, the resulting method, corresponding to $\mathcal{P}_{\text{A-DEF1}}$, will be denoted as the 'Adapted Deflation Variant 1' (A-DEF1).

Subsequently, we can also reverse the order of Q and M^{-1} in (5), i.e., choose $C_1 = M^{-1}$ and $C_2 = Q$, giving

$$\mathcal{P}_{\text{A-DEF2}} = P^T M^{-1} + Q. \tag{14}$$

Using an additive Schwarz preconditioner for M, $\mathcal{P}_{\text{A-DEF2}}$ is the two-level Hybrid-II Schwarz preconditioner [36, p. 48]. In MG methods, $\mathcal{P}_{\text{A-DEF2}}$ is the non-symmetric multigrid V(0, 1)-cycle preconditioner, where M^{-1} is used as a smoother. Similar to A-DEF1, $\mathcal{P}_{\text{A-DEF2}}$ is non-symmetric. Fortunately, we will see in Sect. 3.2 that A-DEF2 is equivalent to a method based on a symmetric operator. As in the case of $\mathcal{P}_{\text{A-DEF1}}$, $\mathcal{P}_{\text{A-DEF2}}$ can also be seen as an adapted deflation preconditioner, since $P^T M^{-1}$ from DEF2 is combined with Q, in an additive way. Therefore, the resulting method will be called the 'Adapted Deflation Variant 2' (A-DEF2).

2.3.4 Abstract Balancing Methods

The operators $\mathcal{P}_{\text{A-DEF1}}$ and $\mathcal{P}_{\text{A-DEF2}}$ can be symmetrized, by using the multiplicative combination of three preconditioners. If one substitutes $C_1 = Q$, $C_2 = M^{-1}$ and $C_3 = Q$ into (6), we obtain

$$\mathcal{P}_{\text{BNN}} = P^T M^{-1} P + Q.$$

The operator $\mathcal{P}_{\text{BNN}}$ is a well-known operator in DDM. In combination with an additive Schwarz preconditioner for M, and after some scaling and special choices of Z, the operator $\mathcal{P}_{\text{BNN}}$ is known as the Balancing-Neumann-Neumann preconditioner, introduced in [21] and further analyzed, e.g., in [7, 22, 23, 33, 43]. In the abstract form, $\mathcal{P}_{\text{BNN}}$ is called the Hybrid-1 preconditioner [43, p. 34]. Here, we will call it 'Abstract Balancing Neumann-Neumann' (BNN).

Of course, $\mathcal{P}_{\text{A-DEF1}}$ and $\mathcal{P}_{\text{A-DEF2}}$ could also be symmetrized by using twice M^{-1} instead of Q (i.e., $C_1 = M^{-1}$, $C_2 = Q$ and $C_3 = M^{-1}$) in (6). This results in the well-known symmetric multigrid V(1,1)-cycle iteration scheme, where a pre-smoothing step is followed by a coarse-grid correction and ended with a post-smoothing step. The resulting preconditioner is then explicitly given by

$$\mathcal{P} = M^{-1}P + P^T M^{-1} + Q - M^{-1} P A M^{-1}. \tag{15}$$

Note that this operator also follows by combining the A-DEF1 and A-DEF2 operators in a multiplicative way. In (15), a structural difference can be observed between BNN and the multigrid V(1, 1)-cycle iteration. As mentioned before, in MG, the operator M^{-1} is the smoothing operator and the coarse-grid system typically has half of the order of the original system per direction. Hence, smoothing is cheap compared to solving the coarse-grid system. In this case, symmetrizing with another smoothing step is natural. In DDM, M^{-1} contains all local solves of the subdomain systems, while the dimension of the coarse system is typically much smaller than the dimension of the original system. Except for special choices of the restriction and prolongation operator, see, e.g., [4], it is generally difficult to analyze the spectra of the system preconditioned by (15) in comparison with the other methods described in this paper. Therefore, we do not include this preconditioner in our comparison. In [38] a detailed comparison is given.

Moreover, we will also consider two variants of BNN. In the first variant, we omit the term Q from $\mathcal{P}_{\text{BNN}}$, giving us

$$\mathcal{P}_{\text{R-BNN1}} = P^T M^{-1} P,$$

which remains a symmetric operator. To our knowledge, $\mathcal{P}_{\text{R-BNN1}}$ is unknown in the literature, and this is the first time that its properties are analyzed. The corresponding method is called 'Reduced Balancing Neumann-Neumann Variant 1' (R-BNN1). Next, in the second variant of BNN, we omit both the P and Q terms from $\mathcal{P}_{\text{BNN}}$, resulting in

$$\mathcal{P}_{\text{R-BNN2}} = P^T M^{-1}, \tag{16}$$

and this method will be denoted as 'Reduced Balancing Neumann-Neumann Variant 2' (R-BNN2). Notice that the operators of both R-BNN2 and DEF2 are equal, i.e., $\mathcal{P}_{\text{DEF2}} = \mathcal{P}_{\text{R-BNN2}} = P^T M^{-1}$, where only the implementation appears to be different, see Sect. 2.4.1. In fact, the implementation of DEF2 is equivalent to the approach as applied in, e.g., [34], where the deflation method has been derived by combining a deflated Lanczos procedure and the standard CG algorithm. On the other hand, R-BNN2 is the approach where deflation is incorporated into the CG algorithm in a direct way [19], and it is also the approach where a hybrid variant has been employed in DDM [43]. Finally, as mentioned earlier, $P^T M^{-1}$ is a non-symmetric preconditioner, but it will be shown in Sect. 3.2 that both $\mathcal{P}_{\text{R-BNN1}}$ and $\mathcal{P}_{\text{R-BNN2}}$ are equivalent to $\mathcal{P}_{\text{BNN}}$, for certain starting vectors. Hence, we classify these methods as variants of the original abstract balancing method, rather than as variants of deflation methods.

2.4 Aspects of Two-Level PCG Methods

For the sake of completeness, the two-level PCG methods that are considered here are given in Table 1. More details about the methods can be found in the references, given in the last column of this table. Subsequently, the implementation and the computational cost of these methods will be considered in this subsection.

Table 1 List of methods which will be compared. The operator of each method can be interpreted as the preconditioner $\mathcal{P}$, given in (3) with $\mathcal{A} = A$. Where possible, references to the methods and their implementations are given in the last column

Name	Method	Operator	References
PREC	Traditional Preconditioned CG	M^{-1}	[14, 26]
AD	Additive Coarse-Grid Correction	$M^{-1} + Q$	[2, 36, 43]
DEF1	Deflation Variant 1	$M^{-1}P$	[45]
DEF2	Deflation Variant 2	$P^T M^{-1}$	[19, 31, 34]
A-DEF1	Adapted Deflation Variant 1	$M^{-1}P + Q$	[36, 44, 47]
A-DEF2	Adapted Deflation Variant 2	$P^T M^{-1} + Q$	[36, 44, 47]
BNN	Abstract Balancing	$P^T M^{-1} P + Q$	[21]
R-BNN1	Reduced Balancing Variant 1	$P^T M^{-1} P$	–
R-BNN2	Reduced Balancing Variant 2	$P^T M^{-1}$	[21, 43]

Algorithm 1 General Two-Level PCG Method for solving $Ax = b$.

Select arbitrary $\bar{x}$ and $\mathcal{V}_{\text{start}}, \mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3, \mathcal{V}_{\text{end}}$ from Table 2
$x_0 := \mathcal{V}_{\text{start}},\ r_0 := b - Ax_0$
$y_0 := \mathcal{M}_1 r_0,\ p_0 := \mathcal{M}_2 y_0$
for $j := 0, 1, \ldots,$ until convergence **do**
  $w_j := \mathcal{M}_3 A p_j$
  $\alpha_j := (r_j, y_j)/(p_j, w_j)$
  $x_{j+1} := x_j + \alpha_j p_j$
  $r_{j+1} := r_j - \alpha_j w_j$
  $y_{j+1} := \mathcal{M}_1 r_{j+1}$
  $\beta_j := (r_{j+1}, y_{j+1})/(r_j, y_j)$
  $p_{j+1} := \mathcal{M}_2 y_{j+1} + \beta_j p_j$
end for
$x_{\text{it}} := \mathcal{V}_{\text{end}}$

Table 2 Choices of $\mathcal{V}_{\text{start}}, \mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3, \mathcal{V}_{\text{end}}$ for each method, as used in Algorithm 1

Method	$\mathcal{V}_{\text{start}}$	$\mathcal{M}_1$	$\mathcal{M}_2$	$\mathcal{M}_3$	$\mathcal{V}_{\text{end}}$
PREC	$\bar{x}$	M^{-1}	I	I	x_{j+1}
AD	$\bar{x}$	$M^{-1} + Q$	I	I	x_{j+1}
DEF1	$\bar{x}$	M^{-1}	I	P	$Qb + P^T x_{j+1}$
DEF2	$Qb + P^T \bar{x}$	M^{-1}	P^T	I	x_{j+1}
A-DEF1	$\bar{x}$	$M^{-1}P + Q$	I	I	x_{j+1}
A-DEF2	$Qb + P^T \bar{x}$	$P^T M^{-1} + Q$	I	I	x_{j+1}
BNN	$\bar{x}$	$P^T M^{-1} P + Q$	I	I	x_{j+1}
R-BNN1	$Qb + P^T \bar{x}$	$P^T M^{-1} P$	I	I	x_{j+1}
R-BNN2	$Qb + P^T \bar{x}$	$P^T M^{-1}$	I	I	x_{j+1}

Remark 2.1 We emphasize that the parameters of the two-level PCG methods that will be compared can be arbitrary, so that the comparison between these methods is based on their abstract versions. This means that the results of the comparison are valid for any full-rank matrix Z and SPD matrices A and M.

2.4.1 Implementation Issues

The general implementation of the two-level PCG methods given in Table 1 can be found in Algorithm 1. For each method, the corresponding matrices, $\mathcal{M}_i$, and vectors, $\mathcal{V}_{\text{start}}$ and $\mathcal{V}_{\text{end}}$, are presented in Table 2. For more details, we refer to [37].

From Algorithm 1 and Table 2, it can be observed that one or more preconditioning and projection operations are carried out in the steps where the terms $\mathcal{M}_i$, with $i = 1, 2, 3$, are involved. For most two-level PCG methods, these steps are combined to obtain the preconditioned/projected residuals, y_{j+1}. DEF2 is the only method where a projection step is applied to the search directions, p_{j+1}. Likewise, DEF1 is the only method where the projection is performed to create w_j. In this case, $r_{j+1} = P(b - Ax_{j+1})$ should hold, while $r_{j+1} = b - Ax_{j+1}$ is satisfied for the other methods. This does not lead to problems if one

wants to compare the two-level PCG methods in a fair way. This can be seen as follows. Denote first the iterates of DEF1 and any other method as $\tilde{x}_{j+1}$ and x_{j+1}, respectively. Then, for DEF1, we have

$$r_{j+1} = P(b - A\tilde{x}_{j+1}) = Pb - AP^T\tilde{x}_{j+1} = b - A(Qb + P^T\tilde{x}_{j+1}) = b - Ax_{j+1},$$

where $x_{j+1} = Qb + P^T\tilde{x}_{j+1}$ and $AP^T = PA$ have been used. Hence, $r_{j+1} = b - Ax_{j+1}$ is satisfied for all methods that we consider. In this case, the two-level PCG methods can be compared fairly by terminating the iterative process of each method when the norm of the relative residual, $\|r_{j+1}\|_2/\|r_1\|_2$, is below a tolerance, δ.

Moreover, notice that we use the same arbitrary starting vector, $\bar{x}$, in each method, but the actual starting vector, $\mathcal{V}_{\text{start}}$, may differ for each method. Finally, it can also be noticed that the ending vector, $\mathcal{V}_{\text{end}}$, is the same for all methods, except for DEF1.

Recall that $\mathcal{P}$, as given in (3), should be SPD to guarantee convergence of CG, see also [10]. This is obviously the case for PREC, AD, DEF1, and BNN. It can be shown that DEF2, A-DEF2, R-BNN1, and R-BNN2 also rely on appropriate operators, where it turns out that $\mathcal{V}_{\text{start}} = Qb + P^T\bar{x}$ plays an important role in this derivation, see Theorem 3.4. A-DEF1 is the only method which does not have an SPD operator, and which can also not be decomposed or transformed into an SPD operator, $\mathcal{P}$. Hence, it cannot be guaranteed that A-DEF1 always works, but it performs rather satisfactorily for most of the test cases considered in Sect. 4.

2.4.2 Computational Cost

The computational cost of each method depends not only on the choices of M and Z, but also on the implementation and on the storage of the matrices. It is easy to see that, for each iteration, PREC requires 1 matrix-vector multiplication (MVM), 2 inner products (IP), 3 vector updates (VU) and 1 preconditioning step.

Note that AZ and E should be computed and stored beforehand, so that only one MVM with A is required in each iteration of the two-level PCG methods. Moreover, we distinguish between two cases considering Z and AZ:

- Z is sufficiently sparse, so that Z and AZ can be stored in approximately two vectors;
- Z is full, so that Z and AZ are full matrices.

The first case, which is the best case in terms of efficiency, occurs often in DDM, where the columns of Z correspond to subdomains, while the second case occurs, for example, in (approximated) eigenvector deflation methods. In the coarse grid operators we use the following operations: $Z^T y$ and $(AZ)y$. In the sparse case the amount of work for both operations is equal to the cost of an inner product. If Z is full the costs are equal to a matrix-vector product where the matrix (Z or AZ) has dimensions $n \times k$. For this reason we have the column 'inner/matrix-vector multiplications' in Table 3. For each two-level PCG method, we give the extra computational cost per iteration above that of PREC, see Table 3. In the table, the number of operations of the form Py and Qy, for a given vector, y, per iteration is also provided. Note that, if both Py and Qy should be computed for the same vector, y, such as in A-DEF1, and BNN, then Qy can be determined efficiently, since it only requires one IP if Z is sparse, or one MVM if Z is full. Moreover, we remark that the given computational cost is based on the resulting abstract operators and implementation as presented in Algorithm 1.

From Table 3, it can be seen that AD is obviously the cheapest method per iteration, while BNN and R-BNN1 are the most expensive two-level PCG methods, since two operations

Table 3 Extra computational cost per iteration of the two-level PCG methods compared to PREC. IP = inner products, MVM = matrix-vector multiplications, VU = vector updates and CSS = coarse system solves. Note that IP holds for sparse Z and MVM holds for full Z

Method	Theory		Implementation		
	Py, $P^T y$	Qy	IP/MVM	VU	CSS
AD	0	1	2	0	1
DEF1	1	0	2	1	1
DEF2	1	0	2	1	1
A-DEF1	1	1	3	1	1
A-DEF2	1	1	4	1	2
BNN	2	1	5	2	2
R-BNN1	2	0	4	2	2
R-BNN2	1	0	2	1	1

with P and P^T are involved. With respect to the implementation, this implies that AD only needs 2 inner/matrix-vector products and 1 coarse system solve extra compared to PREC, while both BNN and R-BNN1 require obviously more inner/matrix-vector products, coarse system solves and additional vector updates. Note that in the 'theory' columns of Table 3 the computational cost is given, if the operators are used according to their definitions. The actual implementation can differ a lot as can be seen from the 'implementation' columns. Finally, observe that using a two-level PCG method is only efficient if Z is sparse, or if the number of projection vectors is relatively small in the case of a full matrix, Z.

3 Theoretical Comparison

In this section, a comparison of eigenvalue distributions of the two-level preconditioned matrices corresponding to the methods will be made. Thereafter, some relations between the abstract balancing method and the other methods will be derived. Some parts of the results are closely related to results known [28, 29, 43], but most of the presented results are new. We emphasize that all results presented in this section are valid for any full-rank matrix Z and SPD preconditioner M.

3.1 Spectral Analysis of the Methods

We start this subsection with some notation. Suppose that arbitrary matrices $C, D \in \mathbb{R}^{n\times n}$ have spectra $\sigma(C) := \{\lambda_1, \lambda_2, \ldots, \lambda_n\}$ and $\sigma(D) := \{\mu_1, \mu_2, \ldots, \mu_n\}$. The addition of the sets, $\sigma(C)$ and $\sigma(D)$, is defined as $\sigma(C) + \sigma(D) := \{\mu_1 + \lambda_1, \mu_2 + \lambda_2, \ldots, \mu_n + \lambda_n\}$. Suppose now that $B \in \mathbb{R}^{n\times n}$ is an arbitrary symmetric positive definite matrix with eigenvalues $\{\lambda_i\}$ for $i = 1, 2, \ldots, n$, sorted increasingly. Then, the (spectral) condition number, κ, of B is defined as $\kappa(B) := \|B\|_2\|B^{-1}\|_2 = \frac{\lambda_n}{\lambda_1}$. If B has s zero eigenvalues with $s < n$, then the effective condition number of B, $\tilde{\kappa}$, is defined as $\tilde{\kappa}(B) := \frac{\lambda_n}{\lambda_{s+1}}$.

In [46], it has been shown that

$$\tilde{\kappa}\left(M^{-1}PA\right) < \kappa\left(M^{-1}A\right),$$

for any SPD matrices A and M, and any Z with full rank. This means that the system corresponding to DEF1 is better conditioned than that of PREC. It will follow from the analysis

below that the system corresponding to PREC is always worse conditioned compared to all two-level PCG methods described in this paper.

In [28, 29], it has been shown that the effective condition number of DEF1 is not worse than the condition number of both AD and BNN, i.e.,

$$\tilde{\kappa}\left(M^{-1}PA\right) \leq \kappa\left(M^{-1}A + QA\right),$$
$$\tilde{\kappa}\left(M^{-1}PA\right) \leq \kappa\left(P^T M^{-1}PA + QA\right),$$

for all full-rank Z and SPD matrices A and M.

In addition to the comparisons of AD, DEF1, and BNN, performed in [28–30], more relations between the eigenvalue distribution of these and other two-level PCG methods are given below. We first show in Theorem 3.1, that DEF1, DEF2, R-BNN1, and R-BNN2 have identical spectra, and that the same is true for BNN, A-DEF1, and A-DEF2.

Theorem 3.1 *Suppose that $A \in \mathbb{R}^{n\times n}$ is SPD. Let M, Q and P be as in Definition 2.1. Then, the following two statements hold*:

- $\sigma(M^{-1}PA) = \sigma(P^T M^{-1}A) = \sigma(P^T M^{-1}PA)$;
- $\sigma((P^T M^{-1}P + Q)A) = \sigma((M^{-1}P + Q)A) = \sigma((P^T M^{-1} + Q)A)$.

Proof Note first that $\sigma(CD) = \sigma(DC)$, $\sigma(C + I) = \sigma(C) + \sigma(I)$ and $\sigma(C) = \sigma(C^T)$ hold, for arbitrary matrices $C, D \in \mathbb{R}^{n\times n}$, see also [37, Lemma 3.1]. Using these facts and Lemma 2.1, we obtain immediately that

$$\sigma\left(M^{-1}PA\right) = \sigma\left(AM^{-1}P\right) = \sigma\left(P^T M^{-1}A\right),$$

and that

$$\begin{aligned}\sigma\left(M^{-1}PA\right) &= \sigma\left(M^{-1}P^2A\right)\\ &= \sigma\left(M^{-1}PAP^T\right)\\ &= \sigma\left(P^T M^{-1}PA\right),\end{aligned}$$

which proves the first statement. Moreover, we also have that

$$\begin{aligned}\sigma\left(P^T M^{-1}PA + QA\right) &= \sigma\left(P^T M^{-1}PA - P^T + I\right)\\ &= \sigma\left((M^{-1}PA - I)P^T\right) + \sigma(I)\\ &= \sigma\left(M^{-1}P^2A - P^T\right) + \sigma(I)\\ &= \sigma\left(M^{-1}PA + QA\right),\end{aligned}$$

and, likewise,

$$\begin{aligned}\sigma\left(P^T M^{-1}A + QA\right) &= \sigma\left(P^T M^{-1}A - P^T\right) + \sigma(I)\\ &= \sigma\left(AM^{-1}P - P\right) + \sigma(I)\\ &= \sigma\left(PAM^{-1}P - P\right) + \sigma(I)\\ &= \sigma\left(P^T M^{-1}AP^T - P^T\right) + \sigma(I)\\ &= \sigma\left(P^T M^{-1}PA + QA\right),\end{aligned}$$

which completes the proof of the second statement. □

As a consequence of Theorem 3.1, DEF1, DEF2, R-BNN1, and R-BNN2 can be interpreted as one class of two-level PCG methods with the same spectral properties, whereas BNN, A-DEF1, and A-DEF2 lead to another class of two-level PCG methods. These two classes can be related to each other by [29, Theorem 2.8], which states that if $\sigma(M^{-1}PA) = \{0, \ldots, 0, \mu_{k+1}, \ldots, \mu_n\}$ is given, then $\sigma(P^T M^{-1} PA + QA) = \{1, \ldots, 1, \mu_{k+1}, \ldots, \mu_n\}$. We can even show that the reverse statement also holds. These two results are presented in Theorem 3.2.

Theorem 3.2 *Suppose that $A \in \mathbb{R}^{n\times n}$ are SPD. Let M, Q and P be as in Definition 2.1. Let the spectra of DEF1 and BNN be given by*

$$\sigma(M^{-1}PA) = \{\lambda_1, \ldots, \lambda_n\}, \qquad \sigma(P^T M^{-1} PA + QA) = \{\mu_1, \ldots, \mu_n\},$$

respectively. Then, the eigenvalues within these spectra can be ordered such that the following statements hold:

- $\lambda_i = 0$ *and* $\mu_i = 1$, *for* $i = 1, \ldots, k$;
- $\lambda_i = \mu_i$, *for* $i = k+1, \ldots, n$.

Proof The proof of this theorem goes along the same lines as that of [29, Theorem 2.8]. For the details we refer to [37]. □

Due to Theorem 3.2, both DEF1 and BNN provide almost the same spectra with the same clustering. The zero eigenvalues of DEF1 are replaced by eigenvalues equal to one in the case of BNN. Moreover, note that if $1 \in [\mu_{k+1}, \mu_n]$ then the effective condition numbers of BNN and DEF1 are identical. On the other hand, if $1 \notin [\mu_{k+1}, \mu_n]$, then DEF1 has a more favorable effective condition number compared to BNN. It appears that if eigenvalue 1 is an outlier, then it can take a number of iterations before superlinear convergence sets in and from that iteration on the effective condition numbers of both methods are the same (see Fig. 4.1 in [29]).

Next, Theorem 3.3 relates all methods in terms of their spectra and provides a strong connection between the two classes as given in Theorem 3.1.

Theorem 3.3 *Let the spectrum of DEF1, DEF2, R-BNN1, or R-BNN2 be given by* $\{0, \ldots, 0, \lambda_{k+1}, \ldots, \lambda_n\}$, *satisfying* $\lambda_{k+1} \leq \lambda_{k+2} \leq \cdots \leq \lambda_n$. *Let the spectrum of BNN, A-DEF1, or A-DEF2 be* $\{1, \ldots, 1, \mu_{k+1}, \ldots, \mu_n\}$, *with* $\mu_{k+1} \leq \mu_{k+2} \leq \cdots \leq \mu_n$. *Then,* $\lambda_i = \mu_i$ *for all* $i = k+1, \ldots, n$.

Proof The theorem follows immediately from Theorem 3.1 and 3.2. □

From Theorem 3.3, it can be concluded that all two-level PCG methods have almost the same clusters of eigenvalues. Therefore, we expect that the convergence of all methods will be similar, see Sect. 4.3 for some test cases. Moreover, the zeros in the spectrum of the first class (consisting of DEF1, DEF2, R-BNN1, or R-BNN2) might become nearly zero, due to roundoff errors or the approximate solution of coarse systems in the two-level preconditioner. This gives an unfavorable spectrum, resulting in slow convergence of the method. This phenomenon does not appear in the case of BNN, A-DEF1, or A-DEF2. Small perturbations in those two-level PCG methods lead to small changes in their spectra and

condition numbers. Theoretically, this can be analyzed using Z consisting of eigenvectors, see [28, Sect. 3], but, in general, it is difficult to examine for general Z. This issue will be further illustrated in Sects. 4.4 and 4.5 using numerical experiments.

3.2 Equivalences between the Methods

In this subsection, it will be shown that DEF2, A-DEF2, R-BNN1, and R-BNN2 produce identical iterates in exact arithmetic. More importantly, we will prove that these two-level PCG methods are equivalent to the more expensive BNN method for certain starting vectors. First, Lemma 3.1 shows that some steps in the BNN implementation can be reduced, see also [21] and [43, Sect. 2.5.2].

Lemma 3.1 *Let Q and P be as in Definition 2.1. Suppose that $\mathcal{V}_{start} = Qb + P^T\bar{x}$ instead of $\mathcal{V}_{start} = \bar{x}$ is used in BNN, where $\bar{x} \in \mathbb{R}^n$ is an arbitrary vector. Then*

- $Qr_{j+1} = \mathbf{0}$;
- $Pr_{j+1} = r_{j+1}$,

for all $j = -1, 0, 1, \ldots$, in the BNN implementation of Algorithm 1.

Proof For the first statement, the proof is as follows. It can be verified that $Qr_0 = \mathbf{0}$ and $QAp_0 = \mathbf{0}$. By the inductive hypothesis, $Qr_j = \mathbf{0}$ and $QAp_j = \mathbf{0}$ hold. Then, for the inductive step, we obtain $Qr_{j+1} = \mathbf{0}$ and $QAp_{j+1} = \mathbf{0}$, since $Qr_{j+1} = Qr_j - \alpha_j QAp_j = \mathbf{0}$, and

$$QAp_{j+1} = QAy_{j+1} + \beta_j QAp_j = QAP^T M^{-1} Pr_{j+1} + QAQr_{j+1} = \mathbf{0},$$

where we have used Lemma 2.1.

Next, for the second statement, $Pr_0 = r_0$ and $PAp_0 = Ap_0$ can be easily shown. Assume that $Pr_j = r_j$ and $PAp_j = Ap_j$. Then, both $Pr_{j+1} = r_{j+1}$ and $PAp_{j+1} = Ap_{j+1}$ hold, because

$$Pr_{j+1} = Pr_j - \alpha_j PAp_j = r_j - \alpha_j Ap_j = r_{j+1},$$

and

$$\begin{aligned} PAp_{j+1} &= PAy_{j+1} + \beta_j PAp_j \\ &= PAP^T M^{-1} Pr_{j+1} + \beta_j Ap_j \\ &= AP^T M^{-1} r_{j+1} + \beta_j Ap_j \\ &= AP^T M^{-1} Pr_{j+1} + \beta_j Ap_j \\ &= A(y_{j+1} + \beta_j p_j) \\ &= Ap_{j+1}, \end{aligned}$$

where we have applied the result of the first statement. This concludes the proof. □

Subsequently, we will provide a more detailed comparison between BNN and DEF1, in terms of errors in the A-norm, see Lemma 3.2. In fact, it is a generalization of [29, Theorems 3.4 and 3.5], where we now apply arbitrary starting vectors, $\bar{x}$, instead of zero starting vectors.

Lemma 3.2 *Suppose that $A \in \mathbb{R}^{n\times n}$ is SPD. Let Q and P be as in Definition 2.1. Let $(x_{j+1})_{DEF1}$ and $(x_{j+1})_{BNN}$ denote iterates x_{j+1} of DEF1 and BNN given in Algorithm 1, respectively. Then, they satisfy*

- $\|x - (x_{j+1})_{DEF1}\|_A \leq \|x - (x_{j+1})_{BNN}\|_A$, *if* $(x_0)_{DEF1} = (x_0)_{BNN}$;
- $(x_{j+1})_{DEF1} = (x_{j+1})_{BNN}$, *if* $(x_0)_{DEF1} = \bar{x}$ *and* $(x_0)_{BNN} = Qb + P^T\bar{x}$;

Proof The proof is analogous to the proofs as given in [29, Theorems 3.4 and 3.5]. □

From Lemma 3.2, we conclude that the errors of the iterates built by DEF1 are never larger than those of BNN in the A-norm. Additionally, DEF1 and BNN produce the same iterates in exact arithmetic, if $\mathcal{V}_{\text{start}} = Qb + P^T\bar{x}$ is used in BNN.

Next, Lemma 3.1 and 3.2 can now be combined to obtain the following important result.

Theorem 3.4 *Let Q and P be as in Definition 2.1. Let $\bar{x} \in \mathbb{R}^n$ be an arbitrary vector. The following methods produce exactly the same iterates in exact arithmetic*:

- *BNN with* $\mathcal{V}_{start} = Qb + P^T\bar{x}$;
- *DEF2, A-DEF2, R-BNN1 and R-BNN2 (with* $\mathcal{V}_{start} = Qb + P^T\bar{x}$);
- *DEF1 (with* $\mathcal{V}_{start} = \bar{x}$*) whose iterates are based on* $Qb + P^T x_{j+1}$.

Proof The theorem follows immediately from Lemma 3.1 and 3.2. □

As a result of Theorem 3.4, BNN is mathematically equivalent to R-BNN1, R-BNN2, A-DEF2 and DEF2, if $\mathcal{V}_{\text{start}} = Qb + P^T\bar{x}$ is used. They even produce the same iterates as DEF1, if its iterates, x_{j+1}, are transformed into $Qb + P^T x_{j+1}$. This will be illustrated in Sect. 4.3.

Another consequence of Theorem 3.4 is that the corresponding operators for DEF2, A-DEF2, R-BNN1 and R-BNN2 are all appropriate in a certain subspace, although they are not symmetric. Hence, CG in combination with these operators should, in theory, work fine. In Sect. 4.6 we investigate the results of Theorem 3.4 if rounding errors are involved.

4 Numerical Comparison

In this section, a numerical comparison of the two-level PCG methods will be performed. We consider two test problems, a 2-D porous media and a 2-D bubbly flow problem.

4.1 Test Problems and Choice of Projection Vectors

The main differential equation in both porous media and bubbly flow problem is the following elliptic equation with a discontinuous coefficient,

$$-\nabla \cdot (\mathcal{K}(\mathbf{x})\nabla p(\mathbf{x})) = \mathbf{0}, \quad \mathbf{x} = (x, y) \in \Omega = (0, 1)^2, \tag{17}$$

where p denotes the pressure, and $\mathcal{K}$ is a piecewise-constant coefficient that is equal to

$$\mathcal{K}(\mathbf{x}) = \begin{cases} \sigma(\mathbf{x}), & \sigma \text{ is the permeability in porous media flows;} \\ \frac{1}{\rho(\mathbf{x})}, & \rho \text{ is the density in bubbly flows.} \end{cases}$$

The exact description of the problems and the corresponding choices for projection vectors are given below.

A standard second-order finite-difference scheme is applied to discretize (17), where we use a uniform Cartesian grid. This results in our main linear system, $Ax = b$, with $A \in \mathbb{R}^{n\times n}$. Moreover, we choose as preconditioner, M, the Incomplete Cholesky decomposition without fill-in [26], IC(0), but it seems that other traditional SPD preconditioners could also be used instead, leading to similar results, see [45].

4.1.1 Porous Media Flow

In the porous media flow problem, Ω consists of equal shale and sandstone layers with uniform thickness, see Fig. 1(a). The contrast, which is the ratio between the high and low permeabilities, is $\epsilon = 10^6$. We impose a Dirichlet condition on the boundary $y = 1$ and homogeneous Neumann conditions on the other boundaries. The layers are denoted by the disjoint sets, Ω_j, $j = 1, 2, \ldots, k$, such that $\overline{\Omega} = \bigcup_{j=1}^{k} \overline{\Omega}_j$. The discretized domain and layers are denoted by Ω_h and Ω_{h_j}, respectively.

The projection vectors are chosen such that they are strongly related to the geometry of the problem. For each Ω_{h_j}, with $j = 1, 2, \ldots, k$, a projection vector, z_j, is defined as follows:

$$(z_j)_i := \begin{cases} 0, & x_i \in \Omega_h \setminus \overline{\Omega}_{h_j}; \\ 1, & x_i \in \Omega_{h_j}, \end{cases} \tag{18}$$

where x_i is a grid point of Ω_h. In this case, each projection vector corresponds to a unique layer, see also Fig. 2(a). With (18) we then set $Z := [z_1\ z_2\ \cdots\ z_k]$, thus the columns of Z consists of orthogonal disjoint piecewise-constant vectors. This choice of Z corresponds to non-overlapping subdomains, which are often used in DDM.

4.1.2 Bubbly Flow Problem

In the bubbly flow problem, we consider circular air bubbles in Ω filled with water, see Fig. 1(b) for the geometry. In this case, the density contrast is equal to $\epsilon = 10^3$. We impose non-homogeneous Neumann boundaries such that the resulting linear system (17) is still compatible. In contrast to the porous media flow problem, A is now singular.

The projection vectors are again based on (18), but with a significant difference that the subdomains are taken independently of the bubbles. Instead, identical square subdomains are chosen, where the number of them can be varied, see also Fig. 2(b). This means that a subdomain might cover two densities, yielding a more sophisticated situation compared to the porous media flow problem. It can be shown that the corresponding projection vectors approximate slow-varying eigenvectors corresponding to small eigenvalues, see e.g. [41]. This is even the case for bubbly flow problems with a more complex geometry, provided that a sufficient number of subdomains is taken. We omit the last column of Z in order to get a nonsingular matrix E [39].

4.2 Setup of the Experiments

We will start with a numerical experiment using standard parameters, which means that an appropriate termination criterion, exact computation of E^{-1}, and exactly computed starting vectors are used. Results for both test problems are given. Subsequently, numerical experiments will be performed with an approximation of E^{-1}, severe termination tolerances, and

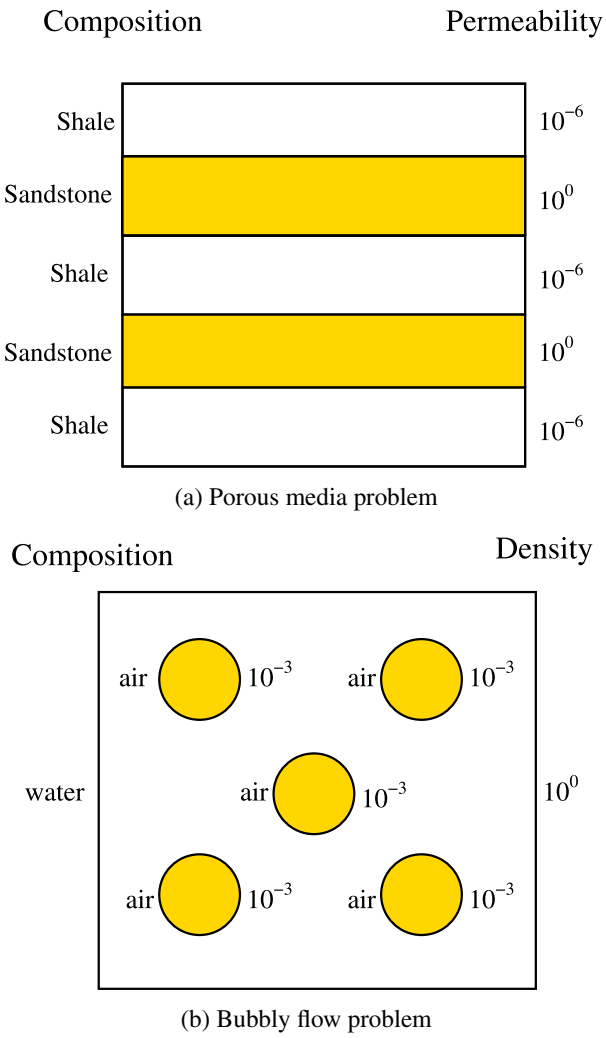
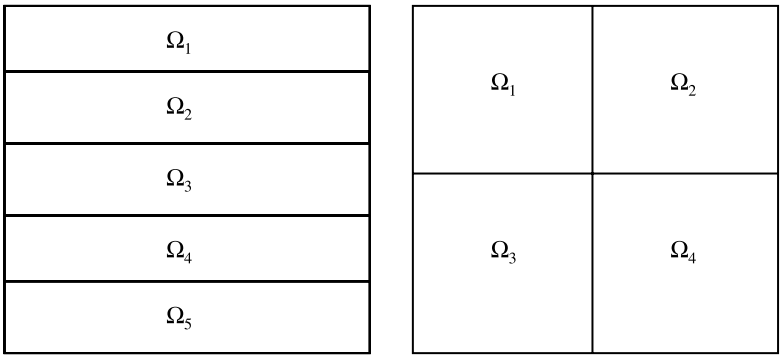


Fig. 1 Geometry of the piecewise-constant coefficient, $\mathcal{K}(\mathbf{x})$, in the two test problems

perturbed starting vectors. We restrict ourselves to the porous media flow problem in these experiments, since the results for the bubbly flows turn out to be similar, see [37].

The results for each method will be presented in two ways. First, results will be summarized in a table, presenting the number of iterations and the standard norm of the relative errors (i.e., $\|x_{\text{it}} - x\|_2/\|x\|_2$ with the iterated solution, x_{it}). Second, the results will be presented graphically. Finally, for each test case, the iterative process of each method will be terminated if the maximum allowed number of iterations (chosen to be equal to 250) is reached, or if the norm of the relative residual, $\|r_{j+1}\|_2/\|r_0\|_2$, falls below a given tolerance. As mentioned in Sect. 2.4.1, this termination criterion leads to a fair comparison of the two-level PCG methods.

Finally, we remark that the choice of parameters, Z, M and the direct solver for E^{-1}, are the same for each two-level PCG method. This allows us to compare these methods fairly. However, in practice, the two-level PCG methods come from different fields, where typical



(a) Porous media problem (b) Bubbly flow problem

Fig. 2 Geometry of subdomains Ω_j. Number of subdomains is fixed in the porous media problem, whereas it can be varied in the bubbly flow problem

Table 4 Number of required iterations for convergence of all proposed methods, for the porous media problem with 'standard' parameters

Method	$k=5$		$k=7$	
	$n=29^2$	$n=54^2$	$n=41^2$	$n=55^2$
PREC	102	174	184	222
AD	59	95	74	90
DEF1	58	94	75	90
DEF2	68	94	75	90
A-DEF1	58	95	86	103
A-DEF2	58	94	75	90
BNN	58	94	75	90
R-BNN1	58	94	75	90
R-BNN2	58	94	75	90

choices associated with these fields are made for these parameters. This is also mentioned in Sect. 2.1. A comparison of the two-level PCG methods with their typical parameters is done in [20].

4.3 Experiment using Standard Parameters

In the first numerical experiment, standard parameters are used with stopping tolerance $\delta = 10^{-10}$, an exact small matrix inverse E^{-1} and an unperturbed starting vector, $\mathcal{V}_{\text{start}}$.

4.3.1 Porous Media Problem

The results are presented in Table 4 and Fig. 3. The relative errors are approximately the same for all methods. The figure presents only one test case, since a similar behavior is seen for the other test cases. Moreover, for the sake of a better view, the results for PREC are omitted in Fig. 3. We note that the A-norm errors form a monotonically decreasing sequence as we expect from the CG theory.

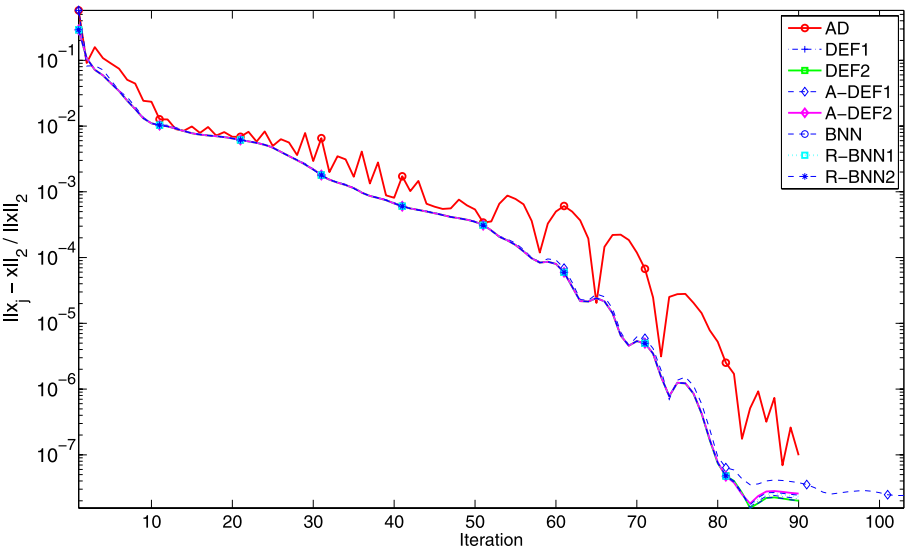


Fig. 3 Relative errors in 2-norm during the iterative process, for the porous media problem with $n = 55^2$, $k = 7$ and 'standard' parameters

From Table 4, we observe that PREC needs more iterations to converge when the number of grid points, n, or number of layers, k, is increased. This only holds partly for the two-level PCG methods. The convergence of these methods is less sensitive to the number of layers, since the number of projection vectors is chosen to be equal to the number of layers. PREC is obviously the slowest method, and the two-level PCG methods, except for A-DEF1, show approximately the same performance, which confirms the theory (cf. Theorem 3.1 and 3.3). Notice that even though AD converges comparably well as the other two-level PCG methods (except A-DEF1), it can be observed in Fig. 3 that AD shows a very erratic behavior with respect to the errors in the 2-norm. The AD errors measured in the A-norm, however, appear to be close to those of the other methods [37].

Subsequently, we present the same results in terms of computational cost. We restrict ourselves to the test case with $n = 55^2$ and $k = 7$, see Table 5. Analogous results are obtained for the other test cases. The total computational cost within the iterations is given, following the analysis carried out in Sect. 2.4.2. Due to the sparsity of Z, both Z and AZ can be stored as approximately two vectors, resulting in the fact that there is no need to perform extra matrix-vector multiplications, in addition to those required by PREC. It depends on the exact implementation of the methods (such as the storage and computation with Z, AZ and E) to determine which two-level PCG method requires the smallest computational cost. For example, if both IP, VU, CSS and PR require the same amount of computing time, then it can be deduced from Table 5 that BNN is the most expensive method, whereas AD, following by DEF1, DEF2 and R-BNN2, has the lowest computational cost per iteration.

4.3.2 Bubbly Flow Problem

The results with the bubbly flow problem can be found in Table 6 and Fig. 4. Now, we keep the number of grid points, n, constant and we vary the number of projection vectors, k.

By considering Table 6 and Fig. 4, we observe that all methods perform the same, except for PREC, AD and A-DEF1. A-DEF1 has difficulties to converge, especially for the

Table 5 Total computational cost within the iterations in terms of number of inner products ('IP'), vector updates ('VU'), coarse system solves ('CSS'), preconditioning step with M^{-1} ('PR'), for the porous media problem with $n = 55^2$, $k = 7$ and 'standard' parameters

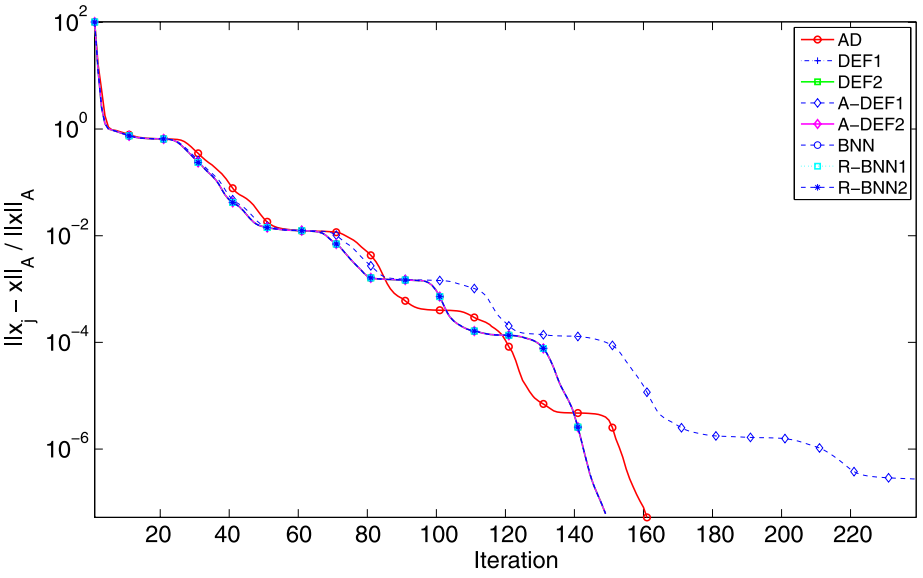
Method	IP	VU	CSS	PR
PREC	222	666	0	222
AD	270	270	90	90
DEF1	270	360	90	90
DEF2	270	360	90	90
A-DEF1	412	412	103	103
A-DEF2	450	360	180	90
BNN	540	450	180	90
R-BNN1	450	450	180	90
R-BNN2	270	360	90	90

Table 6 Number of required iterations for convergence and the 2-norm of the relative errors of all methods, for the bubbly flow problem with $n = 64^2$ and 'standard' parameters. 'NC' means no convergence within 250 iterations

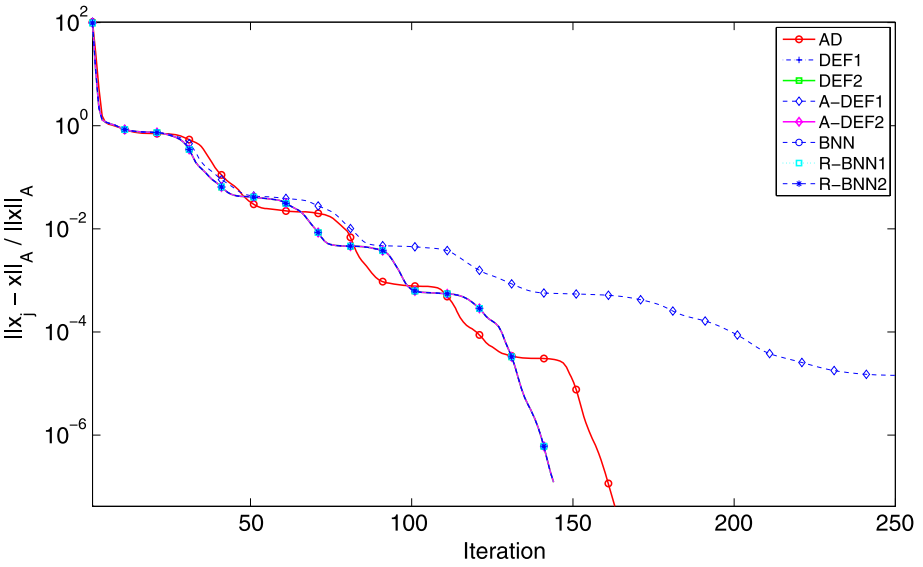
Method	$k = 2^2$		$k = 4^2$		$k = 8^2$	
	# It.	$\frac{\|x_{it}-x\|_2}{\|x\|_2}$	# It.	$\frac{\|x_{it}-x\|_2}{\|x\|_2}$	# It.	$\frac{\|x_{it}-x\|_2}{\|x\|_2}$
PREC	137	4.6×10^{-7}	137	4.6×10^{-7}	137	1.8×10^{-7}
AD	161	1.1×10^{-8}	163	8.4×10^{-9}	60	1.1×10^{-8}
DEF1	149	1.5×10^{-8}	144	3.1×10^{-8}	42	1.8×10^{-8}
DEF2	149	1.5×10^{-8}	144	3.1×10^{-8}	42	1.8×10^{-8}
A-DEF1	239	3.5×10^{-7}	NC	9.0×10^{-6}	48	1.5×10^{-9}
A-DEF2	149	1.5×10^{-8}	144	3.1×10^{-8}	42	1.1×10^{-8}
BNN	149	1.5×10^{-8}	144	3.1×10^{-8}	42	1.1×10^{-8}
R-BNN1	149	1.5×10^{-8}	144	3.1×10^{-8}	42	1.1×10^{-8}
R-BNN2	149	1.5×10^{-8}	144	3.1×10^{-8}	42	1.1×10^{-8}

cases with $k = 2^2$ and $k = 4^2$. This is not surprising, since it cannot be shown that it is an appropriate preconditioner, see Sect. 2.4.1. In addition, the number of projection vectors is apparently too low to approximate the eigenvectors corresponding to the small eigenvalues, which is the result of the presence of the bubbles. Therefore, we hardly see any improvements by comparing all two-level PCG methods to PREC in the case of $k = 2^2$ and $k = 4^2$. It is unexpected that PREC requires fewer iterations in these cases, but observe that the corresponding solution is somewhat less accurate than the others. Moreover, we remark that AD performs obviously worse, compared to the other two-level PCG methods.

The computational cost of the methods in this experiment is presented in Table 7. We restrict ourselves to the test case with $k = 8^2$, since analogous results are obtained for the other test cases. As mentioned in Sect. 4.3.1, it depends on the exact implementation of the methods to determine which two-level PCG method requires the lowest computational cost.



(a) $k = 2^2$

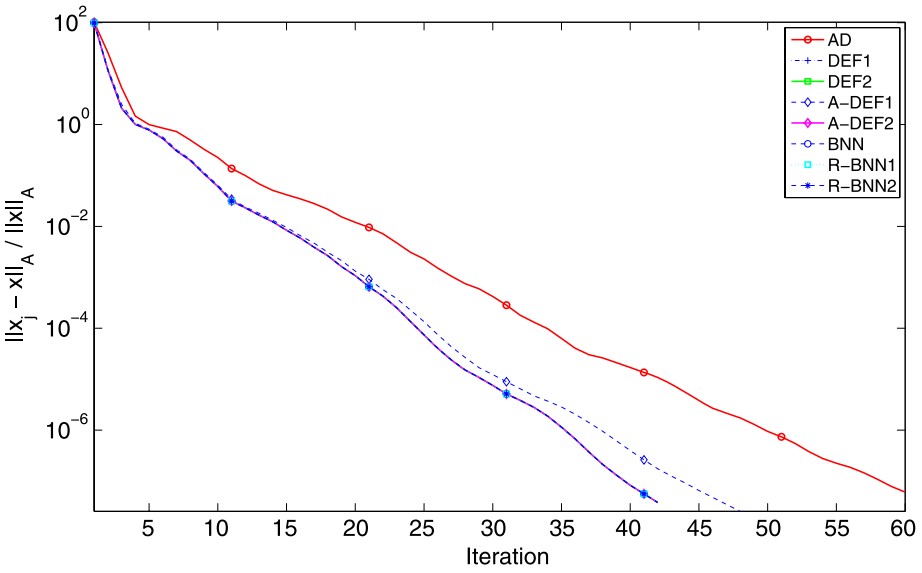


(b) $k = 4^2$

Fig. 4 Relative errors during the iterative process, for the bubbly flow problem with $n = 64^2$ and 'standard' parameters

4.4 Experiment using Inaccurate Coarse Solves

In the remaining part of this paper, we restrict ourselves to the porous media problem, since the results for the bubbly flow problems are similar, see [37].



(c) $k = 8^2$

Fig. 4 (*Continued*)

Table 7 Computational cost within the iterations in terms of number of inner products ('IP'), vector updates ('VU'), coarse system solves ('CSS') and preconditioning step with M^{-1} ('PR'), for the bubbly flow problem with $n = 64^2$, $k = 8^2$ and 'standard' parameters

Method	IP	VU	CSS	PR
PREC	137	411	0	137
AD	180	180	42	42
DEF1	126	168	42	42
DEF2	126	168	42	42
A-DEF1	192	192	48	48
A-DEF2	210	168	84	42
BNN	252	210	84	42
R-BNN1	210	210	84	42
R-BNN2	126	168	42	42

For problems with a relatively large number of projection vectors, it might be expensive to find an accurate solution of the coarse system, $Ey = v$, by a direct solver at each iteration of a two-level PCG method. Instead, only an approximate solution, $\tilde{y}$, can be determined, using, for example, approximate solvers based on SSOR or ILUT preconditioners, recursive MG methods or nested iterations, such as a standard (Krylov) iterative solver with a low accuracy. In this case, $\tilde{y}$ can be interpreted as $\widetilde{E}^{-1}v$, where $\widetilde{E}$ is an inexact matrix based on E. This justifies our next experiment, using $\widetilde{E}^{-1}$ defined as

$$\widetilde{E}^{-1} := (I + \psi R)E^{-1}(I + \psi R), \quad \psi > 0, \tag{19}$$

where $R \in \mathbb{R}^{k\times k}$ is a symmetric random matrix with elements from the interval $[-0.5, 0.5]$, see also [28, Sect. 3] for more details. Note that theories, as derived in Sect. 3.2, are not

Table 8 Number of required iterations for convergence and the 2-norm of the relative errors of all methods, for the porous media problem with parameters $n = 55^2$ and $k = 7$. A perturbed small matrix, $\widetilde{E}^{-1}$, is used with varying perturbation ψ

Method	$\psi = 10^{-12}$		$\psi = 10^{-8}$		$\psi = 10^{-4}$	
	# It.	$\frac{\|x_{it}-x\|_2}{\|x\|_2}$	# It.	$\frac{\|x_{it}-x\|_2}{\|x\|_2}$	# It.	$\frac{\|x_{it}-x\|_2}{\|x\|_2}$
PREC	222	2.6×10^{-8}	222	2.6×10^{-8}	222	2.6×10^{-8}
AD	90	1.0×10^{-7}	90	1.4×10^{-7}	92	1.2×10^{-7}
DEF1	90	2.6×10^{-6}	NC	6.8×10^{-7}	178	1.4×10^{-3}
DEF2	90	2.6×10^{-6}	NC	$1.6 \times 10^{+2}$	NC	$2.0 \times 10^{+4}$
A-DEF1	103	2.0×10^{-8}	103	2.2×10^{-8}	120	2.6×10^{-7}
A-DEF2	90	2.2×10^{-8}	90	2.6×10^{-8}	90	2.5×10^{-7}
BNN	90	2.3×10^{-8}	90	2.8×10^{-8}	90	7.1×10^{-8}
R-BNN1	90	6.8×10^{-7}	159	2.2×10^{-8}	213	6.9×10^{-5}
R-BNN2	90	2.6×10^{-6}	NC	2.6×10^{-2}	NC	$1.8 \times 10^{+2}$

valid for any $\psi > 0$, but we will see that some of those theoretical results are still confirmed for relatively large ψ. The sensitivity of the two-level PCG methods to this approximation with various values of ψ will be investigated and the results will be related to Theorem 3.4. Note that the results for PREC are not influenced by this adaptation of E^{-1}. They are only included for reference.

We remark that (19) does not reflect the way that inexact coarse solves typically enter two-level PCG methods, but it does provide us with good insights into approximate coarse solves applied to these methods. Additionally, the approximation of E^{-1} can be quantified explicitly using (19). Experiments with coarse solves that are done iteratively (i.e., nested iterations) can be found in [42]. In that paper, it has been shown that it is reasonable to apply (19), since they give similar results, as will be shown in this subsection. Moreover, it turns out that the original PCG rather than a flexible variant can still be used in these experiments, as long as the inner stopping tolerance is sufficiently small. More details about inexact Krylov subspace methods can also be found in [35].

The results of the experiment can be found in Table 8 and Fig. 5. The behavior of the A-norm errors is comparable to that of the 2-norm error. We observe that the most robust two-level PCG methods are AD, BNN and A-DEF2, since they appear to be largely insensitive to perturbations in E^{-1}. On the other hand, DEF1, DEF2, R-BNN1 and R-BNN2 are obviously the worst methods, as expected, since the zero eigenvalues of the corresponding systems become small nonzero eigenvalues due to the perturbation, ψ (cf. Sect. 3.1). In addition, it can be observed that the errors diverge or stagnate for all test cases with DEF2 and R-BNN2, whereas they remain bounded and tend to converge in the case of DEF1 and R-BNN1.

4.5 Experiment using Severe Termination Tolerances

In practice, the two-level PCG methods are sometimes compared with a 'too strict' termination criterion. Suppose a method is stopped if $\frac{\|r_j\|_2}{\|b\|_2} \leq \delta$. If δ is taken less than $\kappa(A)\mu$, where μ is the machine precision, we call this 'too strict'. Such a comparison can be unfair, since certain two-level PCG methods are sensitive to severe termination criteria, see e.g. [15]. We will investigate this in more detail, performing a numerical experiment with various values

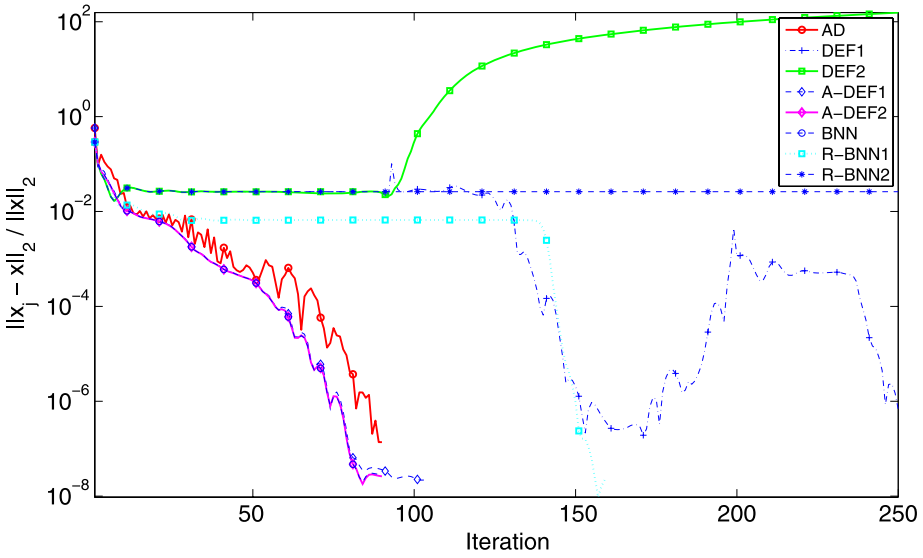


Fig. 5 Relative errors in 2-norm during the iterative process for the porous media problem with $n = 55^2$, $k = 7$ and $\widetilde{E}^{-1}$, where a perturbation $\psi = 10^{-8}$ is taken

Table 9 Number of required iterations for convergence and the 2-norm of the relative errors of all methods, for the porous media problem with parameters $n = 55^2$ and $k = 7$. Various termination tolerances, δ, are tested

Method	$\delta = 10^{-8}$		$\delta = 10^{-12}$		$\delta = 10^{-16}$	
	# It.	$\frac{\|x_{it}-x\|_2}{\|x\|_2}$	# It.	$\frac{\|x_{it}-x\|_2}{\|x\|_2}$	# It.	$\frac{\|x_{it}-x\|_2}{\|x\|_2}$
PREC	134	3.7×10^{-1}	>250	2.4×10^{-8}	>250	2.4×10^{-8}
AD	80	5.2×10^{-6}	123	2.4×10^{-8}	139	2.4×10^{-8}
DEF1	80	7.5×10^{-8}	121	2.0×10^{-8}	NC	4.4×10^{-7}
DEF2	80	7.5×10^{-8}	144	1.9×10^{-8}	NC	$6.6 \times 10^{+1}$
A-DEF1	80	9.4×10^{-8}	121	2.5×10^{-8}	190	2.5×10^{-8}
A-DEF2	80	7.7×10^{-8}	121	2.5×10^{-8}	138	2.5×10^{-8}
BNN	80	7.7×10^{-8}	121	2.4×10^{-9}	138	2.4×10^{-8}
R-BNN1	80	7.6×10^{-8}	121	2.3×10^{-8}	NC	2.3×10^{-8}
R-BNN2	80	7.5×10^{-8}	121	1.9×10^{-8}	NC	1.9×10^{-8}

of the tolerance, δ. Note that for a relatively small δ, this may lead to a 'too severe' termination criterion with respect to machine precision. However, the aims of this experiment are to test the sensitivity of the two-level PCG methods to δ and to investigate the maximum accuracy that can be reached, rather than to perform realistic experiments.

The results of the experiment are presented in Table 9 and Fig. 6. It can be seen that all methods perform well, even in the case of a relatively strict termination criterion (i.e., $\delta = 10^{-12}$). PREC also converges in all cases, but not within 250 iterations. Note, moreover, that it does not give an accurate solution if δ is chosen too large, see also [45]. For $\delta <$

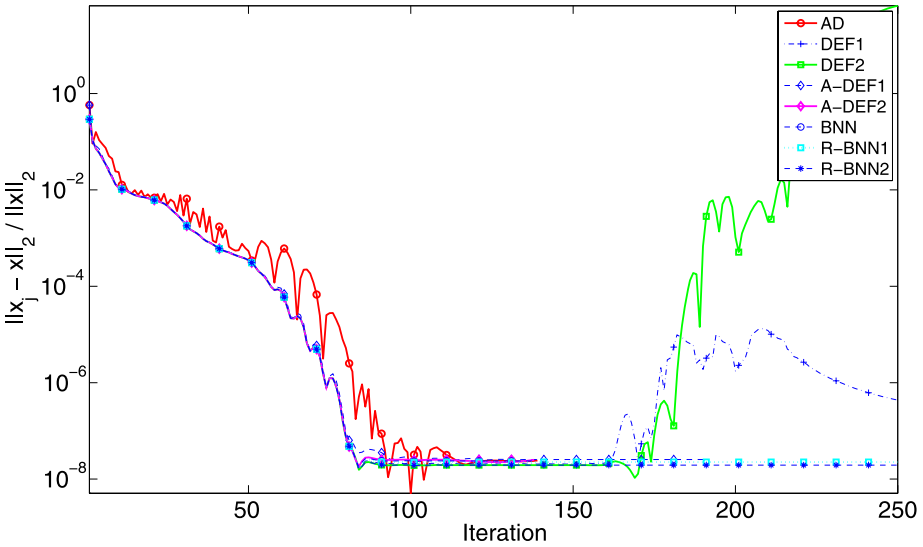


Fig. 6 Relative 2-norm errors during the iterative process for the porous media problem with $n = 55^2$, $k = 7$ and termination tolerance $\delta = 10^{-16}$

10^{-12}, DEF1, DEF2, R-BNN1 and R-BNN2, show difficulties, since they do not converge appropriately and may even diverge. This is in contrast to PREC, AD, BNN, A-DEF1 and A-DEF2, which give good convergence results for $\delta = 10^{-16}$. Therefore, these two-level PCG methods can be characterized as robust methods with respect to severe termination criteria.

4.6 Experiment using Perturbed Starting Vectors

In Sect. 3.2, it has been proven that BNN with $\mathcal{V}_{\text{start}} = Qb + P^T \bar{x}$ gives exactly the same iterates as DEF2, A-DEF2, R-BNN1 and R-BNN2, in exact arithmetic. In this case, the resulting operators are well-defined and they should perform appropriately. In our next experiment, we will perturb $\mathcal{V}_{\text{start}}$ in DEF2, A-DEF2, R-BNN1 and R-BNN2, and examine whether this influences the convergence results. The motivation of this experiment is the same as for the experiment carried out in Sect. 4.4: for relatively large problems, it can be complicated to determine $\mathcal{V}_{\text{start}}$ accurately, due to, for example, the inaccurate computation of coarse solves. It is important to note that if we use approximate starting vectors, then there is no longer any equivalence between BNN and its reduced methods, as provided in the results of Sect. 3.2. In this case, it is interesting to see how these methods perform in practice.

The perturbed $\mathcal{V}_{\text{start}}$, denoted as $\mathcal{W}_{\text{start}}$, will be defined as a component-wise multiplication of a random vector and $\mathcal{V}_{\text{start}}$, i.e., each element of $\mathcal{W}_{\text{start}}$ is defined as

$$(\mathcal{W}_{\text{start}})_i := (1 + \gamma(v_0)_i)\,(\mathcal{V}_{\text{start}})_i, \quad i = 1, 2, \ldots, n,$$

where $\gamma \geq 0$ gives control over the accuracy of the starting vector, and vector v_0 is a random vector with elements from the interval $[-0.5, 0.5]$, taken to give each element of $\mathcal{V}_{\text{start}}$ a different perturbation. As in the experiment performed in Sect. 4.4, the choice of $\mathcal{W}_{\text{start}}$ does not reflect the way in which starting vectors are perturbed in practice, but it provides us with some valuable insights where the perturbation can be quantified in an easy way.

Table 10 Number of required iterations for convergence and the 2-norm of the relative errors of some methods, for the porous media problem with parameters $n = 55^2$, $k = 7$ and perturbed starting vectors. An asterisk (*) means that an extra uniqueness step is applied in that test case

Method	$\gamma = 10^{-10}$		$\gamma = 10^{-5}$		$\gamma = 10^{0}$	
	# It.	$\frac{\|x_{it}-x\|_2}{\|x\|_2}$	# It.	$\frac{\|x_{it}-x\|_2}{\|x\|_2}$	# It.	$\frac{\|x_{it}-x\|_2}{\|x\|_2}$
DEF2	90	2.2×10^{-8}	NC	$2.1 \times 10^{+11}$	NC	$3.5 \times 10^{+18}$
A-DEF2	90	2.5×10^{-8}	90	2.5×10^{-8}	90	2.4×10^{-8}
R-BNN1	90	2.5×10^{-8}	NC	2.5×10^{-8}*	NC	1.3×10^{-5}*
R-BNN2	90	2.0×10^{-8}	NC	2.9×10^{-6}*	NC	2.5×10^{-1}*

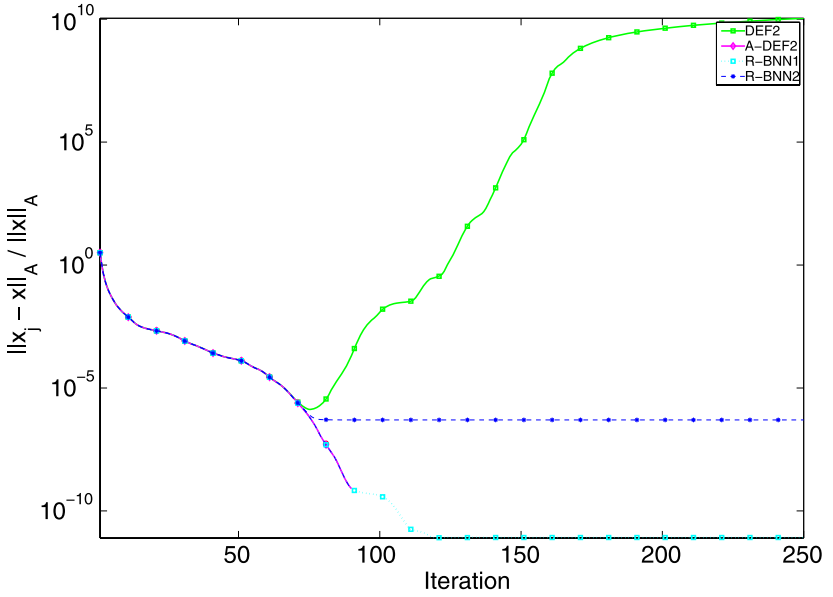


Fig. 7 Relative errors in A-norm during the iterative process for the porous media problem with $n = 55^2, k = 7^2$, and perturbed starting vectors with $\gamma = 10^{-5}$. The plot of the relative errors in 2-norm is omitted, since the two plots are approximately the same

Furthermore, note that if DEF2, R-BNN1 or R-BNN2 converge using $\mathcal{W}_{\text{start}}$, then we may obtain a non-unique solution, since the corresponding operator is singular. Therefore, as in the case of DEF1, we should apply the 'uniqueness' step, as mentioned in Sect. 2.3.2, at the end of the iteration process. Note that this procedure is not required for A-DEF2, because this method corresponds to a nonsingular operator.

We perform the numerical experiment using $\mathcal{W}_{\text{start}}$ for different γ. The results can be found in Table 10 and Fig. 7. Here, we use asterisks to stress that an extra uniqueness step is applied in the specific method. Moreover, notice that PREC, AD, DEF1 and BNN are not included in this experiment, since they apply an arbitrary vector, $\mathcal{V}_{\text{start}} = \bar{x}$, by definition.

From the results, it can be noticed that all involved methods converge appropriately for $\gamma = 10^{-10}$. For $\gamma \geq 10^{-5}$, DEF2, R-BNN1 and R-BNN2 fail to converge, although R-BNN1 seems already to be converged and the current stopping criterion is apparently unreliable for

this method in this experiment. The most robust method is, obviously, A-DEF2. This method seems to be completely insensitive to the perturbation, γ. This experiment has shown that the 'reduced' variants of BNN have different robustness properties with respect to perturbations in starting vectors.

4.7 Summary and Discussion

The theoretical results given in Sect. 3 only hold in exact arithmetic and under the assumptions required to prove them. However, from a numerical point of view, we have observed that some of these assumptions are necessary, whereas others are only sufficient for certain two-level PCG methods. The numerical results confirm the theoretical fact that all two-level PCG methods perform approximately the same, although A-DEF1 showed problems in some test cases. This is understood by the fact that A-DEF1 corresponds to a non-SPSD operator, as also discussed in Sect. 2.4.1.

If the dimensions of the matrix E become large, it is favorable to solve the corresponding systems iteratively, with a low accuracy. In this case, we have seen that DEF1, DEF2, R-BNN1 and R-BNN2 showed difficulties in convergence. It can be observed that the errors during the iterative process of DEF2 explode, whereas DEF1 converges slowly to the solution, but in an erratic way. The most robust methods turn out to be AD, BNN, A-DEF1 and A-DEF2.

If A is ill-conditioned and the tolerance of the termination criterion, chosen by the user, becomes too severe, it is advantageous if the two-level PCG method still works appropriately. However, we have observed that DEF1, DEF2, A-DEF1, R-BNN1 and R-BNN2 cannot deal with too strict tolerances. This is in contrast to AD, BNN, A-DEF2, which remain robust in all test cases.

In theory, BNN gives the same iterates as DEF2, A-DEF2, R-BNN1 and R-BNN2, for certain starting vectors. In addition to the fact that these 'reduced' variants, except A-DEF2, are not able to deal with inaccurate coarse solves, some of them are also sensitive to perturbations of the starting vector. In contrast to the other methods, A-DEF2 appears to be independent of these perturbations. This can be of great importance, if one uses multigrid-like subdomains, where the number of subdomains, k, is very large, and the starting vector cannot be obtained accurately. We recall that the robustness of the two-level PCG methods is difficult to analyze theoretically, so that the analysis of this robustness issue is especially based on numerical experiments.

In the numerical experiments, we have observed that several methods showed divergence, stagnation or erratic behavior of the errors during the iterative process. This may be caused by the fact that the residuals gradually lose orthogonality with respect to the columns of Z, see also [34]. It can easily be shown that

$$Z^T r_j = \mathbf{0}, \quad j = 1, 2, \ldots, \tag{20}$$

should hold for DEF1, DEF2, A-DEF2, R-BNN1 and R-BNN2. However, it appeared that (20) is not always satisfied in the experiments. A remedy to recover this orthogonality in the badly-converging methods is described in, e.g., [34]. If we define the 'reorthogonalization' matrix W as

$$W := I - Z(Z^T Z)^{-1} Z^T, \tag{21}$$

then W is orthogonal to Z, i.e.,

$$Z^T W = Z^T - Z^T Z(Z^T Z)^{-1} Z^T = \mathbf{0}. \tag{22}$$

Now, orthogonality of the residuals, r_j, can be preserved, by premultiplying r_j by W right after r_j is computed in the algorithm:

$$r_j := W r_j, \quad j = 1, 2, \ldots. \tag{23}$$

As a consequence, these adapted residuals satisfy (20), due to (22).[1]

In the numerical experiments of [37, Sect. 4.6], we have shown that the adapted versions of the methods, including the reorthogonalization strategy, converge better in terms of the residuals. Unfortunately, it appeared that accurate solutions could not be obtained using this approach. To preserve the relation, $r_j = b - Ax_j$, each iterate x_j should be adapted via

$$x_j := x_j - A^{-1}Z(Z^TZ)^{-1}Z^Tr_j, \quad j = 1, 2, \ldots. \tag{24}$$

However, it is clear that (24) is not useful to apply in practice due to the presence of A^{-1} in that expression. Hence, the reorthogonalization strategy cannot be used in practice.

5 Conclusions

Several abstract two-level PCG methods, listed in Table 1, have been considered, coming from the fields of deflation, domain decomposition and multigrid. A comparison of these methods, whose parameters can be arbitrary, has been carried out by investigating their theoretical and numerical aspects.

Theoretically, DEF1 appears to be the best method [28–30]. We have seen that all two-level PCG methods, except for PREC and AD, have comparable eigenvalue distributions. Two classes of two-level PCG methods could be made, each having the same spectral properties. The first class consists of DEF1, DEF2, R-BNN1 and R-BNN2, and the second class includes BNN, A-DEF1 and A-DEF2. Although the differences are surprisingly marginal and, therefore, similar convergence behaviors are expected, it has been shown that the associated spectrum of the methods of the first class is possibly more favorable than those of the second class.

In numerical experiments with realistic termination criteria and relatively small perturbations in the starting vector and coarse solves, it has been observed that all two-level PCG methods always converge faster than PREC. More importantly, all two-level PCG methods show approximately the same convergence behavior, although the residuals of AD sometimes have a nonmonotonical convergence behavior. Both DEF1 and DEF2 are sensitive to sufficiently large perturbations in the coarse solves or too strict termination criterion. In contrast to DEF1, DEF2 also has the drawbacks that it cannot deal with perturbed starting vectors and that the method diverges when the convergence deteriorates. The errors are usually bounded in DEF1, if this method does not converge.

It has been shown that, for certain starting vectors, the expensive operator of BNN can be reduced to simpler and cheaper operators, which are used in DEF2, A-DEF2, R-BNN1 and R-BNN2. Hence, some two-level PCG methods of the two spectral classes are mathematically equivalent in exact arithmetic. However, these reduced variants, except for A-DEF2,

[1] Note that (20) is not valid for AD, A-DEF1 and BNN. In the case of AD and BNN, this is not a problem, because they appear extremely robust in most test cases. This is in contrast to A-DEF1, which is not robust in several test cases, since it is not an appropriate preconditioner, see Sect. 2.4.1. The non-robustness of this projector cannot be resolved using the reorthogonalization strategy. Moreover, note that the reorthogonalization operator (23) is relatively cheap, provided that Z is sparse.

do not appear to be robust in the numerical experiments, when applying inaccurate coarse solves, strict stopping tolerances or perturbed starting vectors. In fact, one should realize that the reduced variants of BNN, except A-DEF2, are not as robust as DEF1 or DEF2.

Finally, by examining all theoretical and numerical aspects, we conclude that BNN and A-DEF2 are the best two-level PCG methods in the sense of robustness. However, BNN is slightly more expensive to use. Hence, A-DEF2 seems to be the best and most robust method, considering the theory, numerical experiments and the computational cost.

Acknowledgements We would like to thank Scott MacLachlan for the helpful discussions and comments on the paper and for his thorough proof-reading of this manuscript. We also like to thank the referees for their remarks, which improve the paper considerably.

References

1. Bastian, P., Hackbusch, W., Wittum, G.: Additive and multiplicative multigrid: a comparison. Computing **60**, 345–364 (1998)
2. Bramble, J.H., Pasciak, J.E., Schatz, A.H.: The construction of preconditioners for elliptic problems by substructuring. Math. Comput. **47**, 103–134 (1986)
3. Brandt, A.: Multi-level adaptive solutions to boundary-value problems. Math. Comput. **31**, 333–390 (1977)
4. Carpentieri, B., Giraud, L., Gratton, S.: Additive and multiplicative two-level spectral preconditioning for general linear systems. SIAM J. Sci. Comput. **29**, 1593–1612 (2007)
5. Dryja, M.: An additive Schwarz algorithm for two- and three-dimensional finite element elliptic problems. In: Domain Decomposition Methods, pp. 168–172. SIAM, Philadelphia (1989)
6. Dryja, M., Widlund, O.B.: Towards a unified theory of domain decomposition algorithms for elliptic problems. In: Third International Symposium on DDM for PDEs, pp. 3–21. SIAM, Philadelphia (1990)
7. Dryja, M., Widlund, O.B.: Schwarz methods of Neumann-Neumann type for three-dimensional elliptic finite element problems. Commun. Pure Appl. Math. **48**, 121–155 (1995)
8. Eiermann, M., Ernst, O.G., Schneider, O.: Analysis of acceleration strategies for restarted minimal residual methods. J. Comput. Appl. Math. **123**, 261–292 (2000)
9. Erlangga, Y.A., Nabben, R.: Multilevel projection-based nested Krylov iterations for boundary value problems. SIAM J. Sci. Comput. **30**, 1572–1595 (2008)
10. Faber, V., Manteuffel, T.: Necessary and sufficient conditions for the existence of a conjugate gradient method. SIAM J. Numer. Anal. **21**, 352–362 (1984)
11. Frank, J., Vuik, C.: On the construction of deflation-based preconditioners. SIAM J. Sci. Comput. **23**, 442–462 (2001)
12. De Gersem, H., Hameyer, K.: A deflated iterative solver for magnetostatic finite element models with large differences in permeability. Eur. Phys. J. Appl. Phys. **13**, 45–49 (2000)
13. Giraud, L., Ruiz, D., Touhami, A.: A comparative study of iterative solvers exploiting spectral information for SPD systems. SIAM J. Sci. Comput. **27**, 1760–1786 (2006)
14. Golub, G.H., Van Loan, C.F.: Matrix Computations, 3rd edn. The John Hopkins University Press, Baltimore (1996)
15. Graham, I.G., Scheichl, R.: Robust domain decomposition algorithms for multiscale PDEs. Numer. Methods Partial Differ. Equ. **23**, 859–878 (2007)
16. Hackbusch, W.: Multigrid Methods and Applications. Springer, Berlin (1985)
17. Hestenes, M.R., Stiefel, E.: Methods of conjugate gradients for solving linear systems. J. Res. Natl. Bur. Stand. **49**, 409–436 (1952)
18. Kaasschieter, E.F.: Preconditioned conjugate gradients for solving singular systems. J. Comput. Appl. Math. **24**, 265–275 (1988)
19. Kolotilina, L.Y.: Twofold deflation preconditioning of linear algebraic systems, I: theory. J. Math. Sci. **89**, 1652–1689 (1998)
20. MacLachlan, S.P., Tang, J.M., Vuik, C.: Fast and robust solvers for pressure correction in bubbly flow problems. J. Comput. Phys. **227**, 9742–9761 (2008)

21. Mandel, J.: Balancing domain decomposition. Commun. Appl. Numer. Methods **9**, 233–241 (1993)
22. Mandel, J.: Hybrid domain decomposition with unstructured subdomains. In: Proceedings of the Sixth International Symposium on DDM, Como, Italy, 1992. Contemp. Math., 157, pp. 103–112. AMS, Providence (1994)
23. Mandel, J., Brezina, M.: Balancing domain decomposition for problems with large jumps in coefficients. Math. Comput. **216**, 1387–1401 (1996)
24. Mansfield, L.: On the conjugate gradient solution of the Schur complement system obtained from domain decomposition. SIAM J. Numer. Anal. **27**, 1612–1620 (1990)
25. Mansfield, L.: Damped Jacobi preconditioning and coarse grid deflation for conjugate gradient iteration on parallel computers. SIAM J. Sci. Stat. Comput. **12**, 1314–1323 (1991)
26. Meijerink, J.A., Van der Vorst, H.A.: An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix. Math. Comput. **31**, 148–162 (1977)
27. Morgan, R.B.: A restarted GMRES method augmented with eigenvectors. SIAM J. Matrix Anal. Appl. **16**, 1154–1171 (1995)
28. Nabben, R., Vuik, C.: A comparison of deflation and coarse grid correction applied to porous media flow. SIAM J. Numer. Anal. **42**, 1631–1647 (2004)
29. Nabben, R., Vuik, C.: A comparison of deflation and the balancing preconditioner. SIAM J. Sci. Comput. **27**, 1742–1759 (2006)
30. Nabben, R., Vuik, C.: A comparison of abstract versions of deflation, balancing and additive coarse grid correction preconditioners. Numer. Linear Algebra Appl. **15**, 355–372 (2008)
31. Nicolaides, R.A.: Deflation of Conjugate Gradients with applications to boundary value problems. SIAM J. Matrix Anal. Appl. **24**, 355–365 (1987)
32. Padiy, A., Axelsson, O., Polman, B.: Generalized augmented matrix preconditioning approach and its application to iterative solution of ill-conditioned algebraic systems. SIAM J. Matrix Anal. Appl. **22**, 793–818 (2000)
33. Pavarino, L.F., Widlund, O.B.: Balancing Neumann-Neumann methods for incompressible Stokes equations. Commun. Pure Appl. Math. **55**, 302–335 (2002)
34. Saad, Y., Yeung, M., Erhel, J., Guyomarc'h, F.: A deflated version of the conjugate gradient algorithm. SIAM J. Sci. Comput. **21**, 1909–1926 (2000)
35. Simoncini, V., Szyld, D.B.: Recent computational developments in Krylov subspace methods for linear systems. Numer. Linear Algebra Appl. **14**, 1–59 (2007)
36. Smith, B., Bjørstad, P., Gropp, W.: Domain Decomposition. Cambridge University Press, Cambridge (1996)
37. Tang, J.M., Nabben, R., Vuik, C., Erlangga, Y.A.: Theoretical and numerical comparison of various projection methods derived from deflation, domain decomposition and multigrid methods. Delft University of Technology, DIAM Report 07-04, ISSN 1389-6520 (2007)
38. Tang, J.M., MacLachlan, S.P., Nabben, R., Vuik, C.: A comparison of two-level preconditioners based on multigrid and deflation. Submitted
39. Tang, J.M., Vuik, C.: On deflation and singular symmetric positive semi-definite matrices. J. Comput. Appl. Math. **206**, 603–614 (2007)
40. Tang, J.M., Vuik, C.: Efficient deflation methods applied to 3-D bubbly flow problems. Electron. Trans. Numer. Anal. **26**, 330–349 (2007)
41. Tang, J.M., Vuik, C.: New variants of deflation techniques for bubbly flow problems. J. Numer. Anal. Ind. Appl. Math. **2**, 227–249 (2007)
42. Tang, J.M., Vuik, C.: Fast deflation methods with applications to two-phase flows. Int. J. Mult. Comput. Eng. **6**, 13–24 (2008)
43. Toselli, A., Widlund, O.B.: Domain Decomposition: Algorithms and Theory. Comput. Math., vol. 34. Springer, Berlin (2005)
44. Trottenberg, U., Oosterlee, C.W., Schüller, A.: Multigrid. Academic Press, London (2001)
45. Vuik, C., Segal, A., Meijerink, J.A.: An efficient preconditioned CG method for the solution of a class of layered problems with extreme contrasts in the coefficients. J. Comput. Phys. **152**, 385–403 (1999)
46. Vuik, C., Nabben, R., Tang, J.M.: Deflation acceleration for domain decomposition preconditioners. In: Proceedings of the 8th European Multigrid Conference on Multigrid, Multilevel and Multiscale Methods, The Hague, The Netherlands, September 27–30, 2005
47. Wesseling, P.: An Introduction to Multigrid Methods. Wiley, New York (1992). Corrected reprint: Edwards, Philadelphia (2004)
48. Xu, J.: Iterative methods by space decomposition and subspace correction. SIAM Rev. **34**, 581–613 (1992)