 CMES, vol.53, no.1, pp.23-45, 2009

Solution Methods for Nonsymmetric Linear Systems with Large off-Diagonal Elements and Discontinuous Coefficients

Dan Gordon[1] **and Rachel Gordon**[2]

Abstract: Linear systems with very large off-diagonal elements and discontinuous coefficients (LODC systems) arise in some modeling cases, such as those involving heterogeneous media. Such problems are usually solved by domain decomposition methods, but these can be difficult to implement on unstructured grids or when the boundaries between subdomains have a complicated geometry. Gordon and Gordon have shown that Björck and Elfving's (sequential) CGMN algorithm and their own block-parallel CARP-CG are very robust and efficient on strongly convection dominated cases (but without discontinuous coefficients). They have also shown that scaling the equations by dividing each equation by the L_2-norm of its coefficients, called "geometric row scaling" (GRS), improves the convergence properties of Bi-CGSTAB and GMRES on nonsymmetric systems with discontinuous coefficients, provided the convection terms are only small to moderate. Given a system $Ax = b$, it is shown that if C is obtained from A by applying GRS, then the diagonal elements of CC^T are larger than the off-diagonal ones, so the normal equations system is manageable. These two operations are inherent in the Kaczmarz algorithm, and hence also in CGMN and CARP-CG (which are CG-accelerations of Kaczmarz). It is shown that these two methods are also very effective on systems with discontinuous coefficients derived from strongly convection dominated elliptic PDEs. CGNR and CGNE also benefit greatly from this approach, but they are much less efficient.

Keywords: CARP, CARP-CG, CGMN, convection-diffusion, convection dominated, discontinuous coefficients, domain decomposition, geometric scaling, GRS, large off-diagonal elements, linear equations, LODC systems, nonsymmetric systems, parallel processing, partial differential equations.

[1] Corresponding author. Dept. of Computer Science, University of Haifa, Haifa 31905, Israel. gordon@cs.haifa.ac.il

[2] Dept. of Aerospace Engineering, The Technion–Israel Inst. of Technology, Haifa 32000, Israel. rgordon@tx.technion.ac.il

1 Introduction

Many physical phenomena are modeled by partial differential equations (PDEs) which describe the relations between one or more scalar or vector fields and the surrounding media. When boundary conditions are prescribed, a common approach to achieving a numerical solution is to impose a grid and discretize the equations, thus getting a system of linear equations. In some cases, this approach yields a system of equations with very large differences between coefficients of the equations. Examples of such systems arise in modeling flow through heterogeneous media with widely-varying permeability, oil reservoir simulation, electromagnetics and semiconductor modeling, and geomechanics. Such systems, called systems with "discontinuous coefficients", are often handled by the domain decomposition (DD) approach, in which the domain is partitioned into subdomains, with subdomain boundaries conforming to the boundaries between the different media. For some references on DD, see [Glowinski and Wheeler (1988); Smith, Bjørstad, and Gropp (1996); Quarteroni and Valli (1999); Rice, Tsompanopoulou, and Vavalis (2000)]. However, DD may be difficult to implement on unstructured grids or when the boundaries between subdomains have a complicated geometry.

Another problematic feature of some systems is the occurrence of very large off-diagonal elements. This problem occurs in several situations, such as convection-diffusion equations with a large convection, the Helmholtz equation with large wave numbers, circuit simulation, and other cases. Current approaches to this problem consist of scalings, reordering schemes, and problem-specific preconditioners. For some related literature, see [Benzi, Szyld, and van Duin (1999); Duff and Koster (2001); Benzi (2002); Saad (2005); Schend, Röllin, and Gupta (2004)]. To complicate matters further, some problems exhibit both large off-diagonal elements and discontinuous coefficients; this is the topic of the present research.

The above problems are examples of the difficulties that often occur in computational mechanics, but they are by no means unique. Such problems are very often nonlinear in nature; some of them require the solution of nonlinear equations, whereas others are nonlinear optimization problems. Significant progress on the latter class of problems was achieved by Liu and Atluri (2008) with the fictitious time integration method (FTIM). FTIM has also been used successfully to solve systems of ill-posed linear equations arising from the Fredholm integral equation; see [Liu and Atluri (2009)]. This paper also demonstrates that the fictitious time acts as a regularization parameter, with a better effect than alternative filters. The latter paper also demonstrates the robustness of FTIM in the presence of noise. See also [Liu, Yeih, and Atluri (2009)] for another approach to solving ill-conditioned linear systems.

Takei, Yoshimura, and Kanayama (2008, 2009) used DD in a hierarchical mode for the parallel solution of electromagnetic field problems. Jönsthövel, van Gijzen, Vuik, Kasbergen, and Scarpas (2009) use a deflation based preconditioner for the conjugate gradient method (CG) to tackle a problem of composite materials in which the stiffness matrix has a large variance in the size of the coefficients. However, in this problem, the matrix is symmetric and positive definite, whereas the problems that we will be concerned with involve nonsymmetric matrices.

In this paper we extend our previous results [Gordon and Gordon (2008, 2009a,b)] to nonsymmetric linear systems with very large off-diagonal elements *and* discontinuous coefficients; we will call such systems LODC systems. Specifically, our test cases will consist of strongly convection dominated elliptic PDEs in heterogeneous media. Given a system $Ax = b$, we show that if C is obtained from A by applying the GRS scaling method of [Gordon and Gordon (2009b)], then the diagonal elements of CC^T are larger than the off-diagonal ones. This fact makes the normal equations manageable and it explains the efficient behavior of CGMN [Björck and Elfving (1979); Gordon and Gordon (2008)] and its block-parallel equivalent CARP-CG [Gordon and Gordon (2009a)] on LODC systems..

Three nonsymmetric convection-diffusion problems with discontinuous coefficients are considered: two from [Gerardo-Giorda, Tallec, and Nataf (2004)], and one from [Erlangga, Vuik, and Oosterlee (2004)], to which we added convection terms of varying sizes to make it nonsymmetric. The following algorithms were tested: Bi-CGSTAB [van der Vorst (1992)], GMRES [Saad and Schultz (1986)], CGNR, and CGMN. The first three were tested with and without GRS, and the first two were also tested with and without the ILU(0) preconditioner. All these algorithms are easily parallelized, but we did not run multi-processor experiments in this study, so runtime experiments were not done with CARP-CG. It it was found that in all cases when Bi-CGSTAB and GMRES run into difficulties, CGMN produced excellent results both in terms of robustness and efficiency. CGNR with GRS was also very robust, but much less efficient than CGMN.

The rest of this paper is organized as follows. §2 summarizes some previous related work and §3 presents the relevant mathematical background. §4 describes the setup of the numerical experiments, and the results are detailed in sections 5–7. §8 summarizes the results and outlines some future research directions.

2 Previous related work

2.1 CARP, CGMN, and CARP-CG

Kaczmarz (1937) introduced a row projection algorithm (KACZ) for linear systems which operates as follows: starting from an arbitrary initial point, KACZ

successively projects the current iterate onto one of the hyperplanes defined by the equations. KACZ will be detailed in the next section. A sequence of projections on all the hyperplanes will be called a KACZ sweep. This mode of operation makes KACZ inherently sequential, in contrast to the Cimmino algorithm [Cimmino (1938)], which is inherently parallel. KACZ can be parallelized in a block-sequential mode by partitioning the system into blocks of independent equations, i.e., no two equations in a block share a variable, so the row projections within a block can be done in parallel. However, such partitioning may be difficult to implement with unstructured grids.

In a block version of KACZ, introduced by Elfving (1980), row projections are replaced by projections onto affine subspaces defined by blocks of equations. CG accelerations and parallelizations of block KACZ and block Cimmino were examined by Bramley and Sameh (1992), and CG acceleration of block Cimmino was also considered by Arioli, Duff, Noailles, and Ruiz (1992) and Arioli, Duff, Ruiz, and Sadkane (1995).

In [Gordon and Gordon (2005)], we introduced a different approach to parallelizing KACZ, called CARP (component-averaged row projections). CARP operates in block-parallel mode as follows: the equations of a given linear system are partitioned into blocks (which may overlap), and then the following two operations are iterated until convergence: 1. The blocks are processed in parallel by performing KACZ sweep(s) in each block; 2. each variable belonging to two or more blocks is replaced by the average of its values in the separate blocks. It was shown that CARP is equivalent to KACZ in some superspace with cyclic relaxation parameters. This guarantees the theoretical convergence of CARP, provided the projections are performed in entire identical sweeps. CARP turned out to be very robust on linear systems derived from strongly convection dominated elliptic PDEs, but, being a linear method, its efficiency was lacking. The advantage of CARP is that it does not require blocks of independent equations.

Björck and Elfving (1979) introduced several acceleration schemes for row projection methods. One of those algorithms, called CGMN, applied CG acceleration to KACZ. This was done by running KACZ in a forward and backward sweep, thereby obtaining a symmetric iteration matrix, to which CG can be applied. Gordon and Gordon (2008) have shown that CGMN is very robust and efficient on linear systems derived from strongly convection dominated elliptic PDEs. The CGMN concept was extended in [Gordon and Gordon (2009a)] to KACZ with cyclic relaxation, and since CARP is KACZ with cyclic relaxation in some superspace, this enabled the CG acceleration of CARP, called CARP-CG. CARP-CG is as robust as CARP on problems with large convection terms, but significantly more efficient. Note that CARP-CG and CGMN are identical on a single processor.

2.2 Discontinuous coefficients

We have shown in previous work [Gordon and Gordon (2009b)] that a simple preconditioning technique, called *geometric row scaling*, can be applied to nonsymmetric linear systems with discontinuous coefficients in order to improve the convergence properties of algorithms applied to the system. For $p \geq 1$, geometric row scaling with the L_p-norm, which we denote by GRS(p), consists of dividing each equation by the L_p-norm of its vector of coefficients. The results of [Gordon and Gordon (2009b)] were obtained with GRS(2), but GRS(1) yielded similar results. For convenience, we shall use GRS to refer to GRS(2). GRS was tested with restarted GMRES and Bi-CGSTAB, both of them with and without ILU(0).

The effect of GRS is as follows: (a) When the tested algorithm/preconditioner combination converges, GRS speeds up the convergence time. (b) In many cases, when the tested method stagnates or diverges on the original system, it either converges on the scaled system, or achieves some reasonable practical convergence goal. The problems in [Gordon and Gordon (2009b)] had a very large concentration of eigenvalue around the origin, and GRS improved the distribution by moving many eigenvalues away from the origin.

Row and/or column scalings are well known techniques, and their usefulness for discontinuous coefficients has been noted before; see, for example, [van der Sluis (1969); Widlund (1971); Graham and Hagger (1999); Duff and van der Vorst (1998); Gambolati, Pini, and Ferronato (2003)]. However, scalings are most often double-sided in order to preserve symmetry, and the general usefulness of GRS for different algorithm/preconditioner combinations operating on a variety of linear systems with discontinuous coefficients had not been studied before. Furthermore, as noted in [Meurant (1999, §2.8 & §8.2)], scaling is not necessarily useful for all problems.

However, GRS has its limits. It was shown in [Gordon and Gordon (2009b)] that the effectiveness of GRS on Bi-CGSTAB and GMRES is limited by the size of the convection terms: when these were increased beyond a moderate size, GRS became progressively less effective. CGMN and CARP-CG also converged on these problems, but they were not as efficient as Bi-CGSTAB and GMRES with GRS on problems with small to moderate convection terms.

3 Mathematical background

Throughout the rest of the paper, we assume that all vectors are column vectors, and we use the following notation: $\langle p, q \rangle$ denotes the dot product of two vectors p and q, which is also $p^T q$. Given a vector $x = (x_1, \ldots, x_n)^T \in \mathbb{R}^n$, we denote its L_p-norm by $\|x\|_p = (x_1^p + \cdots + x_n^p)^{1/p}$. For $p = 2$, we will omit the index and just write $\|x\| = \|x\|_2 = \sqrt{\langle x, x \rangle}$.

If A is an $n \times n$ matrix, we denote the ith row-vector of A by $a_{i*} = (a_{i1}, \ldots, a_{in})^T$. Similarly, we denote the jth column of A by $a_{*j} = (a_{1j}, \ldots, a_{nj})^T$. Consider a system of n linear equations in n variables:

$$\sum_{j=1}^{n} a_{ij} x_j \;=\; b_i \;\text{ for }\; 1 \le i \le n, \;\text{ or, in matrix form: }\; Ax \;=\; b. \tag{1}$$

We shall assume throughout that (1) is consistent and that A does not contain a row of zeros. For $p \ge 1$, we define a diagonal matrix
$G_p = \text{diag}(1/\|a_{1*}\|_p, \ldots, 1/\|a_{n*}\|_p)$.
The system obtained from (1) by geometric scaling with the L_p-norm is

$$G_p A x \;=\; G_p b. \tag{2}$$

We call this operation "geometric scaling" because for $p = 2$, any algorithm operating on the scaled system (2) depends only on the hyperplanes of $\mathbb{R}^n$ defined by the equations, and not on any particular algebraic representation of those hyperplanes. As mentioned, GRS(p) denotes geometric row-scaling with the L_p-norm, and GRS stands for GRS(2).

Similarly, GCS(p) (geometric column-scaling) denotes the operation of scaling the system (1) by dividing each column by the L_p-norm of its coefficients. We define $H_p = \text{diag}(1/\|a_{*1}\|_p, \ldots, 1/\|a_{*n}\|_p)$, and obtain the system $AH_p y \;=\; b,\; x = H_p y$. GCS stands for GCS(2).

In some algorithms, GRS(p) is inherent in the following sense: either the scaling is executed at the beginning as an intrinsic part of the algorithm, or, executing the algorithm produces identical results to those obtained when GRS(p) is done at the beginning. Two prime examples of this are KACZ and the Cimmino algorithm, because they depend only on the hyperplanes defined by the equations.

In KACZ, the hyperplanes are chosen in cyclic order and each projection may involve a relaxation parameter λ which determines the extent of the projection towards the hyperplane. The term "cyclic relaxation parameters" means that for every $1 \le i \le n$, there is a relaxation parameter λ_i which is always used with the projection towards the ith hyperplane. In our application of KACZ, the projections are always performed with an entire sweep (or pass) of all the equations. Our formulation of KACZ incorporates this in the algorithm. Let $\lambda_1, \ldots, \lambda_n$ be a sequence of relaxation parameters.

Algorithm 3.1 (KACZ):

set $x^0 \in \mathbb{R}^n$ to an arbitrary value
for $k \;=\; 0, 1, 2, \ldots$ until convergence **do**

set $y^0 = x^k$
for $i = 1,2,\ldots,n$ **do**
set $y^i = y^{i-1} + \lambda_i \dfrac{b_i - \langle a_i, y^{i-1}\rangle}{\|a_i\|^2} a_i$ (3)
enddo
set $x^{k+1} = y^n$
enddo

It is well-known that KACZ is the SOR algorithm applied to the "normal equations" system $AA^T y = b,\ x = A^T y$. It is clear from the above that if GRS is applied to the linear system (1) before executing KACZ, then the resulting sequence of iterates will be identical to those produced without GRS, so GRS is inherent in KACZ. The application of GRS leads to the system (2) with $p = 2$. Clearly, it is more efficient to apply GRS at the beginning, because then Eq. (3) can be replaced by

$$y^i = y^{i-1} + \lambda_i \left(d_i - \langle c_{i*}, y^{i-1}\rangle\right) c_{i*}, \tag{4}$$

where $C = G_2 A$ and $d = G_2 b$ (G_2 is the above-defined matrix G_p, with $p = 2$).

It follows from the above that GRS is inherent in any algorithm that is based on KACZ iterations, such as CGMN and CARP-CG.

The following theorem explains why it is useful to apply GRS before using the normal equations, especially when the original system has large off-diagonal elements.

Theorem 3.1 *Let A be an $n \times n$ matrix such that no two rows are colinear. Let C be obtained from A by GRS, i.e., $C = G_2 A$, and let $D = CC^T$. Then, all the diagonal elements of D are equal to 1, and all the off-diagonal elements are < 1.*

Proof: The proof follows from the Cauchy-Schwarz inequality. The following is a geometric proof. For $i \neq j$, denote by θ_{ij} the angle between the row vectors c_{i*} and c_{j*}. Since every two rows are not colinear, we have $|\cos(\theta_{ij})| < 1$. Also, every row of C has an L_2-norm equal to 1. Consider now a diagonal element of D: $d_{ii} = \langle c_{i*}, c_{i*}\rangle = 1$. For the off-diagonal elements of D, we have: if $i \neq j$, then $|d_{ij}| = |\langle c_{i*}, c_{j*}\rangle| = \|c_{i*}\| \times \|c_{j*}\| \times |\cos(\theta ij)| = |\cos(\theta ij)| < 1$. ∎

Note that if A is sparse and for any two (different) rows i, j, the nonzero elements in rows i and j do not occupy identical columns, then A satisfies the condition of Theorem 3.1. It is often remarked that the normal equations are not useful because the spectral radius of AA^T is the square of that of A. However, Theorem 3.1 indicates that if GRS is applied first, then the resulting system is manageable because of the relatively large diagonal elements in the normal equations. Analogously to Theorem 3.1, if E is obtained from A by GCS, then $E^T E$ will have 1's on the diagonal, and the off-diagonal elements will be < 1.

Consider now the two normal equations systems obtained from (1):

$$A^T Ax = A^T b \tag{5}$$

$$AA^T y = b, \;\; x = A^T y \tag{6}$$

Applying CG to (5) is called CGNR, and applying CG to (6) is called CGNE. We will use the notation CGNR+GRS to denote that CGNR is preceded by GRS, and CGNR+GCS denotes that CGNR is preceded by GCS.

CGNR+GRS is of special interest because it turns out to be nothing else but the CG acceleration of the Cimmino algorithm. To see this, consider the (relaxed) Cimmino iteration applied to (1):

$$x^{k+1} \;=\; x^k + \frac{\lambda}{n} \sum_{i=1}^{n} \frac{b_i - \langle a_{i*}, x^k \rangle}{\|a_{i*}\|^2} a_{i*}, \tag{7}$$

where λ is a relaxation parameter. Let C denote the matrix obtained from A by applying GRS (i.e., $c_{ij} = a_{ij}/\|a_{i*}\|$) and let $d = (d_1, \ldots, d_n)^T$, where $d_i = b_i/\|a_{i*}\|$. The Cimmino iteration is then

$$x^{k+1} \;=\; x^k + \frac{\lambda}{n} \sum_{i=1}^{n} \left(d_i - \langle c_{i*}, x^k \rangle \right) c_{i*} \tag{8}$$

A straightforward algebraic derivation of (8) yields

$$x^{k+1} \;=\; x^k + \frac{\lambda}{n} \left(C^T d - C^T C x^k \right). \tag{9}$$

Therefore, a vector x^* is a solution of (7) iff it is a solution of the system

$$C^T Cx = C^T d. \tag{10}$$

To get a CG acceleration of Cimmino, we simply apply CG to (10), which is exactly CGNR applied to the system after GRS was applied. Our implementation of CGNR+GRS is similar to the relation between (8) and (9), i.e., whenever a computation of the form $C^T Cx$ is needed in the CG iteration, it is calculated as $\sum_{i=1}^{n} \langle c_{i*}, x^k \rangle c_{i*}$. Note that this calculation can be easily parallelized when the matrix is stored in some distributed format, such as the DMSR (distributed matrix sparse row) format used by the AZTEC software package [Tuminaro, Heroux, Hutchinson, and Shadid (1999)].

4 Setup of the numerical experiments

In two dimensions, the general form of the second-order differential equations in this study was

$$\frac{\partial}{\partial x}(a(x,y)u_x)+\frac{\partial}{\partial y}(b(x,y)u_y)+\cdots \;=\; F,$$

where a and b are given functions of x and y, "$\cdots$" stands for lower-order derivatives, and F is a prescribed RHS. In three dimensions, there are three given functions a,b,c of x,y,z, and a second order partial derivative w.r.t. z is also included. Boundary conditions in the examples studied here were Dirichlet. The discretization of the second-order derivatives at a given grid point (i,j) was done using central differences, e.g., $\frac{\partial}{\partial x}(au_x)$ was approximated at grid point (i,j) as

$$\begin{aligned}
\frac{\partial}{\partial x}(au_x)_{i,j} &= \left((au_x)_{i+\frac{1}{2},j}-(au_x)_{i-\frac{1}{2},j}\right)/\Delta x \\
&= \left(a_{i+\frac{1}{2},j}(u_{i+1,j}-u_{i,j})/\Delta x - a_{i-\frac{1}{2},j}(u_{i,j}-u_{i-1,j})/\Delta x\right)/\Delta x \\
&= \left(-(a_{i+\frac{1}{2},j}+a_{i-\frac{1}{2},j})u_{i,j}+a_{i+\frac{1}{2},j}u_{i+1,j}+a_{i-\frac{1}{2},j}u_{i-1,j}\right)/\Delta x^2
\end{aligned}$$

All problems were discretized with equally-spaced grids, and the initial value was taken as $u^0=0$.

Our experiments considered 12 algorithm/preconditioner combinations:

- Bi-CGSTAB in four variations: with and without GRS, and both forms with and without ILU(0).
- GMRES, also with the same four versions as Bi-CGSTAB.
- CGNR without scaling, and also with GRS and with GCS (but not both).
- CGMN (which is also CARP-CG on a single processor).

The tests were run on a Pentium IV 2.8GHz processor with 3GB memory, running Linux. The code was compiled with the GNU compiler. Bi-CGSTAB, GMRES and ILU(0) were run within the AZTEC software system, and GMRES was used in a restart mode, with Krylov subspace size of 10.

4.1 Stopping tests

There are several stopping criteria which one may apply to iterative systems. Our stopping criterion was to use the relative residual: $\|b-Ax\|/\|b-Ax^0\|<\varepsilon$, where

ε was taken as 10^{-4}, 10^{-7} and 10^{-10}. In some of the cases, this was not attainable. Since this stopping criterion depends on the scaling of the equations, we always made this test on the geometrically-scaled system using the L_2-norm. In the following sections, the relative residual will be denoted as rel-res. In order to limit the time taken by the methods implemented in AZTEC, the maximum number of iterations was set to 10,000. The AZTEC library has several other built-in stopping criteria: numerical breakdown, numerical loss of precision and numerical ill-conditioning.

One should note that the test for numerical breakdown in AZTEC uses the machine precision `DBL_EPSILON`, and this may result in a premature notice of numerical breakdown in some cases. To get around this problem, we suggest to multiply the variable `brkdown_tol` in the Bi-CGSTAB algorithm by some small number, e.g., 10^{-10}. (`brkdown_tol` is normally set equal to `DBL_EPSILON`, which is approximately 2.22×10^{-16} on our machine).

5 Problem 1

This problem actually consists of three related cases taken from (Gerardo-Giorda, Tallec, and Nataf, 2004, §5.1), where they are solved using DD. The problems are three-dimensional convection-diffusion-reaction equations with discontinuous coefficients. The domain consists of the unit cube divided into two subdomains by the plane $x = \frac{1}{2}$. The basic equation considered here is

$$-\operatorname{div}(\nu(x)\nabla u) + \vec{b}\bullet\nabla u + u = 0, \tag{11}$$

where

$$\nu(x) = \begin{cases} \nu_1 = 10^{-1} & \text{if } x < \frac{1}{2}, \\ \nu_2 = 10^{-5} & \text{otherwise.} \end{cases}$$

The small value of ν_2 results in equations that are strongly convection dominated. Dirichlet boundary conditions are taken as $u = 1$ on the $z = 0$ plane and $u = 0$ on the other boundaries of the unit cube. The vector $\vec{b}$ determines the direction of the flow. Four different cases were examined; the first three are identical to those that appeared in [Gerardo-Giorda, Tallec, and Nataf (2004)], and the fourth case was added in order to answer some questions raised by the results.

- **Problem 1A:** $\vec{b} = (1,0,0)$: the flow is perpendicular to the interface, and Eq. (11) has only one convection term.
- **Problem 1B:** $\vec{b} = (0,1,1)$: the flow is parallel to the interface, and Eq. (11) has two convection terms.

- **Problem 1C:** $\vec{b} = (1,3,5)$: the flow direction is oblique to the interface, and Eq. (11) has three convection terms.
- **Problem 1D:** $\vec{b} = (0,1,0)$: the flow is parallel to the interface, and Eq. (11) has only one convection term.

We also considered a continuous version of Problem 1C with $\nu_1 = \nu_2 = 10^{-5}$. All the problems are indefinite with eigenvalues in the four quadrants of the complex plane. Two discretizations were considered: $40\times40\times40$ (64,000 equations), and $80\times80\times80$ (512,000 equations). The eigenvalue data was obtained with a discretization of $12\times12\times12$. Detailed results are presented in the following subsections. In all the tables, only methods that achieved at least one convergence goal are shown. The timing results include the setup time for ILU(0) (if it is applied), and dashes indicate no convergence. λ is the relaxation parameter used by CGMN.

5.1 Problem 1A

Tables 1 and 2 show the results for Problem 1A, for the two grid sizes. Both runtimes and number of iterations are shown.

Table 1: Results for Problem 1A, grid size $= 40\times40\times40$.

	Runtimes (sec.)			**No. of iterations**		
Convergence goal:	10^{-4}	10^{-7}	10^{-10}	10^{-4}	10^{-7}	10^{-10}
Bi-CGSTAB+ILU(0)	1.01	1.26	1.45	17	26	33
with GRS	0.98	1.26	1.45	16	26	33
GMRES	28.38	51.79	75.45	1679	3034	4411
with GRS	20.45	42.04	63.55	1198	2472	3726
GMRES+ILU(0)	1.09	1.59	2.11	23	43	63
with GRS	1.09	1.56	2.11	23	42	63
CGNR	34.31	51.57	68.75	5362	8059	10744
with GRS	10.06	14.51	17.87	1560	2250	2771
with GCS	8.09	11.67	14.32	1252	1807	2216
CGMN ($\lambda = 1.50$)	3.83	6.44	8.87	350	589	812

The tables show that Bi-CGSTAB without ILU(0) did not converge, and GRS did not help. Also, both Bi-CGSTAB and GMRES with ILU(0) performed extremely well, and GRS had no effect on them. GRS reduced the runtime of GMRES (without ILU(0)) by 16%–28%, but these runtimes were relatively very large. With CGNR, GRS and GCS were more effective, but the runtimes were quite large. Due to its robustness, CGMN converged in all cases, but it was relatively slow.

Table 2: Results for Problem 1A, grid size $= 80\times80\times80$.

	Runtimes (sec.)			**No. of iterations**		
Convergence goal:	10^{-4}	10^{-7}	10^{-10}	10^{-4}	10^{-7}	10^{-10}
Bi-CGSTAB+ILU(0)	20.1	26.0	31.4	30	45	59
with GRS	19.1	26.0	30.6	28	45	57
GMRES	829	1514	—	6002	10956	—
with GRS	591	1364	—	4282	9871	—
GMRES+ILU(0)	24.2	37.3	51.5	58	106	158
with GRS	20.5	34.3	47.9	45	95	145
CGNR	1398	—	—	20591	—	—
with GRS	374	570	712	5520	8404	10485
with GCS	325	476	586	4787	7009	8632
CGMN ($\lambda = 1.50$)	114	182	269	1107	1770	2615

Fig. 1 shows the distribution of eigenvalues for Problem 1A, for the original and the scaled cases. We can see that even though the eigenvalue concentration around the origin was "pushed" away from the origin, the results of Tables 1 and 2 show that this does not necessarily lead to better runtime results (except for GMRES).

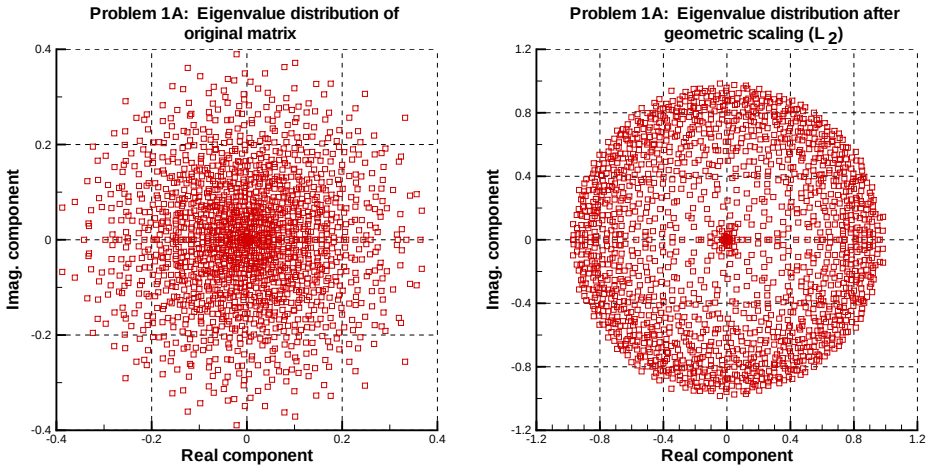

Figure 1: Eigenvalue distribution for Problem 1A, for the original and the scaled cases.

5.2 Problem 1B

Tables 3 and 4 show the results for Problem 1B, for the two grid sizes. Bi-CGSTAB and GMRES+ILU(0) did not converge in any form. GRS decreased the runtimes of GMRES (without ILU(0)) by about 25%–28% for the coarse grid, and even less for the fine grid. GRS and GCS decreased the runtime of CGNR quite significantly and also enabled its convergence in the higher accuracy goal on the fine grid. The relatively good performance of CGMN in this case is quite strikingj

Table 3: Results for Problem 1B, grid size $= 40\times40\times40$.

	Runtimes (sec.)			No. of iterations		
Convergence goal:	10^{-4}	10^{-7}	10^{-10}	10^{-4}	10^{-7}	10^{-10}
GMRES	49.01	88.28	127.74	2866	5164	7455
with GRS	35.24	66.43	96.24	2064	3891	5637
CGNR	38.82	76.71	115.00	6066	11988	17970
with GRS	7.35	11.36	14.28	1140	1761	2214
with GCS	8.22	12.16	15.00	1272	1882	2321
CGMN ($\lambda = 1.50$)	2.35	3.60	4.71	228	350	458

Table 4: Results for Problem 1B, grid size $= 80\times80\times80$.

	Runtimes (sec.)			No. of iterations		
Convergence goal:	10^{-4}	10^{-7}	10^{-10}	10^{-4}	10^{-7}	10^{-10}
GMRES	1012	1851	—	7350	13442	—
with GRS	965	2018	—	7002	965	—
CGNR	1371	—	—	20160	—	—
with GRS	265	444	568	3892	6521	8348
with GCS	300	476	593	4421	7000	8717
CGMN ($\lambda = 1.70$)	68	105	137	661	1020	1340

5.3 Problem 1C

Tables 5 and 6 show the results for Problem 1C. These results are quite similar to those of Problem 1B, except that now, GRS reduces the runtime of GMRES by about 42% in the coarse grid case, and it was also helpful to GMRES in the finer grid case. GRS and GCS were both very helpful to CGNR, but here CGMN also takes a clear lead.

Table 5: Results for Problem 1C, grid size = $40\times40\times40$.

	Runtimes (sec.)			No. of iterations		
Convergence goal:	10^{-4}	10^{-7}	10^{-10}	10^{-4}	10^{-7}	10^{-10}
GMRES	71.26	134	205	4177	7856	11957
with GRS	40.87	78.47	119.5	2385	4579	6972
CGNR	34.0	73.9	111.8	5314	11548	17476
with GRS	7.86	14.8	21.9	1229	2316	3421
with GCS	7.59	14.5	21.4	1170	2239	3294
CGMN ($\lambda=1.20$)	2.75	6.07	9.4	267	590	913

Table 6: Results for Problem 1C, grid size = $80\times80\times80$.

	Runtimes (sec.)			No. of iterations		
Convergence goal:	10^{-4}	10^{-7}	10^{-10}	10^{-4}	10^{-7}	10^{-10}
GMRES	1388	—	—	10040	—	—
with GRS	609	1126	—	4394	8126	—
CGNR	1179	—	—	17995	—	—
with GRS	246	444	608	3731	6737	9231
with GCS	225	367	498	3407	5577	7566
CGMN ($\lambda=1.50$)	67	120	188	638	1152	1800

Table 7 provides some eigenvalue information for Problem 1C, for the original and the scaled cases. The last column shows the number of eigenvalues in the percentile containing the origin, when the interval of (the real part) of the eigenvalues is divided into 100 equal-sized subintervals. GRS increased the condition number and it had only a small effect on the number of eigenvalues around the origin.

Table 7: Basic eigenvalue information for Problem 1C.

Matrix	$\lambda_{\min}$	$\lambda_{\max}$	$\lambda_{\max}/\lambda_{\min}$	No. of eigenvalues around $x=0$
Original	0.240E-3	0.619E+0	0.258E+4	45
With GRS	0.200E-3	1.008E+0	0.480E+4	29

Fig. 2 shows the distribution of the eigenvalues for Problem 1C, for the original and the scaled cases. Note that the eigenvalue distribution for the geometrically scaled matrix of Problem 1C is quite similar to that of the scaled matrix of Problem 1A.

However, the behavior of Bi-CGSTAB and GMRES on the two problems is very different. This fact clearly indicates that the eigenvalue distribution is not the sole factor affecting convergence of these algorithms.

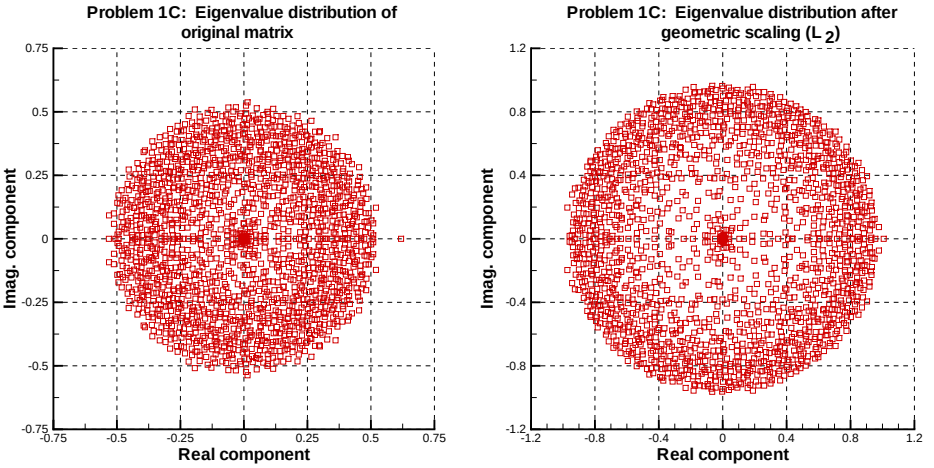

Figure 2: Eigenvalue distribution for Problem 1C, for the original and the scaled cases.

5.4 Problem 1D and the continuous case

The results of the first three versions of Problem 1 raise the following question: what is the cause of the different behavior of the algorithms on Problem 1A and on the other two problems? Is it due to the direction of the flow with respect to the boundary between the two subdomains? The purpose of Problem 1D is to answer this question.

In this case (1D), the flow is parallel to the interface, as in case 1B, but, similarly to case 1A, there is only one convection term in the differential equation (u_y). The results that we got for this variant are very similar to those obtained with Problem 1A, and we omit them. This clearly indicates that the direction of the flow is irrelevant. What is relevant is that in cases 1A and 1D, there is only one convection term, and it contributes only two large off-diagonal elements to each equation in the linear system. In Problems 1B and 1C there are, respectively, four and six large off-diagonal elements in each linear equation.

We conclude from this that the different behavior of the solution methods depends very largely on the *number* of large off-diagonal elements in each linear equation. These results show that ILU(0) is very helpful to Bi-CGSTAB and GMRES, but

only when the number of large off-diagonal elements in the linear equation is no more than two. For the more difficult cases, we have reasonable results with the scaled versions of CGNR and very good results with CGMN.

These results raise a second question: what is the cause of the difficulties of Bi-CGSTAB and GMRES on Problems 1B and 1C? Is it the discontinuous coefficients or the large convection terms, or is it a combination of both factors? In order to answer this question, we tested yet another variant of the problem. We modified Problem 1C so that there were no discontinuities, but just large convection terms. This was done by taking $\nu_1 = \nu_2 = 10^{-5}$ throughout the unit cube, which was discretized with a grid of $40\times40\times40$. The results are shown in Table 8.

Table 8: Results for a continuous version of Problem 1C, with large convection terms and grid size = $40\times40\times40$.

	Runtimes (sec.)			No. of iterations		
Convergence goal:	10^{-4}	10^{-7}	10^{-10}	10^{-4}	10^{-7}	10^{-10}
GMRES	49.23	92.46	136.24	2871	5406	7952
with GRS	46.89	88.98	131.18	2754	5226	7705
CGNR	6.83	13.88	21.07	1067	2169	3292
with GRS	6.66	13.59	20.68	1042	2124	3234
with GCS	6.66	13.59	20.70	1042	2126	3237
CGMN ($\lambda = 1.10$)	2.85	6.29	9.68	277	611	941

The following points should be noted in comparing Tables 5 and 8:

- Bi-CGSTAB, with and without ILU(0) and/or GRS, did not converge at all.
- GMRES (without GRS) performed better on the continuous case.
- GRS was only slightly helpful to GMRES in the continuous case, but its help in the discontinuous case was significant. In fact, GMRES+GRS was even better in the discontinuous case.
- CGNR (without GRS) performed much better on the continuous case.
- CGNR with GRS or GCS performed very similarly on the two cases.
- CGMN also performed very similarly on the two cases (and better than CGNR with GRS or GCS).

The above results indicate that the primary cause of the difficulties to Bi-CGSTAB and GMRES in Problems 1B and 1C is the presence of several large off-diagonal

elements. For GMRES, the discontinuous coefficients were also a contributing factor.

6 Problem 2

Problem 2 is also from (Gerardo-Giorda, Tallec, and Nataf, 2004, §5.2, Test 4), where it was solved with DD. The equation is the same as in Problem 1, namely,

$$-\mathrm{div}(\nu(x)\nabla u) + \vec{b}\bullet\nabla u + u = 0.$$

The domain consists of the cube $\Omega = [-0.5,0.5]\times[-0.5,0.5]\times[0,1]$, which is partitioned into eight equal sub-cubes, numbered in a clockwise helicoidal way from $\Omega_1 = [-0.5,0]\times[-0.5,0]\times[0,0.5]$ to $\Omega_8 = [0,0.5]\times[-0.5,0]\times[0.5,1]$. The velocity field is given by $\vec{b} = (-2\pi y, 2\pi x, \sin(2\pi x))$. The values of ν in domains $\Omega_1,\ldots,\Omega_8$ are, respectively, $\nu_1,\ldots,\nu_8$, where $\nu_1 = 10^{-1}, \nu_3 = 10^{-2}, \nu_6 = 10^{-3}, \nu_8 = 10^{-4}$, and $\nu_2 = \nu_4 = \nu_5 = \nu_7 = 10^{-6}$. Among the four tests of (Gerardo-Giorda, Tallec, and Nataf, 2004, §5.2), this case exhibits the most variance in the values of ν. Dirichlet boundary conditions are taken as $u = 1$ on the $z = 0$ face and $u = 0$ elsewhere on the boundary of Ω. The domain was discretized with a grid of $40\times40\times40$.

Table 9 below presents the runtimes and number of iterations of the methods that achieved at least one convergence goal. We can see that GRS was quite helpful to GMRES, and both GRS and GCS were very helpful to CGNR. Similarly to Problems 1B, 1C, and 1D, CGMN achieved the best timing results.

Table 9: Runtimes and no. of iterations for Problem 2.

	Runtimes (sec.)			**No. of iterations**		
Convergence goal:	10^{-4}	10^{-7}	10^{-10}	10^{-4}	10^{-7}	10^{-10}
GMRES	81.8	156.1	—	4798	9152	—
with GRS	38.5	86.1	133.9	2256	5051	7849
CGNR	47.9	90.8	133.7	7491	14181	20881
with GRS	7.35	10.7	13.9	1150	1670	2170
with GCS	8.81	11.9	14.0	1378	1855	2195
CGMN ($\lambda = 1.45$)	2.44	3.86	4.87	240	380	477

7 Problem 3

This problem is based on a three-dimensional example from [Graham and Hagger (1999)], to which we added convection terms. The differential equation is the

following:

$$-\frac{\partial}{\partial x}(au_x) - \frac{\partial}{\partial y}(au_y) - \frac{\partial}{\partial z}(au_z) + du_x + eu_y + fu_z = 0,$$

where the domain is the unit cube, and $a(x,y,z)$ is defined as

$$a(x,y,z) = \begin{cases} 10^4 & \text{if } \frac{1}{3} < x,\ y,\ z < \frac{2}{3}, \\ 1 & \text{otherwise.} \end{cases}$$

The Dirichlet boundary conditions are prescribed with $u = 1$ on the $z = 0$ plane and $u = 0$ on the other boundaries. The convection terms were taken as equal, $d = e = f$, and the unit cube was divided into a grid of $40 \times 40 \times 40$. The resulting linear systems are indefinite, with eigenvalues in the four quadrants of the imaginary plane. This problem was studied extensively in Gordon and Gordon (2009b), where it was shown that the usefulness of GRS degrades as the convection terms are increased.

In order to compare the performance of the various methods as the convection increases, we ran tests with convection terms of 100, 200, 500, and 1000. The results are summarized in Table 10 below (only for methods which achieved at least one convergence goal).

Table 10: Time in seconds for the different methods to achieve the three convergence goals, as the convection terms are increased. Minimal times are shown in boldface.

Convection:	**100**			**200**			**500**			**1000**		
Convergence goal:	10^{-4}	10^{-7}	10^{-10}	10^{-4}	10^{-7}	10^{-10}	10^{-4}	10^{-7}	10^{-10}	10^{-4}	10^{-7}	10^{-10}
Bi-CGSTAB+GRS	1.00	2.77	3.62	2.77	7.20	9.85	9.74	20.99	38.24	—	—	—
Bi-CGSTAB+ILU(0)	1.35	1.56	1.74	2.30	—	—	—	—	—	—	—	—
with GRS	**0.76**	**1.42**	**1.67**	**1.90**	—	—	—	—	—	—	—	—
GMRES+GRS	2.05	—	—	2.23	—	—	4.50	—	—	—	—	—
with ILU(0)	**0.76**	—	—	—	—	—	—	—	—	—	—	—
CGNR+GRS	5.10	10.42	11.28	4.78	9.46	10.23	5.96	12.40	13.42	7.29	14.96	16.15
CGNR+GCS	9.36	10.44	11.28	8.66	9.43	10.24	11.30	12.35	13.39	13.67	14.90	16.07
CGMN	2.01	4.27	4.60	1.93	**4.56**	**4.95**	**1.96**	**5.77**	**6.30**	**2.45**	**6.92**	**7.59**

Several points are worth noting:

- Bi-CGSTAB, with ILU(0) and GRS, excels with convection = 100, and also with the low-order goal with convection = 200. However, this method failed in the more difficult cases.

- Bi-CGSTAB+GRS managed to handle convection up to 500, but the increase in time was much worse than for CGNR (with GRS or GCS) or CGMN.
- Starting from convection = 200 and convergence goal $= 10^{-7}$, CGMN takes a clear lead.
- CGNR by itself did not converge in any of the cases, but with GRS or GCS, it achieved all the convergence goals at all the convection values.
- For the convergence goal of 10^{-4}, CGNR with GRS was almost twice as fast as CGNR with GCS. For the higher orders, there was little difference between the two scaling methods.

Not shown in Table 10 is the fact that the optimal relaxation parameter λ of CGMN varied somewhat with the convection: 1.65, 1.55, 1.45 and 1.35 for convection values of 100, 200, 500 and 1000, respectively. However, the runtimes of CGMN varied very little when λ was changed between these values. Since CGMN is more relevant for the higher convection values, a fixed value of $\lambda = 1.45$ is sufficient to obtain reasonable results on this problem.

8 Conclusions and further research

This paper extends the authors' previous work on two topics: the solution of linear systems derived from strongly convection dominated PDEs using the CGMN algorithm [Björck and Elfving (1979); Gordon and Gordon (2008)] and its block-parallel version CARP-CG [Gordon and Gordon (2009a)], and the geometric row scaling (GRS) technique [Gordon and Gordon (2009b)] as a useful preconditioner for nonsymmetric linear systems with discontinuous coefficients. GRS consists of dividing each equation by the L_2-norm of its vector of coefficients. The usefulness of this method is limited to small to moderate convection terms.

The main topic of this work is linear systems with very large off-diagonal elements and discontinuous coefficients (LODC systems). It is shown that if a matrix C is obtained from A by GRS, then all the diagonal elements of CC^T are larger than all the off-diagonal ones. This gives a theoretical foundation for the robustness of the CGMN and CARP-CG, and also to CGNR when applied to a system scaled by GCS (geometric column-scaling).

The above algorithms were compared with Bi-CGSTAB and restarted GMRES, in all possible combinations of using GRS and/or ILU(0). The algorithms were tested on various test problems with large convection terms and discontinuous coefficients. While GRS is of some help when the off-diagonal elements are small or few, it fails when the off-diagonal elements become strongly dominant. It is exactly

in those cases that CGMN excels. CGNR with GRS or GCS is equally robust, but less efficient.

The main conclusion of this work is the usefulness of two complementary tools for nonsymmetric LODC systems (with equations containing more than two large off-diagonal elements):

- GRS for LODC systems with small to moderate convection terms.
- CGMN, CARP-CG, and CGNR+GRS/GCS for LODC systems with very large convection terms.

Future work will examine other types of LODC problems, such as the Helmholtz equation with large wave numbers in heterogeneous media, and circuit problems.

Acknowledgement: The authors wish to thank the anonymous reviewers for their helpful comments.

References

Arioli, M.; Duff, I. S.; Noailles, J.; Ruiz, D. (1992): A block projection method for sparse matrices. *SIAM J. on Scientific & Statistical Computing*, vol. 13, pp. 47–70.

Arioli, M.; Duff, I. S.; Ruiz, D.; Sadkane, M. (1995): Block Lanczos techniques for accelerating the block Cimmino method. *SIAM J. on Scientific & Statistical Computing*, vol. 16, pp. 1478–1511.

Benzi, M. (2002): Preconditioning techniques for large linear systems: a survey. *Journal of Compuational Physics*, vol. 182, pp. 418–477.

Benzi, M.; Szyld, D. B.; van Duin, A. (1999): Orderings for incomplete factorization preconditioning of nonsymmetric problems. *SIAM J. on Scientific Computing*, vol. 20, no. 5, pp. 1652–1670.

Björck, Å.; Elfving, T. (1979): Accelerated projection methods for computing pseudoinverse solutions of systems of linear equations. *BIT*, vol. 19, pp. 145–163.

Bramley, R.; Sameh, A. (1992): Row projection methods for large nonsymmetric linear systems. *SIAM J. on Scientific & Statistical Computing*, vol. 13, pp. 168–193.

Cimmino, G. (1938): Calcolo approssimato per le soluzioni dei sistemi di equazioni lineari. *La Ricerca Scientifica XVI, Series II, Anno IX*, vol. 1, pp. 326–333.

Duff, I. S.; Koster, J. (2001): On algorithms for permuting large entries to the diagonal of a sparse matrix. *SIAM J. on Matrix Analysis & Applications*, vol. 22, no. 4, pp. 973–996.

Duff, I. S.; van der Vorst, H. A. (1998): Preconditioning and parallel preconditioning. Technical Report TR/PA/98/23, CERFACS, Toulouse, France, 1998.

Elfving, T. (1980): Block-iterative methods for consistent and inconsistent linear equations. *Numerische Mathematik*, vol. 35, pp. 1–12.

Erlangga, Y. A.; Vuik, C.; Oosterlee, C. W. (2004): On a class of preconditioners for solving the Helmholtz equation. *Applied Numerical Mathematics*, vol. 50, pp. 409–425.

Gambolati, G.; Pini, G.; Ferronato, M. (2003): Scaling improves stability of preconditioned CG-like solvers for FE consolidation equations. *International J. for Numerical & Analytical Methods in Geomechanics*, vol. 27, pp. 1043–1056.

Gerardo-Giorda, L.; Tallec, P. L.; Nataf, F. (2004): A Robin-Robin preconditioner for advection-diffusion equations with discontinuous coefficients. *Computer Methods in Applied Mechanics & Engineering*, vol. 193, pp. 745–764.

Glowinski, R.; Wheeler, M. F. (1988): Domain decomposition and mixed finite element methods for elliptic problems. In Glowinski, R.; Golub, G. H.; Meurant, G. A.; Périaux, J.(Eds): *First International Symposium on Domain Decomposition Methods for Partial Differential Equations*, pp. 144–172. SIAM, Philadelphia, PA, USA.

Gordon, D.; Gordon, R. (2005): Component-averaged row projections: A robust, block-parallel scheme for sparse linear systems. *SIAM J. on Scientific Computing*, vol. 27, pp. 1092–1117.

Gordon, D.; Gordon, R. (2008): CGMN revisited: robust and efficient solution of stiff linear systems derived from elliptic partial differential equations. *ACM Trans. on Mathematical Software*, vol. 35, no. 3, pp. 18:1–18:27.

Gordon, D.; Gordon, R. (2009): CARP-CG: a robust and efficient parallel solver for linear systems, applied to strongly convection dominated PDEs. Technical report, Dept. of Computer Science, University of Haifa, Haifa, Israel, 2009a. Submitted for publication. http://cs.haifa.ac.il/∼gordon/carp-cg.pdf.

Gordon, D.; Gordon, R. (2009): Geometric scaling: a simple and effective preconditioner for linear systems with discontinuous coefficients. Technical report,

Dept. of Computer Science, University of Haifa, Haifa, Israel, 2009b. Submitted for publication. http://cs.haifa.ac.il/∼gordon/gs.pdf.

Graham, I. G.; Hagger, M. J. (1999): Unstructured additive Schwarz-conjugate gradient method for elliptic problems with highly discontinuous coefficients. *SIAM J. on Scientific Computing*, vol. 20, no. 6, pp. 2041–2066.

Jönsthövel, T. B.; van Gijzen, M. B.; Vuik, C.; Kasbergen, C.; Scarpas, A. (2009): Preconditioned conjugate gradient method enhanced by deflation of rigid body modes applied to composite materials. *CMES: Computer Modeling in Engineering & Sciences*, vol. 47, no. 2, pp. 97–118.

Kaczmarz, S. (1937): Angenäherte Auflösung von Systemen linearer Gleichungen. *Bulletin de l'Académie Polonaise des Sciences et Lettres*, vol. A35, pp. 355–357.

Liu, C.-S.; Atluri, S. N. (2008): A fictitious time integration method (FTIM) for solving mixed complementarity problems with applications to non-linear optimization. *CMES: Computer Modeling in Engineering & Sciences*, vol. 34, no. 2, pp. 155–178.

Liu, C.-S.; Atluri, S. N. (2009): A fictitious time integration method for the numerical solution of the Fredholm integral equation and for numerical differentiation of noisy data, and its relation to the filter theory. *CMES: Computer Modeling in Engineering & Sciences*, vol. 41, no. 3, pp. 243–261.

Liu, C.-S.; Yeih, W.; Atluri, S. N. (2009): On solving the ill-conditioned system Ax=b: general purpose conditioners obtained from the boundary-collocation solution of the Laplace equation, using Trefftz expansions with multiple length scales. *CMES: Computer Modeling in Engineering & Sciences*, vol. 44, no. 3, pp. 281–311.

Meurant, G. (1999): *Computer Solution of Large Linear Systems*. Elsvier, Amsterdam.

Quarteroni, A.; Valli, A. (1999): *Domain Decomposition Methods for Partial Differential Equations*. Oxford University Press, Oxford.

Rice, J. R.; Tsompanopoulou, P.; Vavalis, E. (2000): Interface relaxation methods for elliptic differential equations. *Applied Numerical Mathematics*, vol. 32, pp. 219–245.

Saad, Y. (2005): Multilevel ILU with reorderings for diagonal dominance. *SIAM J. on Scientific Computing*, vol. 27, no. 3, pp. 1032–1057.

Saad, Y.; Schultz, M. H. (1986): GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. on Scientific & Statistical Computing*, vol. 7, pp. 856–869.

Schend, O.; Röllin, S.; Gupta, A. (2004): The effects of unsymmetric matrix permutations and scalings in semiconductor device and circuit simulation. *IEEE Trans. on Computer-Aided Design of Integrated Circuits & Systems*, vol. 23, no. 3, pp. 400–411.

Smith, B.; Bjørstad, P.; Gropp, W. (1996): *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Diffferential Equations*. Cambridge University Press, Cambridge, UK.

Takei, A.; Yoshimura, S.; Kanayama, H. (2008): Large-scale parallel finite element analyses of high frequency electromagnetic field in commuter trains. *CMES: Computer Modeling in Engineering & Sciences*, vol. 31, no. 1, pp. 13–23.

Takei, A.; Yoshimura, S.; Kanayama, H. (2009): Large-scale full wave analysis of electromagnetic field by hierarchical domain decomposition method. *CMES: Computer Modeling in Engineering & Sciences*, vol. 40, no. 1, pp. 63–81.

Tuminaro, R. S.; Heroux, M. A.; Hutchinson, S. A.; Shadid, J. N. (1999): AZTEC user's guide. Technical Report SAND99-8801J, Sandia National Laboratories, Albuquerque, New Mexico, 1999.

van der Sluis, A. (1969): Condition numbers and equilibration of matrices. *Numerische Mathematik*, vol. 14, pp. 14–23.

van der Vorst, H. A. (1992): Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM J. on Scientific & Statistical Computing*, vol. 13, pp. 631–644.

Widlund, O. B. (1971): On the effects of scaling of the Peaceman-Rachford method. *Mathematics of Computation*, vol. 25, no. 113, pp. 33–41.