

# On Complex Preconditioning for the solution of the Kohn-Sham Equation for Molecular Transport \*

Daniel Osei-Kuffuor <sup>†</sup>    Lingzhu Kong <sup>‡</sup>    Yousef Saad <sup>†</sup>    James R. Chelikowsky <sup>§</sup>

December 23, 2007

## Abstract

This paper analyzes the performance of a few preconditioners for solving the complex linear systems which originate from the application of real space pseudopotential methods in the study of electron transport properties of nanoscale junctions. These linear systems of equations are part of a self-consistent loop to compute the charge density and corresponding potential at the nanoscale junctions. The coefficient matrices for these systems have a regular structure with two dense blocks associated with boundary conditions. These dense blocks cause difficulties to general preconditioners, due to the amount of fill-in which they tend to generate. The preconditioners studied are of the general-purpose kind, such as ILU with threshold (ILUT), ILU with level of fill (ILUK), and the Algebraic Recursive Multilevel Solvers (ARMS). The study shows that ARMS with diagonal dominance – based nonsymmetric ordering (ddPQ) is generally more robust than the other preconditioners which were tested.

## 1 Introduction

Molecular transport is a rapidly growing field which spans disciplines such as physics, chemistry, and materials science [1]. It opens the opportunity to build electronic devices of nanoscale size. Many novel phenomena such as quantized conductance have been reported [2]. To understand these phenomena and to predict other transport properties as well, fully quantum mechanical approaches based on pseudopotential Density Functional Theory (DFT) [3] have been proposed [4]. The underlying equation which governs this approach is the Kohn-Sham equation [3].

The physical system of interest is a molecular junction, which consists of two semi-infinite electrodes connected by a molecule or group of atoms as shown in figure 1. For the purpose of calculating the current through the molecule, the scattering solutions of the Kohn-Sham

---

\*This work is supported by the National Science Foundation under grants DMR-0325218 and DMR-0551195, by the U.S. Department of Energy under grants DE-FG02-03ER25585, DE-FG02-89ER45391, and DE-FG02-03ER15491, and by the Minnesota Supercomputer Institute.

<sup>†</sup>Department of Computer Science and Engineering, University of Minnesota, 200 Union Street S.E, Minneapolis, MN 55455. Email: {dosei, saad}@cs.umn.edu.

<sup>‡</sup>Department of Chemical Engineering and Materials Science, University of Minnesota, Minneapolis, Minnesota 55455, USA

<sup>§</sup>Center for Computational Material, Institute for Computational Engineering and Sciences, Departments of Physics and Chemical Engineering, University of Texas, Austin, Texas 78712, USA

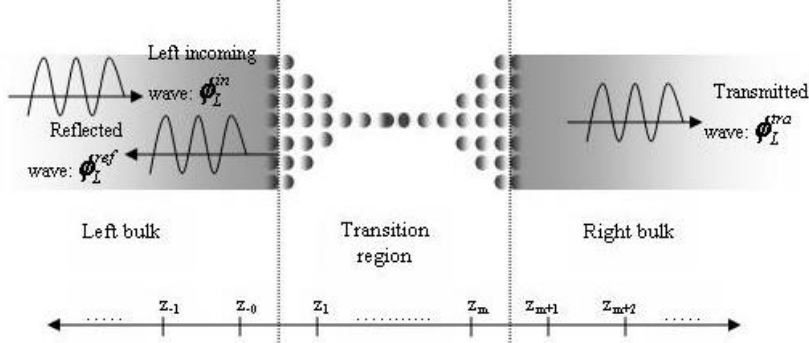


Figure 1: Diagram depicting the physical system of interest (as in [7]). The incoming wave,  $\Phi_L^{in}$ , originating from the left bulk is partially reflected,  $\Phi_L^{ref}$ , back into the left bulk at the boundary of the transition region, and partially transmitted through the transition region, into the right bulk,  $\Phi_L^{tra}$ .

equation that extend over the entire system are considered. The open boundary conditions of this system are modeled by the real-space pseudopotential method [7, 4]. The approach involves solving complex systems of linear equations, derived from a high order finite difference discretization of the Kohn-Sham equation, as part of a self-consistent calculation.

This paper presents a comparative study of a number of complex-valued preconditioners to solve these linear systems. The paper is organized as follows: Section 2 describes the formulation of the discretized model and identify interesting numerical properties inherent in the resulting system. Section 3 introduces the complex-valued preconditioners used in our tests, and Section 4 presents a comparative analysis of their performance on the application problem. A summary of our observations and a conclusion are proposed in section 5.

## 2 Model Formulation

### 2.1 Discretization and Linear System Formation

The central equation for the various DFT-based approaches in molecular transport is the Kohn-Sham equation [3]:

$$-\frac{\nabla^2}{2}\psi(\vec{r}) + V_{eff}\psi(\vec{r}) = \epsilon\psi(\vec{r}), \quad (1)$$

where  $\psi(\vec{r})$  and  $\epsilon$  are the Kohn-Sham orbital and eigenvalue respectively. The effective potential,  $V_{eff}$ , is defined to be

$$V_{eff}(\vec{r}) = v_{ext}(\vec{r}) + v_H(\vec{r}) + v_{xc}(\vec{r}). \quad (2)$$

Here the  $v_{ext}$  is the external potential, typically a sum of the ion-core potential centered at the atoms. The next two terms,  $v_H$  and  $v_{xc}$ , are the Hartree potential and exchange-correlation potential respectively. They have functional dependence on the electron charge density,  $n$ , which can be written as the summation of the squares of the occupied Kohn-Sham orbitals

$$n(\vec{r}) = \sum_i |\psi_i(\vec{r})|^2. \quad (3)$$

The Hartree potential is usually evaluated by solving the Poisson's equation

$$\nabla^2 v_H(\vec{r}) = -4\pi n(\vec{r}). \quad (4)$$

The exchange-correlation potential, however, has no known exact form. We used the Wigner approximation [13]

$$v_{xc} = -0.985n^{1/3} - \frac{0.056n^{2/3} + 0.0059n^{1/3}}{(0.079 + n^{1/3})^2}. \quad (5)$$

Using a higher order finite difference approximation for the Laplacian, this equation can be discretized in real space as [5]

$$\begin{aligned} & -\frac{1}{2} \sum_{n=-N}^N [c_n^x \psi(x_i + nh_x, y_j, z_k) + c_n^y \psi(x_i, y_j + nh_y, z_k) \\ & + c_n^z \psi(x_i, y_j, z_k + nh_z)] + V_{eff}(x_i, y_j, z_k) \psi(x_i, y_j, z_k) \\ & = \epsilon \psi(x_i, y_j, z_k) \end{aligned} \quad (6)$$

where  $N$  is the order of the finite difference expansion. For simplicity, central finite differences are used in the following derivations. The scalars  $c_n^\mu$  ( $\mu = x, y, z$ ) represent the expansion coefficients [6], and  $h_x, h_y$  and  $h_z$  are the grid spacing in  $x, y$  and  $z$  directions respectively.

The system is usually assumed to be periodic in the  $xy$ -plane, parallel to the nanoscale junction, so that the wave functions satisfy the Bloch conditions in the  $x$  and  $y$  directions. Mathematically, one has

$$\begin{aligned} \psi(x + a_x, y, z) &= e^{ik_x a_x} \psi(x, y, z) \\ \psi(x, y + a_y, z) &= e^{ik_y a_y} \psi(x, y, z), \end{aligned} \quad (7)$$

where  $a_x$  and  $a_y$  are the periodicities along  $x$  and  $y$  directions, and  $k_x$  and  $k_y$  are the wavevectors. Along the  $z$ -direction (perpendicular to the nanoscale junction), the current flows from one electrode lead through the molecule, into the other lead. Along this direction the system is aperiodic and Bloch theorem does not hold. The molecular junction, (or transition region), is chosen to be large enough so that the scattering potential is negligible outside it. We then apply matching conditions to the scattering waves requiring the wave functions to approach asymptotically the eigenfunctions of the bulk electrodes.

As suggested by Fujimoto and Hirose [7], we assign the wave functions on the  $xy$ -plane, positioned at  $z_k$ , in a columnar vector,  $\Psi(z_k) = [\psi(x_1, y_1, z_k), \psi(x_2, y_1, z_k), \dots, \psi(x_n, y_n, z_k)]^T$ . In this notation, Eq. (6), can be rewritten in matrix form as

$$(E - \hat{H}) \begin{bmatrix} \Psi(z_0) \\ \Psi(z_1) \\ \vdots \\ \Psi(z_{m+1}) \end{bmatrix} = \begin{bmatrix} B_z^\dagger \Psi(z_{-1}) \\ 0 \\ \vdots \\ B_z \Psi(z_{m+2}) \end{bmatrix} \quad (8)$$

where  $E, \hat{H}$  and  $B_z$  are the corresponding eigenvalue, Hamiltonian matrix and coefficient matrix as in [7]. The  $z$  coordinates  $z_{-1}, z_0$  and  $z_{m+1}, z_{m+2}$  are chosen to be in the left and right bulk regions respectively, while  $z_1, \dots, z_m$  are in the transition region (as shown in figure 1).

Now, consider an incoming wave originating from the left bulk electrode. We analyze the boundary conditions along the  $z$ -direction as follows: Upon reaching the transition region, this incoming wave is partly reflected back into the left lead and partly transmitted into the right lead by the scattering atom (or molecule). The reflected and transmitted waves will asymptotically take the form of a linear combination of the eigenfunctions of the corresponding electrode leads. This result may be formally written as:

$$\Psi(z_k) = \Phi^{in}(z_k) + \sum_l r_l \Phi_l^{ref}(z_k) \quad (9)$$

for the left bulk region, and

$$\Psi(z_k) = \sum_l t_l \Phi_l^{tra}(z_k) \quad (10)$$

for the right bulk region. Here  $\Phi^A$  ( $A=in$  (incident),  $ref$  (reflected) and  $tra$  (transmitted)) are the eigenstates of the leads evaluated on the  $z_k$  plane, and can be obtained by solving a generalized eigenproblem [7]. The scalars  $r_l$  and  $t_l$  are the reflection and transmission coefficients respectively.

Note that since  $z_0$  and  $z_{-1}$  are both in the left region, the wave functions on these two planes satisfy the same Eq. (9). In order to avoid numerical instability (see [7]), we reformulate Eq. (9) by eliminating the reflection coefficients  $r_l$ , to obtain  $\Psi(z_{-1})$  expressed in terms of  $\Psi(z_0)$  as follows:

$$\Psi(z_{-1}) = \Phi^{in}(z_{-1}) + Q(z_{-1})Q^{-1}(z_0)(\Psi(z_0) - \Phi^{in}(z_0)) \quad (11)$$

with  $Q(z_k) = [\Phi_1(z_k), \Phi_2(z_k), \dots, \Phi_{n \times n}(z_k)]$ . Similarly, for the right region, one has:

$$\Psi(z_{m+2}) = Q(z_{m+2})Q^{-1}(z_{m+1})\Psi(z_{m+1}) \quad (12)$$

Substituting Eq. (11) and (12) into Eq. (8), one obtains for left incoming waves (i.e. originating from the left bulk electrode)  $\Phi_L^{in}$ :

$$(E - \hat{H} - \tilde{H}) \begin{bmatrix} \Psi(z_0) \\ \Psi(z_1) \\ \vdots \\ \Psi(z_{m+1}) \end{bmatrix} = \begin{bmatrix} \Omega \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad (13)$$

where  $\tilde{H}$  represents the matrix of boundary conditions in the  $z$  direction. It is essentially a zero matrix, except that it contains two dense blocks in the upper left and lower right regions of the matrix, corresponding to  $B_z^\dagger Q(z_{-1})Q^{-1}(z_0)$  and  $B_z Q(z_{m+2})Q^{-1}(z_{m+1})$ , for the left and right bulk regions respectively. The size of these dense blocks is typically  $\alpha_z n_{xy}$  by  $n_{xy}$ , where  $\alpha_z$  is the order of the finite difference expansion in the  $z$  direction, and  $n_{xy}$  is the number of grid points in one  $z$  slice plane. Here,  $\Omega = B_z^\dagger [\Phi_L^{in}(z_{-1}) - Q(z_{-1})Q^{-1}(z_0)\Phi_L^{in}(z_0)]$ . A similar equation can be obtained for the right incoming waves (i.e. originating from the right bulk electrode).

See [4] for more details on the formulation of the linear system.

## 2.2 Numerical Properties

Equation (13) defines the final form of the linear system we wish to solve. The operator,  $A = (E - \hat{H} - \tilde{H})$ , is a relatively sparse complex matrix except for the two dense blocks contributed by  $\tilde{H}$ . The matrix  $A$  is also not diagonally dominant and the contribution from  $\tilde{H}$  makes it neither symmetric nor hermitian. Moreover, the properties of  $A$  are highly dependent on the value of the incoming wave vector  $k$ . The solution of these types of linear systems for high values of  $k$  has been recognized as one of the most challenging problems for modern numerical techniques [14].

Figure 2 shows the structure of the matrix  $A$ , for the above linear system in Eq. (13). Here, the number of grid points in the  $x$ ,  $y$  and  $z$  directions are 12, 12, and 28 respectively, and a fourth-order finite difference expansion was used for all three directions, resulting in a matrix of size 4032.

In the structure, we observe the dense blocks in the upper-left, and lower-right corners, corresponding to  $\tilde{H}$ . The main diagonal contains the Kohn-Sham energy  $E$  and the local part of the potential. The fourth-order finite difference expansion of the hamiltonian along the  $x$  and  $y$  directions contribute to the block structure along the main diagonal. The block region at the center of the diagonal (and the matrix) represents contributions from the non-local part of the potential. The non-zero regions above and below the main diagonal blocks represent the fourth-order finite difference expansion of the hamiltonian, in the  $z$  direction.

As mentioned earlier, the choice of the wave vector  $k$ , for the incoming wave, dictates the level of difficulty for the numerical solution of the system in Eq. (13). High values of  $k$  tend to shift the spectrum of the matrix to the right and generate more complex eigenvalues, thus making the resulting system very indefinite. The typical range of values for the components of the wave vector is (0.0, 1.0).

Figure 3 shows the eigenvalue spectrum of the matrix in Eq. (13). Figure (3a) shows the

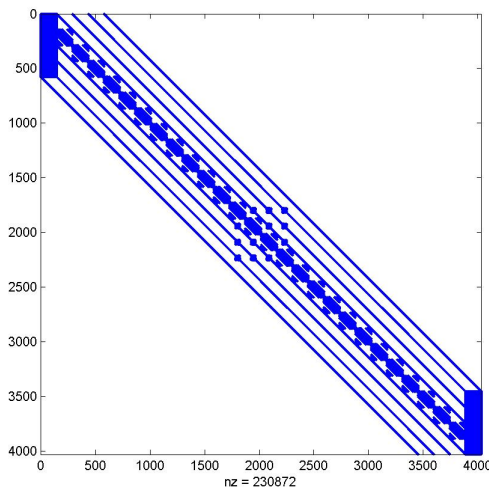


Figure 2: Structure of the matrix in Eq. (13).  $(n_x, n_y, n_z) = (12, 12, 28)$ , and  $(\alpha_x, \alpha_y, \alpha_z) = (4, 4, 4)$  respectively.

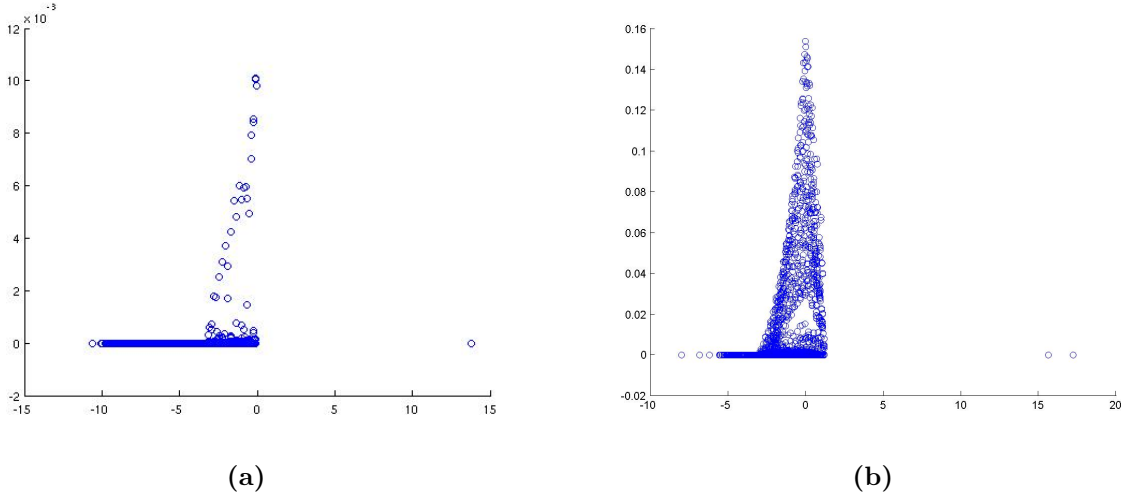


Figure 3: Eigenvalue spectrum of the matrix in Eq. (13). (a) - With wave vector  $k = (0.1, 0.1, 0.1)$ ; (b) - With wave vector  $k = (0.9, 0.9, 0.9)$ .

spectrum of the matrix with wave vector  $k = (0.1, 0.1, 0.1)$ , and figure (3b) shows the spectrum of the matrix with wave vector  $k = (0.9, 0.9, 0.9)$ . From the figures, we observe the shift in the spectrum from left to right, resulting in a greater degree of clustering of the eigenvalues around zero, for the case with the higher wave vector. Furthermore, we observe a greater number of complex eigenvalues in (b), compared to (a). Thus we can clearly see that solving the system with the matrix corresponding to (b) will be more numerically challenging, compared with solving the system with the matrix corresponding to (a).

### 2.3 Related Formulations

The open boundary conditions of this system are often treated by two approaches. One approach makes use of the “nonequilibrium Green function method” [8] combined with DFT. The other approach solves the “Lippmann-Schwinger” [9] equation to obtain the scattering states. In both approaches, the open boundary conditions are incorporated into the Green’s function of the bare electrodes, and used to calculate the electronic structure and transport properties of the system. A number of techniques have been applied to obtain the Green’s function and the scattering states by exploiting different bases [10, 11, 12].

An alternate approach, the one considered in the preceding sections, is to solve the Kohn-Sham equation directly in real space, without any explicit basis. One advantage of this approach is that the open boundary conditions can be easily treated in real space. Moreover, the resulting Hamiltonian is sparse, and efficient sparse solvers can be used. Fujimoto and Hirose recently proposed a real space formalism that models the system [7]. The approach implements a finite difference discretization of the Kohn-Sham equation to obtain the global states and conductance. Furthermore, the approach requires that a large Hamiltonian matrix be inverted in solving for the wave functions. This can be very computationally expensive.

The formulation presented in this paper follows the approach in [7] but avoids direct inversions of the Hamiltonian matrix, by transforming the equations into a set of simultaneous

linear equations.

### 3 Solution Methods

We can rewrite the system (13) using standard notation of linear algebra as

$$Ax = b. \tag{14}$$

Since the original linear system is intrinsically 3-dimensional in nature, it is advisable to avoid the use of direct methods as they tend to be quite expensive both in terms of memory and computations. Preconditioned Krylov subspace methods [16] are considered as the most general-purpose iterative techniques available and offer a good compromise between the robustness of direct solvers and the effectiveness of special purpose procedures such as multigrid. For completeness we begin with a brief background on these methods.

A preconditioned Krylov subspace method for solving the linear system (14) consists of an accelerator and a preconditioner [16]. In what follows we call  $M$  the preconditioning matrix, so that, for example, the right-preconditioned system

$$AM^{-1}y = b \quad \text{where} \quad x = M^{-1}y \tag{15}$$

is solved instead of the original system (14). The above preconditioned system is solved via an “accelerator”, a term used to include a number of methods of the Krylov subspace class. Thus, given an initial guess  $x_0$  to the solution, and  $r_0 = b - Ax_0$  its initial residual, a right-preconditioned Krylov subspace method computes an approximate solution from the affine space

$$x_0 + \text{Span}\{r_0, AM^{-1}r_0, \dots, (AM^{-1})^{m-1}r_0, \} , \tag{16}$$

which verifies certain conditions. For example, the GMRES algorithm [16] requires that the residual  $r_m = b - Ax_m$  has a minimal 2-norm among all possible approximate solutions from the affine Krylov subspace shown above.

#### 3.1 Incomplete LU factorizations

The most common way to define the preconditioning matrix  $M$  is through Incomplete LU factorizations. An ILU factorization is obtained from an approximate Gaussian elimination process. When Gaussian elimination is applied to a sparse matrix  $A$ , a large number of nonzero elements may appear in locations originally occupied by zero elements. These fill-ins are often small elements and may be dropped to obtain Incomplete LU factorizations.

The simplest of these procedures, ILU(0) is obtained by performing the standard  $LU$  factorization of  $A$  and dropping all fill-in elements that are generated during the process. Thus, the  $L$  and  $U$  factors have the same pattern as the lower and upper triangular parts of  $A$  (respectively). More accurate factorizations denoted by ILU(k) have been defined which drop fill-ins according to their ‘levels’ in the elimination process, where the levels attempt to reflect size and are defined recursively [16]. It is most common to use these techniques with levels of fill not exceeding 2 in practice, as higher levels of fill lead to more expensive factorizations, both in terms of storage and computations.

Another class of preconditioners is based on dropping fill-ins according to their numerical values. One of these methods is ILUT (ILU with Threshold). This procedure uses basically a form of Gaussian elimination which generates the rows of  $L$  and  $U$  one by one. Small values are dropped during the elimination, using a parameter  $\tau$ . A second parameter,  $p$ , is then used to keep the largest  $p$  entries in each of the rows of  $L$  and  $U$ . This procedure is denoted by  $ILUT(\tau, lfil)$  of  $A$ .

### 3.2 Multilevel ILUs and ARMS

A related class of preconditioners developed in [18, 24] exploit the idea of multi-level or multi-stage factorization. The original coefficient matrix is symmetrically permuted to the form

$$A = P \begin{pmatrix} B & F \\ E & C \end{pmatrix} P^T \quad (17)$$

in which the  $B$  block is block diagonal. Such a structure is obtained by using independent set - or group-independent set orderings. The Algebraic Recursive Multilevel Solver (ARMS, [18]) exploits such orderings to define multilevel techniques. Thus, at the  $l$ -th level, the coefficient matrix is reordered as in (17) where  $B$  is block-diagonal and then the following block factorization is computed ‘approximately’ (subscripts corresponding to level numbers are introduced):

$$\begin{pmatrix} B_l & F_l \\ E_l & C_l \end{pmatrix} \approx \begin{pmatrix} L_l & 0 \\ E_l U_l^{-1} & I \end{pmatrix} \times \begin{pmatrix} U_l & L_l^{-1} F_l \\ 0 & A_{l+1} \end{pmatrix}, \quad (18)$$

where

$$B_l \approx L_l U_l \quad A_{l+1} \approx C_l - (E_l U_l^{-1})(L_l^{-1} F_l) \quad (19)$$

There are essentially two main steps to the procedure. First, obtain a group-independent set and reorder the matrix in the form (17); second, obtain an ILU factorization  $B_l \approx L_l U_l$  for  $B_l$  and approximations to the matrices  $L_l^{-1} F_l$ ,  $E_l U_l^{-1}$ , and  $A_{l+1}$ . The process is repeated recursively on the matrix  $A_{l+1}$  until a selected number of levels is reached. At the last level, a simple ILUT factorization, possibly with pivoting, or an approximate inverse method can be applied.

The  $A_i$ ’s remain sparse but become denser as the number of levels increases, so small elements are dropped in the block factorization to maintain sparsity. Note that the matrices  $E_l U_l^{-1}$ ,  $L_l^{-1} F_l$  or their approximations  $G_l \approx E_l U_l^{-1}$  and  $W_l \approx L_l^{-1} F_l$  which are used to obtain the Schur complement  $A_{l+1}$  via (19) need not be saved. They are computed only for the purpose of obtaining an approximation to  $A_{l+1}$  and are discarded thereafter to save storage. Subsequent operations with  $L_l^{-1} F_l$  and  $E_l U_l^{-1}$  are performed using  $U_l$ ,  $L_l$  and the blocks  $E_l$  and  $F_l$ .

### 3.3 Use of nonsymmetric permutations

Instead of insisting that the matrix  $B$  in (17) be block diagonal, we can try to reorder it in such a way that the  $B$  block has some “good” numerical properties, specifically of diagonal dominance. The rationale is that a poor  $B$  block will have a negative impact on the rest of the



factorization. ARMS-ddPQ uses nonsymmetric factorizations with this goal in mind. Instead of (17), we now use two permutations, one for the rows and the other for columns:

$$A = P \begin{pmatrix} B & F \\ E & C \end{pmatrix} Q^T. \quad (20)$$

No particular structure is required of  $B$ , but the permutations are selected so that the  $B$  block is as diagonally dominant as possible. The name of the procedure for obtaining the permutation is diagonal-dominance PQ (ddPQ) ordering. The rest of the multi-level procedure is identical with that of ARMS. In other words, the only difference between ARMS-ddPQ and ARMS lies in the permutations used to put the matrix at the  $l$ -th level in  $2 \times 2$  block form.

It is also possible to use nonsymmetric permutations in the context of ILUT. The method known as ILUT with pivoting, see [16], implements a column pivoting ILU, so in effect, we have  $AQ \approx LU$ . In this case, only the columns are permuted. ILUTP is known to be somewhat more robust than ILUT *at the condition to allow substantially more fill-in*.

## 4 Numerical Results

We apply the above preconditioning techniques to solve a test system consisting of a single Fe atom between the two electrodes. A fourth-order finite difference scheme is used for all the directions. The number of grid points in the  $x$ ,  $y$ , and  $z$  planes are 24, 24, and 54 respectively. The resulting system has size 31,104 with 3,577,008 nonzero elements. We allow a maximum of 1500 iterations, with a convergence tolerance of  $1.0 \times 10^{-6}$ . To measure memory cost, we use the *fill-factor* which is defined as the ratio of the number of nonzero entries for the LU factors to the number of nonzero entries in  $A$ . In all cases, the maximum fill factor allowed is 4.0. The accelerator used for all tests is the restarted GMRES, with a restart dimension of  $m = 120$ , which is the maximum dimension of the subspace (16) before restarting the process.

We present results for the system with different values for the wave vector. For each test case, we solve two systems, each corresponding to an orientation of the electron spin - (“spin-up” and “spin-down”). However, we construct the preconditioner only once, and use it to solve both systems each time. The number of iterations and the iteration time is then averaged over these two solves. For the ILUK approach, we chose the level of fill parameter to be = 1, as this was the largest level of fill for which the fill-factor does not exceed 4.

Table 1 shows results of tests involving the ILUK, ILUT, and ARMS-ddPQ preconditioners. The results show a limitation in the performance of the ILUK preconditioner for higher-valued wave vectors. Increasing the level of fill parameter may result in better convergence at the cost of severely compromising memory efficiency. For example, in the above tests, allowing a level of fill parameter = 2 (instead of 1) resulted in a fill factor (memory cost) of  $\approx 8.0$ . The ILUT preconditioner, on the other hand, performed quite well for all three test cases, converging within a reasonable number of iterations and within the required limits of memory cost. However, clearly ARMS-ddPQ outperforms the rest when taking into account both convergence and memory efficiency.

To better understand the performance of the ARMS-ddPQ technique and how it compares with ILUT, we construct a small test problem and show the effect of the two approaches on the

Preconditioner	$k = (k_x, k_y, k_z)$	No. iters	Setup Time (s)	Iter. Time (s)	Fill Factor
ILU(1)	(0.11, 0.11, 0.12)	120	101	11.32	3.18
	(0.55, 0.55, 0.65)	1500	94.50	257.50	3.18
	(0.91, 0.91, 1.00)	1500	96.00	273.50	3.18
ILUT	(0.11, 0.11, 0.12)	14	23.30	2.34	1.71
	(0.55, 0.55, 0.65)	103	69.40	27.10	3.52
	(0.91, 0.91, 1.00)	67	51.30	16.25	3.07
ARMS-ddPQ	(0.11, 0.11, 0.12)	13	30.80	2.41	1.49
	(0.55, 0.55, 0.65)	21	39.80	4.76	2.13
	(0.91, 0.91, 1.00)	12	51.80	2.91	2.69

Table 1: Comparison of preconditioners on application problem with different wave numbers. The number of iterations and the iteration time is averaged over the two realizations of electron spin.

preconditioned matrices. The test problem is of size 3,456, with 244,928 nonzero entries, and the number of grid points in the  $x$ ,  $y$  and  $z$  planes are 12, 12, and 24 respectively. We chose the wave vector to be  $k = (0.55, 0.55, 0.95)$ , and we set the fill factor to be fixed at  $\approx 2.0$ . Table 2 shows a summary of the results. As before, we observe a better convergence performance in terms of iteration count for ARMS-ddPQ. To show the effect of the preconditioners that produced these results, we analyze the spectrum of the preconditioned matrices obtained by applying each of these preconditioners to the above test problem. Our iterative solver approach implements right preconditioning:

$$Ax = b \equiv AM^{-1}y = b, \quad (21)$$

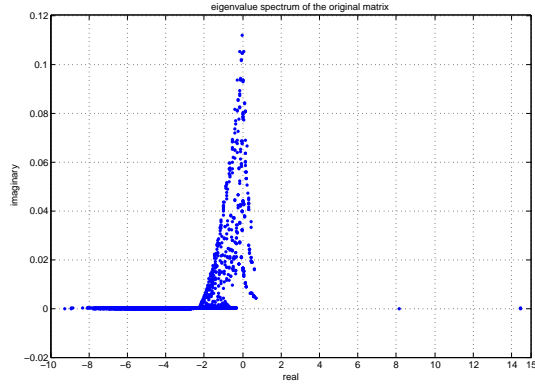
where  $y = Mx$  and  $M$  is the preconditioner. It is known that eigenvalues of preconditioned matrices do not fully explain the convergence behavior of iterative methods, due in part to the effects of high departures from normality [16]. However, in practical situations involving PDEs, a clustering of eigenvalues around one is generally highly desirable.

Figure 4 shows the eigenvalue spectrum of the original matrix (a), the ILUT preconditioned matrix (b), and the ARMS-ddPQ preconditioned matrix (c) respectively. Figure 5 shows the details of the dense cluster regions of the spectrum of the preconditioned matrices.

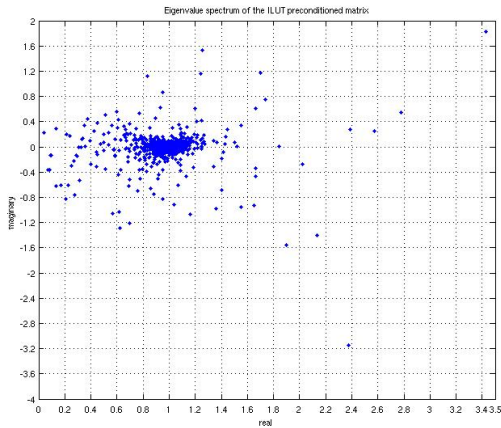
From figure 4 we observe that both preconditioning techniques do a decent job of clustering the eigenvalues around the point  $(1, 0)$ . Blowing up these cluster regions (figure 5), we observe a marked difference between the two spectra. For the ILUT preconditioned matrix, we observe that the majority of the eigenvalues in the dense cluster region are complex - (figure 5a). For the ARMS-ddPQ preconditioned matrix, a significant proportion of the eigenvalues in the dense cluster region lie closer to the real line - (figure 5b). In fact, the clustering appears to be better than that achieved by ILUT, with the additional characteristic that it is along the real line, leading to fewer complex eigenvalues.

Preconditioner	No. iters	Setup Time (s)	Iter. Time (s)	Fill Factor
ILUT	75	0.979	0.819	2.0
ARMS-ddPQ	17	2.49	0.169	2.0

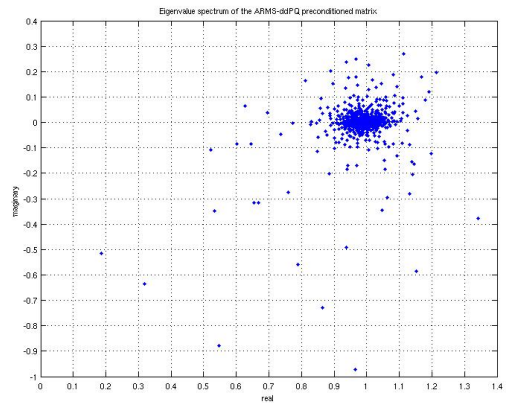
Table 2: Comparison of the ILUT and ARMS-ddPQ preconditioners on a small test problem. Wave number  $k = (0.55, 0.55, 0.95)$ .



(a)



(b)



(c)

Figure 4: Spectra of the preconditioned matrices for the small test problem. (a) Original matrix  $k = (0.55, 0.55, 0.95)$ ; (b) ILUT preconditioned matrix; (c) ARMS-ddPQ preconditioned matrix.

## 4.1 Column Pivoting

In order to alleviate some of the shortcomings of the standard ILUT technique [16], the ILUTP approach implements some form of one-sided nonsymmetric pivoting (usually column partial pivoting) in the construction of the preconditioner. The resulting preconditioner is generally more robust than ILUT, but is less memory efficient since ILUTP tends to generate more fill-in

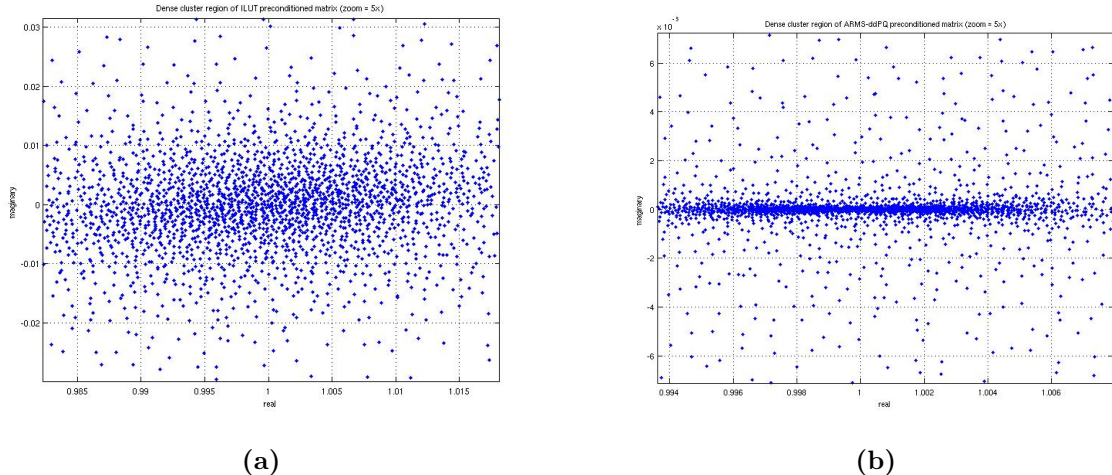


Figure 5: Enlarged dense regions of eigenvalue spectra (magnification = 5x). (a) ILUT preconditioned matrix; (b) ARMS-ddPQ preconditioned matrix.

[16, 20]. We analyze the effect of column pivoting on this application problem by comparing the performance of ILUT with that of ILUTP on a number of test cases.

In figure 6, we show the convergence profiles when solving three different systems using the ILUT and ILUTP preconditioning techniques. Each system is of size 31,104 with 3,577,008 nonzero elements as before. To allow for a good comparison with difficult cases too, we fixed the fill factor at 4.0 for both preconditioners for all test cases. The profile depicts convergence for solving each system corresponding to the first spin direction only (“spin-up”). Figure 6 (a) shows results for a system with  $k = (0.25, 0.25, 0.25)$ , (b) shows results for a system with  $k = (0.7, 0.7, 0.7)$ , and (c) shows the results for a system with  $k = (1.0, 1.0, 1.0)$ .

Interestingly, the results show that the ILUT approach performs better than ILUTP for all three cases. This may be attributed to the fact that performing column partial pivoting can destroy some good structural properties of the matrix that benefit the ILUT technique. Another explanation is that ILUTP requires more memory to work well and limiting the fill-factor to 4, puts it at a big disadvantage. The more indefinite the system is, the greater the effect of pivoting, and we observe convergence for ILUT in less than half the number of iterations for the ILUTP approach. Also, note that the number of iterations for each of these test cases was directly proportional to the iteration time for convergence.

## 4.2 Symmetric and Nonsymmetric Permutation

To examine the effect of symmetric and nonsymmetric permutations on our system, we compare the performance of standard (symmetric) ARMS and ARMS-ddPQ on the same three test cases as in the preceding example.

Figure 7 shows the convergence profiles of solving the three systems using the standard ARMS and ARMS-ddPQ preconditioning techniques. Note that the results for the last case, 7(c), has been truncated to show only up to 100 iterations, since standard ARMS failed to converge after the 1500 maximum allowed iterations. The results indicate that implementing a

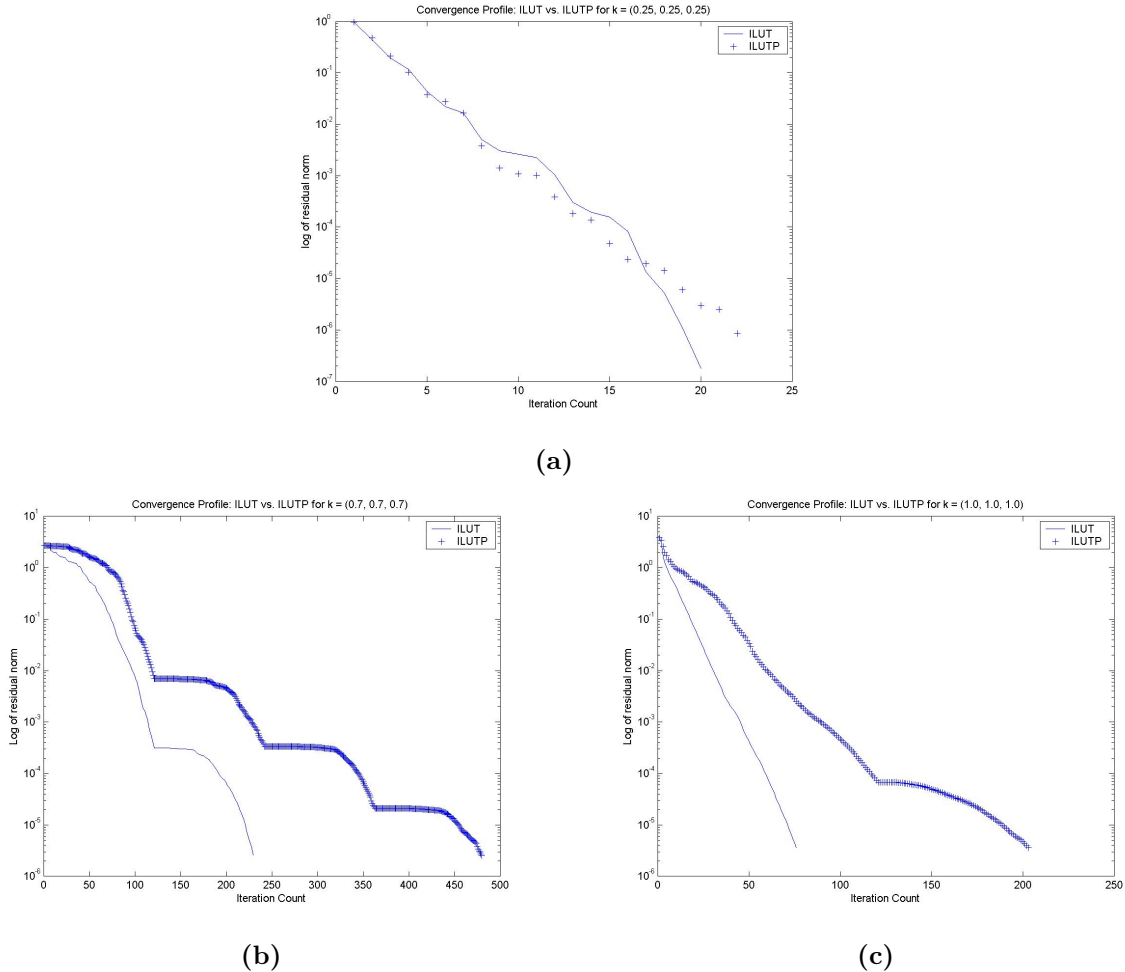


Figure 6: Convergence Histories of ILUT and ILUTP - Iteration Count vs. Residual Norm. (a) System with  $k = (0.25, 0.25, 0.25)$ ; (b) System with  $k = (0.7, 0.7, 0.7)$ ; (c) System with  $k = (1.0, 1.0, 1.0)$ .

two-sided nonsymmetric permutation greatly improves performance, particularly for the most difficult cases. Here too, the number of iterations was directly related to the iteration time for convergence.

In our final test, we further examine the robustness of the preconditioning techniques by comparing them based on their convergence properties with respect to solving the two systems corresponding to the two electron spins. In what follows, we construct the preconditioner once for the system corresponding to the first spin, and use it to solve both systems. The wave vector for this test case is  $k = (0.7, 0.7, 0.7)$ . Figure 8 shows the results for: (a) ILUT, (b) ILUTP, (c) Standard ARMS, and (d) ARMS-ddPQ. Ideally, we would expect the effect of spin on the performance of the preconditioners to be minimum, since the two systems merely differ from each other by virtue of the direction (i.e. sign) of the electron spin. The results are formally summarized in table 3.

These results indicate that the standard ARMS and ARMS-ddPQ preconditioners exhibited minimum sensitivity to the effect of spin. This may be attributed to their multilevel nature. The

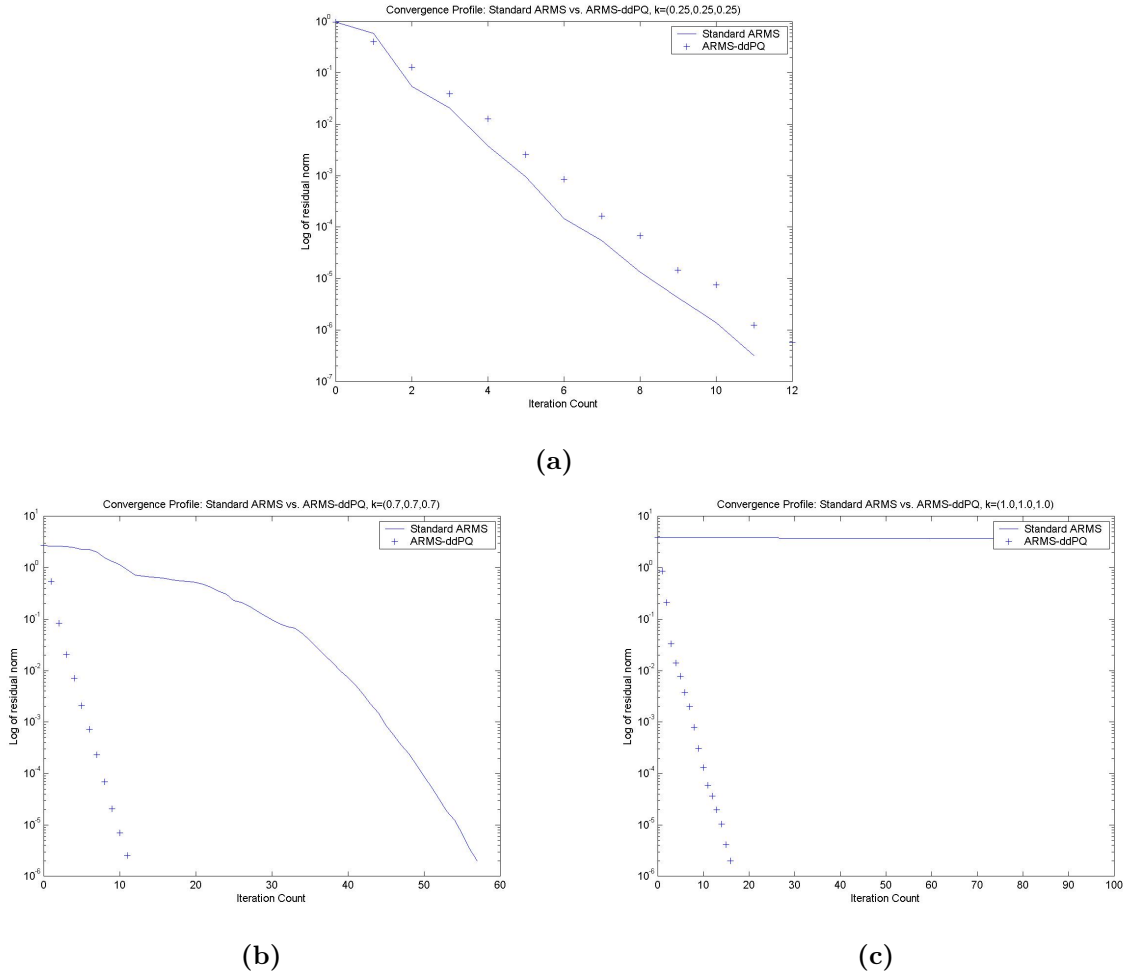


Figure 7: Convergence Histories of standard ARMS and ARMS-ddPQ - Iteration Count vs. Residual Norm. (a) System with  $k = (0.25, 0.25, 0.25)$ ; (b) System with  $k = (0.7, 0.7, 0.7)$ ; (c) System with  $k = (1.0, 1.0, 1.0)$ .

difficulty for standard numerical solvers to solve a particular problem depends largely on the properties of the model. By their nature, multilevel methods automatically build smaller scale models, that exploit the “nice” properties of the linear system, and that could be aggregated to solve the original problem. This key feature of multilevel methods allows the performance of the standard ARMS and ARMS-ddPQ preconditioning techniques to be much less sensitive

Preconditioner	Spin 1 (“spin-up”)	Spin 2 (“spin-down”)
ILUT	228	284
ILUTP	476	583
Standard ARMS	57	59
ARMS-ddPQ	11	13

Table 3: Table of results for the number of iterations required to solve the two systems corresponding to each electron spin direction.

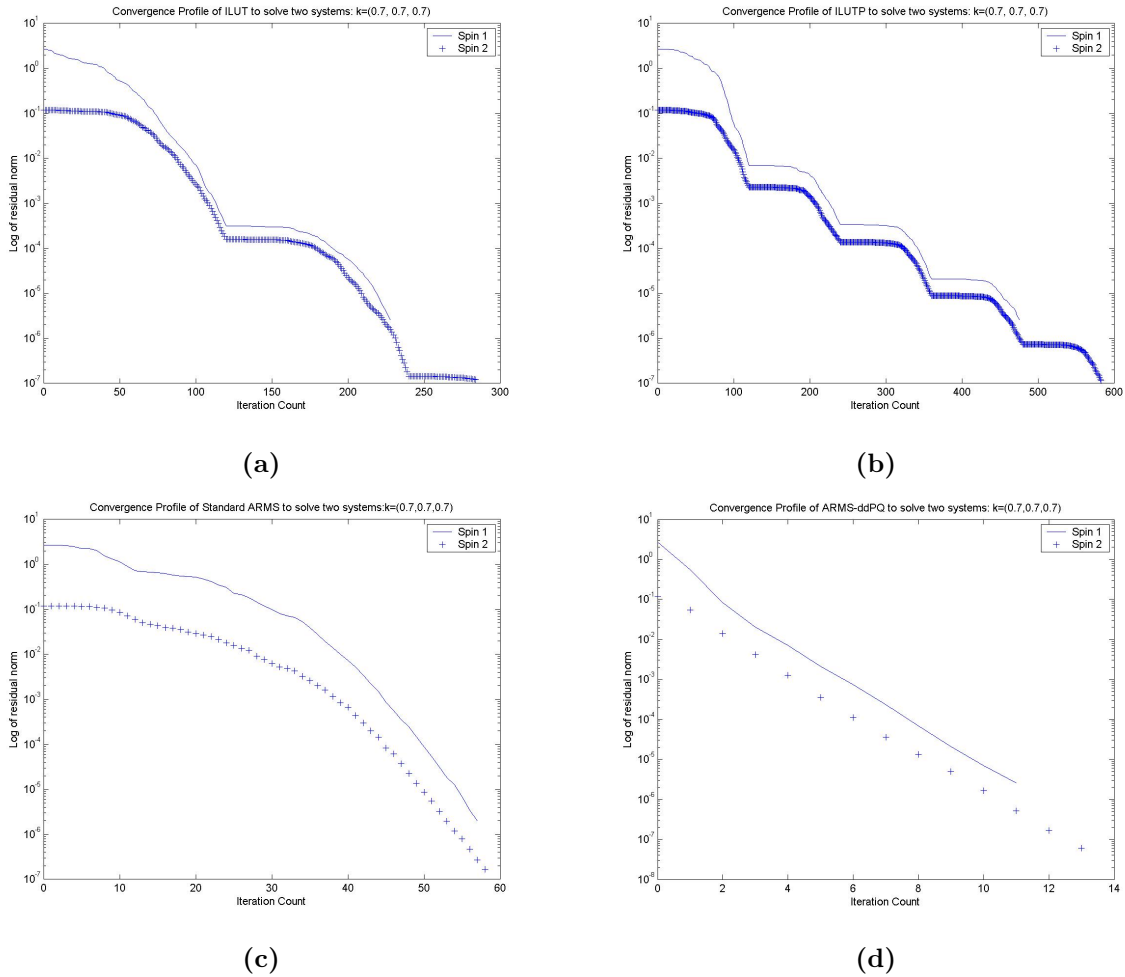


Figure 8: Convergence Histories for solving two systems corresponding to each electron spin direction, with each preconditioner constructed only once -  $k = (0.7, 0.7, 0.7)$ . Iteration Count vs. Residual Norm. (a) ILUT; (b) ILUTP; (c) Standard ARMS; (d) ARMS-ddPQ

to small changes due to spin, than the other techniques.

The results seem also to indicate that multilevel-type methods are less sensitive to small changes in the system. From figures 6, 7, and 8 we see a marked difference in the shape of the profiles for the ILUT and ILUTP methods, compared with multilevel methods. The ILUT and ILUTP techniques exhibit a stepped profile with a frequency of 120 which is the restart dimension for GMRES. The multilevel methods converge before reaching the restart dimension of 120, and so this phenomenon is not observed in the corresponding convergence curves.

## 5 Conclusion

In this paper, we have analyzed the performance of some ILU-based preconditioners to solve highly indefinite systems that result from an application in molecular transport. Among the preconditioners tested, we have shown that the ARMS-ddPQ technique performs best, in terms

of storage and computational cost. The method builds the preconditioning matrix from a multilevel ILU approach, combined with a diagonal-dominance criterion for permuting the matrix in a nonsymmetric way. ARMS-ddPQ implements dropping techniques to reduce fill-in and preserve sparsity. It also includes a number of parameters that can be adjusted to obtain a robust preconditioner for the particular problem at hand.

For most indefinite problems, performing some form of nonsymmetric permutation [21, 23, 22] during the incomplete factorization tends to improve the performance of the numerical solver. Two such approaches are the ILUT factorization with partial pivoting, and ARMS with nonsymmetric permutations (i.e. ddPQ ordering). For the problems in the application studied in this paper, one-sided column partial pivoting techniques such as ILUTP resulted in a worse performance compared with standard ILUT. However, within a multilevel approach, such as ARMS, nonsymmetric permutations can be very beneficial. Thus, a two-sided nonsymmetric permutation approach, such as ARMS with ddPQ ordering, showed significant improvement in performance over standard ILU methods and standard ARMS (with symmetric permutations).

The standard version of ARMS (symmetric permutations) performed better than ILUT and ILUTP for the cases with low and average difficulty, but failed to converge for the most difficult problems. This may be attributed to the implementation of symmetric permutations to reorder the matrix. For highly indefinite systems, this symmetric reordering can result in a preconditioned matrix with poor properties.

## References

- [1] M.C. Petty, M.R. Bryce and D. Bloor(Eds.), Introduction to molecular Electronics, Oxford University Press , New York, 1995.
- [2] H. Ohnishi, Y. Kondon and K. Takayanagi, Nature, **398**, 780 (1998).
- [3] P. Hohenberg and W. Kohn, Phys. Rev. **136**, B864 (1964); W. Kohn and L.J. Sham, Phys. Rev. **140**, A1133 (1965).
- [4] L. Kong, M.L. Tiago and J.R. Chelikowsky, Phys. Rev. B, **73**, 195118 (2006).
- [5] J.R. Chelikowsky, N. Troullier, and Y. Saad, Phys. Rev. Lett. **72**, 1240 (1994).
- [6] B. Fornberg and D.M. Sloan, in *Acta numerica* **94** , edited by A. Iserles, Cambridge University Press, Cambridge, UK, 1994.
- [7] Y. Fujimoto and K. Hirose, Phys. Rev. B, **67**, 195315 (2003).
- [8] S. Datta, Electronic Transport in Mesoscopic Systems (Cambridge University Press, Cambridge, UK, 1997).
- [9] N. D. Lang, Phys. Rev. B, **52**, 5335 (1995).
- [10] H. Ness and A. J. Fisher, Phys. Rev. B, **56**, 12469 (1997).
- [11] P. S. Damle, A. W. Ghosh, and S. Datta, *ibid*, **64**, 201403(R) (2001).



- [12] J. J. Palacios, A. J. Perez-Jimenez, E. Louis, and J.A. Verges, *ibid*, **64**, 115411 (2001).
- [13] E. P. Wigner, *Phys. Rev.* **46**, 1002 (1934).
- [14] O. Zienkiewicz, Achievements and some Unsolved Problems of the Finite Element Method. *Int. J. Numer. Mthds. Eng.* **47**, 9-28 (2000).
- [15] R. Kechroud, A. Soulaïmani, Y. Saad and S. Gowda, Preconditioning techniques for the solution of the Helmholtz equation by the finite element method. *Math. Comput. Simul.* **65**, 4-5 (2004), 303-321.
- [16] Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd edition. SIAM, Philadelphia, PA, (2003)
- [17] Z. Li, Y. Saad and M. Sosonkina, pARMS: A parallel version of the Algebraic Recursive Multilevel Solver. *nlaa*, **10**, (2003) 485-509
- [18] Y. Saad, B. Suchomel, ARMS: An Algebraic Recursive Multilevel Solver for general Sparse Linear Systems. *nlaa*, **9**, (2002)
- [19] M. Sosonkina, Y. Saad and X. Cai, Using the parallel Algebraic Recursive Multilevel Solver in modern physical applications. *Future generation Comoptur Systems*, **20**, (2004), 489-500
- [20] D. Osei-Kuffuor and Y. Saad, A Comparison of Preconditioners for Complex Valued Matrices, *submitted to nlaa: NLA-07-0029*.
- [21] M. Benzi, J. C. Haws, and M. Tuma, Preconditioning Highly Indefinite and Nonsymmetric Matrices. *Siam Journal on Scientific Computing*, **22**, 4(2001), 1333-1353
- [22] I.S. Duff, and J. Koster, On algorithms for permuting large entries to the diagonal of a sparse matrix. *SIAM Journal on Matrix Analysis and Applications*, **22**, 4(2001), 973-996
- [23] M. Olschowka, and A. Neumaier, A new pivoting strategy for Gaussian elimination. *Linear Algebra and its Applications*, **240**, 1-3(1996), 131-151
- [24] Y. Saad, Multilevel ilu with reorderings for diagonal dominance. *SIAM J. Sci. Comput.*, **27**, 3(2005), 1032-1057