Recycling Preconditioners for Sequences of Linear Systems and Matrix Reordering

Ming Li

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Mathematics

Eric de Sturler, Chair
Christopher A. Beattie
Serkan Gugercin
Tao Lin

December 8, 2015
Blacksburg, Virginia

Keywords: sequence of linear systems, updating preconditioners, inexact Krylov subspace methods, matrix reordering

Recycling Preconditioners for Sequences of Linear Systems and Matrix Reordering

Ming Li

(ABSTRACT)

In science and engineering, many applications require the solution of a sequence of linear systems. There are many ways to solve linear systems and we always look for methods that are faster and/or require less storage. In this dissertation, we focus on solving these systems with Krylov subspace methods and how to obtain effective preconditioners inexpensively.

We first present an application for electronic structure calculation. A sequence of slowly changing linear systems is produced in the simulation. The linear systems change by rank-one updates. Properties of the system matrix are analyzed. We use Krylov subspace methods to solve these linear systems. Krylov subspace methods need a preconditioner to be efficient and robust. This causes the problem of computing a sequence of preconditioners corresponding to the sequence of linear systems. We use recycling preconditioners, which is to update and reuse existing preconditioner. We investigate and analyze several preconditioners, such as ILU(0), ILUTP, domain decomposition preconditioners, and inexact matrix-vector products with inner-outer iterations.

Recycling preconditioners produces cumulative updates to the preconditioner. To reduce the cost of applying the preconditioners, we propose approaches to truncate the cumulative preconditioner updates, which is a low-rank matrix. Two approaches are developed. The first one is to truncate the low-rank matrix using the best approximation given by the singular value decomposition (SVD). This is effective if many singular values are close to zero. If not, based on the ideas underlying GCROT and recycling, we use information from an Arnoldi recurrence to determine which directions to keep. We investigate and analyze

their properties. We also prove that both truncation approaches work well under suitable conditions.

We apply our truncation approaches on two applications. One is the Quantum Monte Carlo (QMC) method and the other is a nonlinear second order partial differential equation (PDE). For the QMC method, we test both truncation approaches and analyze their results. For the PDE problem, we discretize the equations with finite difference method, solve the nonlinear problem by Newton's method with a line-search, and utilize Krylov subspace methods to solve the linear system in every nonlinear iteration. The preconditioner is updated by Broyden-type rank-one updates, and we truncate the preconditioner updates by using the SVD finally. We demonstrate that the truncation is effective.

In the last chapter, we develop a matrix reordering algorithm that improves the diagonal dominance of Slater matrices in the QMC method. If we reorder the entire Slater matrix, we call it global reordering and the cost is $O(N^3)$, which is expensive. As the change is geometrically localized and impacts only one row and a modest number of columns, we propose a local reordering of a submatrix of the Slater matrix. The submatrix has small dimension, which is independent of the size of Slater matrix, and hence the local reordering has constant cost (with respect to the size of Slater matrix).

This material is based upon work supported by the National Science Foundation under Grant No. NSF 1025327 and NSF 1217256. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Dedication

To my loving parents, Xiujuan and Jinyu.

Acknowledgments

I would very much like to express my deepest gratitude to my advisor, Dr. Eric de Sturler. His vast knowledge and consistent guidance have helped me greatly in my entire study for the past five years. He patiently assisted me in my writing, financially supported my research, and inspired me through challenging problems. I would never have been able to finish my dissertation without him.

Besides my advisor, I would like to thank the rest of my committee, Dr. Christopher Beattie, Dr. Serkan Gugercin, and Dr. Tao Lin. Their time, encouragement, and suggestions are very important and mean a lot to me. I would also like to thank Dr. Tao Lin for teaching me numerical analysis.

It has been a great pleasure to study and work in the Math department. I am grateful to all the wonderful teachers and mentors I have had in the Math Department. I would like to thank Eileen Shugart, Rachel Arnold, Jessica Schmale for their time, advice, and suggestions that greatly helped me on my teaching. Many thanks to Nicole Sutphin, Tammi Johnston, Ken Hinson, Bill Reilly and Benjamin Williams for their time and generous help through my study program. I am also thankful to Dr. Peter Haskell for the teaching assistant support and for providing me travel fund for conferences.

All my fellow students and friends make my life in Blacksburg pleasant and colorful. I would like to express my special thanks to my fellow students and friends, Arielle McNally and Vishwas Rao, for the discussions that inspire me on my research and for the time that we

were working together before the deadlines. I thank my fellow students Guangyue Gao, Shuo Wang for the friendship and suggestions. I also thank Chen-Chi Shing, Steven, Sheng Xie, Jianxiang Zhang, and Sheng Zhou for the wonderful time we have spent together.

Finally, I would like to thank my parents, two elder brothers. They were always supporting me and encouraging me with their best wishes.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In science and engineering, many applications require the solution of a sequence of linear systems of the form

$$A_i x_i = b_i, \quad i = 0, 1, 2, \ldots, \tag{1.1}$$

where the A_i's are $N \times N$ nonsingular matrices, b_i's are right-hand side vectors and A_i, b_i change from one system to the next as the physical system evolves. Some examples of these applications include nonlinear partial differential equations [11, 78], large scale electromagnetism [36], quantum Monte Carlo methods [2, 4, 6, 65], optimization problems, topology optimization and other optimal design problems, applications involving one or more parameters, and inverse problems. To solve the sequence of linear systems, Krylov subspace methods especially with recycling, have become a popular technique and people might adapt the Krylov solver according to the problem at hand [66, 68]. In order to be efficient and robust, Krylov subspace methods need to be preconditioned. This causes the problem of computing a sequence of preconditioners corresponding to the sequence of linear systems. For example, an initial preconditioner P_0 needs to be computed for solving $A_0 x_0 = b_0$. In general, a preconditioner P_i needs to be obtained for solving $A_i x_i = b_i$.

However, computing preconditioners $P_0, P_1, P_2, \ldots$ for every single system separately can be very expensive and impractical. There is a strong need to reduce the overall cost of

preconditioning these linear systems. Using the same preconditioner for all the linear systems is an alternative. However, this hampers the convergence process and also may increase the total cost. Therefore, most effective approaches adopt a middle ground by updating the existing preconditioner and reusing it for successive linear systems [36, 11, 2, 66], which we can call recycling preconditioners.

In Ahuja et al. [2, 1], preconditioners are multiplied by cumulative inverses of rank-one updates. Sherman–Morrison formula is used to obtain the inverse of rank-one or low-rank updates. Similarly in Bergamaschi et al. [11], Broyden-type rank-one updates are used to update the preconditioners. These are two common ways to update the existing preconditioner and reuse it for successive linear systems. The advantage of these approaches is that they are cheaper than computing a new preconditioner for each system and more effective than using the same preconditioner for all linear systems. However, due to the cumulative updates to the preconditioner, the cost of applying the preconditioner increases. At some point, the cost of applying a preconditioner is even higher than computing a new preconditioner. Therefore, in Ahuja et al. [2, 1] and Bergamaschi et al. [11], a new preconditioner is computed from scratch periodically.

One of our contribution is to reduce the cost of multiplying by preconditioners by truncating low-rank preconditioner update into a smaller rank matrix. We develop two truncation approaches to reduce or even bound the cost of applying the preconditioners. Therefore, we can postpone or even avoid the recomputation of a new preconditioner.

Another contribution in this dissertation is focused on the application of this approach to a Markov Chain Monte Carlo (MCMC) method. We analyze the properties of the singular values of Slater matrices occurring in the QMC method. We also test and analyze the effect of several new preconditioning techniques for this application, such as domain decomposition preconditioners and inexact matrix-vector product by inner-outer GMRES iterations. Although these preconditioners are effective they appear to be too expensive for QMC, producing fast convergence with relatively high cost. The goal of recent works on the QMC

method is to obtain $O(N^2)$ cost.

The last contribution is the accumulated local reordering scheme to improve the diagonal dominance of system matrices. We develop a new reordering algorithm that is a combination of popular algorithms. This dissertation is structured as follows.

In Chapter 2, we introduce and explain the application for simulating electronic structure systems with QMC methods, and analyze a few properties of the Slater matrices arising in the QMC method. In Chapter 3, three preconditioners are analyzed and tested.

In Chapter 4, we propose and analyze two approaches to reduce the cost of applying preconditioners with a sequence of low-rank updates.

In Chapter 5, a reordering algorithm is developed to improve the diagonal dominance of the Slater matrix. To obtain a cheap and effective reordering, we also develop a local reordering scheme.

A Few Preconditioners for the Quantum Monte Carlo Method

QMC method is a large class of computational algorithms. The QMC method produces a sequence of linear systems. The system matrix is called a Slater matrix. The main cost of many QMC methods, for example, the variational Monte Carlo (VMC) method, consists of two parts. The first part is to construct a sequence of Slater matrices. In Alfé and Gillian [4, 5], William [86], the Slater matrix is optimally sparse by optimizing the orbitals of the physical system (for insulators). An appropriate type of basis functions is chosen to construct the Slater matrix [34, 35]. Therefore only a linear number of elements of the Slater matrix is filled and updated, and the cost of constructing a Slater matrix is $O(N)$. These methods are referred to as linear scaling methods. The second part is to compute determinant ratios of successive Slater matrices (we explain why we need to compute this in Chapter 2). Historically, the cost is $O(N^3)$ by using the exact inverses of Slater matrices [17]. In Ahuja et al. [2], the cost is reduced to $O(N^{2.19})$ by introducing iterative solvers,

approximate preconditioning technique, and updates as well as regular reorderings. But still, the cost of computing determinant ratios dominates.

To obtain effective and cheap approaches, we investigate several new preconditioners. First, the popular incomplete LU preconditioner is tested. The second preconditioner is a domain decomposition preconditioner. When the Slater matrix is updated by a rank-one update, only one row of the Slater matrix changes. The change is a small and local change to both physical system and the Slater matrix. So, we consider the entire domain divided into two subdomains and employ domain decomposition preconditioner. Since we attempt to move a particle at every Monte Carlo step, the first subdomain consists of the neighbor particles and orbitals of the moving particle. The second subdomain is the remaining particles and orbitals. We mainly test two-level domain decomposition preconditioner.

The last preconditioner is Schur complement based domain decomposition method with inexact matrix-vector products. Generally, inexact preconditioning is a scenario when the preconditioner requires a linear solution with a second iterative method. In our case, the Slater matrix A is divided as $\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}$ using domain decomposition methods. GMRES is used twice. As an outer iteration, we use GMRES to solve the Schur complement system:

$$\left(A_{11} - A_{21}A_{22}^{-1}A_{12}\right) z_1 = b_1,$$

where further details are explained in Chapter 3. An inner iteration with GMRES is used to approximate A_{22}^{-1}. A theoretical analysis shows that the inexact matrix-vector products with inner and outer GMRES work well under conditions that are generally easy to satisfy.

Truncation of Preconditioner Updates

In this dissertation, we are concerned about successive solutions to the linear systems where the matrices are updated by low-rank updates [26]. In other words, the system matrices are updated by multiplying with $I + UV^T$, where I is an identity matrix, and U and V are

low rank matrices. Using a single preconditioner for all the linear systems causes significant deterioration in convergence. The aforementioned systems can be effectively preconditioned by constructing a good initial preconditioner and subsequently performing low rank updates to it based on the Sherman-Morrison-Woodbury formula [3, 63, 69].

This idea of 'recycling' preconditioners is used by Ahuja et al. [2] and Bergamaschi et al. [11].

At every Monte Carlo step, we solve the system $A_i x_i = b_i$ iteratively by GMRES. From one system to the next, one row of the Slater matrix A is changed. Let the row index be i_k and the difference vector is q_i. The change is a rank-one update and the relationship between two consecutive Slater matrices is

$$\begin{aligned} A_{i+1} &= A_i + e_{i_k} q_i^T, \\ &= A_i (I + A_i^{-1} e_{i_k} q_i^T), \quad i = 0, 1, 2, \cdots . \end{aligned} \tag{1.2}$$

Because the Slater matrix changes by rank-one updates, the preconditioner needs to be updated by inverses of rank-one updates [44]. Let P_0 be a good initial preconditioner and we subsequently update P_0 by multiplying it with the inverse of each low-rank update, $P_{i+1} = (I + A_i^{-1} e_{i_k} q_i^T)^{-1} P_i, i = 0, 1, \cdots$. This produces the accumulated updates of the preconditioner, $P_{i+1} = \prod_{j=0}^{i} (I + A_i^{-1} e_{i_k} q_i^T)^{-1} P_0$. As a result, this sequence of preconditioners is such that

$$A_0 P_0 = A_1 P_1 = \cdots = A_i P_i = \cdots . \tag{1.3}$$

Since P_0 is a good preconditioner, the same convergence is maintained for all systems as a result of (1.3). However, a cost issue arises when applying the preconditioners. Because $P_{i+1} = \prod_{j=0}^{i} (I + A_i^{-1} e_{i_k} q_i^T)^{-1})^{-1} P_0$, the total cost of multiplying P_{i+1} is $4iN$ per preconditioned matrix-vector product. So, the cost over i steps in the QMC method or i Newton iterations for nonlinear system, even with good preconditioners resulting in a roughly constant number of iterations, is $O(i^2 N)$. If we can bound the rank of "the total preconditioner update", the cost will be $O(irN)$, where r is fixed. Therefore, we propose to truncate the

accumulated updates to reduce the rank and hence reduce the cost of applying the preconditioner. In other words, if the number of linear systems becomes significant, we will truncate $\prod_{j=0}^{i}(I + A_i^{-1}e_{i_k}q_i^T)^{-1})^{-1}$ into a lower rank matrix. If we truncate at k-th system (Monte Carlo step), we have $A_kP_k \approx A_0P_0$. The convergence of the preconditioned system A_kP_k may not be as good as that of A_0P_0. So, we aim to truncate such that the convergence is not affected drastically and the cost of applying the preconditioners is reduced, and we provide theoretical analysis.

We propose two approaches for truncating the accumulated updates. The main goal of both approaches is to reduce the cost of applying the preconditioners to $O(N)$ while keeping relatively good convergence. The first approach is to truncate the accumulated preconditioner update using the best approximation given by the singular value decomposition (SVD). This is effective if many singular values are close to zero. Unfortunately, this is not always the case. The second approach, based on the ideas underlying GCROT and recycling, uses the Arnoldi (or Lanczos) recurrence to determine which directions to keep [22]. Details of these procedures follow in the subsequent sections.

Numerical results have demonstrated the effectiveness of the truncation approaches.

Matrix Reordering

Some preconditioners require diagonal dominance of the system matrix [39, 58, 40]. For example, the incomplete LU (ILU) preconditioner works well when the matrix is diagonally dominant or close to diagonal dominant [80]. Further, if the matrix is far from diagonal dominant, the ILU algorithm is unstable.

There are many heuristic techniques to improve the diagonal dominance of a matrix [2, 9, 10, 24, 25]. Some approaches are to maximize the diagonal and some are to maximize the trace. This leads to an interesting problem in graph theory and combinatorial optimization, that is, we want to find the maximum matching or the maximum weighted matching. The

most popular algorithm to find the maximum matching is max–flow method. There are a couple of options for maximum weighted matching, such as linear programming, the Ford-Fulkerson algorithm, the Edmonds-Karp algorithm, the Hungarian algorithm, and the push-relabel algorithm. In our experiments, we use the Hungarian algorithm and the push-relabel algorithm.

However, simply maximizing the diagonal or trace may still lead to matrices with one or even more zeros on the diagonal. Zeros on the diagonal may deteriorate the incomplete LU preconditioner. Therefore, we develop a reordering algorithm which is a combination of maximizing both the diagonal and the trace. We first maximize the minimum absolute value for all diagonal elements, and then maximize the trace while maintaining a lower bound based on the minimum absolute value. In practice, we preset a cutoff value on the diagonal to be the lower bound.

We test our reordering on Slater matrices arising in the QMC method. Due to the local changes in the Slater matrix, we have two options. We can apply this reordering algorithm either locally or globally. We prefer frequent local reordering because it is very cheap, that is, $O(1)$ cost per reordering. The global reordering is carried out much less often, normally a few times or just once for every N linear systems. We also present the results of our reordering.

For simplicity, in this dissertation, $\|\cdot\|$ refers to the 2-norm, unless otherwise stated.

Chapter 2

The Quantum Monte Carlo Method

QMC is a class of computational algorithms. Each algorithm employs, in one way or another, the Monte Carlo method to handle the many-dimensional integrals that arise. The origins of QMC methods are often attributed to Fermi and Richtmyer, who developed a mean field particle interpretation of neutron-chain reactions [27]. However, the first generic type particle algorithm for estimating ground state energies of quantum systems (in reduced matrix models) is due to Jack Hetherington [46]. Now QMC methods are commonly used in condensed matter physics [79, 71], molecular chemistry [45, 70], and computational biology, etc. In our main application problem, the QMC method is utilized to solve a quantum many-body problem [2, 1, 30, 14]. We first explain the main problem formulation. Our discussion is based on [2, 14].

Let N be the number of particles (electrons) in the system, which also equals the number of orbitals (nuclei). Then N is the size of the many-body system. Suppose all particles and orbitals are spaced in a body centered cubic (b.c.c.) lattice, positions of particles are given by r_i's, and positions of orbitals are given by $\mathbf{O}_j$'s, $1 \leq i, j \leq N$. We represent the coordinates of all particles and orbitals in the system by a vector R. For simplicity, we ignore the spin.

The main goal of QMC here involves sampling over configurations R with the probability

density induced by the many-body wave function. Because the wave function is highly localized, direct sampling is not efficient. Therefore we utilize Markov Chain Monte Carlo algorithm (MCMC) with Metropolis algorithm [8, 17, 60, 30]. The MCMC algorithm samples configurations as follows. At every step, the current configuration R is changed by moving one particle with a random displacement, generating a trial configuration R'. By Metropolis algorithm, we accept the trial configuration with a transition probability. The transition probability needs to be computed at every step, which is called acceptance/rejection test. For each independent configuration R generated from the Markov Chain, the local energy is calculated and averaged. In the rest of this section, we will further detail the separate pieces of this process.

2.1 Wave Function and Slater Matrix

We denote the many body trial wave function as $\Psi_\alpha(R)$, where α is a vector of variational parameters that needs to be optimized, $\alpha = [\alpha_1, \alpha_2, \cdots, \alpha_s]$. Since the optimization of these parameters is not our main concern, we will ignore α in later sections in this dissertation.

Each orbital is a single particle wave function, and the specific details of the single particle wave functions depend sensitively on the exact material being simulated. There are many choices to represent the single particle wave function [16, 4, 49]. In our experiments, we use a set of Gaussian functions as representative insulator single particle orbitals [17]. We also ignore spin for simplicity. For any orbital (nucleus) located at $\mathbf{O}_j$, its wave function is presented by

$$\phi_j(\mathbf{r}) = e^{-K\|\mathbf{r}-\mathbf{O}_j\|^2}, j = 1, \cdots, N,$$

where $\mathbf{r}$ is a particle (electron) position, and K is parameter of the Gaussian functions. K describes the decay rate of the Gaussian functions and we typically set $K = 1$. With Gaussian functions as basis functions, we get the following system matrix which is called

Slater matrix. Let particles positions be $r_1, r_2, \cdots, r_N$. We define the Slater matrix as

$$A = \begin{pmatrix} \phi_1(r_1) & \phi_2(r_1) & \phi_3(r_1) & \dots & \phi_N(r_1) \\ \phi_1(r_2) & \phi_2(r_2) & \phi_3(r_2) & \dots & \phi_N(r_2) \\ \phi_1(r_3) & \phi_2(r_3) & \phi_3(r_3) & \dots & \phi_N(r_3) \\ \vdots & & & \ddots & \vdots \\ \phi_1(r_N) & \phi_2(r_N) & \phi_3(r_N) & \dots & \phi_N(r_N) \end{pmatrix}.$$

Various forms can be used to represent the system wave function and the most popular form is

$$\Psi(r_1, r_2, \dots, r_n) = \mathcal{J}(r_1, r_2, \dots, r_n) \det\left(A(r_1, r_2, \dots, r_n)\right), \tag{2.1}$$

where A is the Slater matrix and $\mathcal{J}(r_1, r_2, \dots, r_n)$ is the Jastrow factor. The Jastrow factor is not expensive to compute, and we do not consider it in this thesis. The square of the wave function, Ψ^2, provides the probability density. Therefore, the transition probability from one configuration R to the next is the squared ratio between two wave functions, which is $\left|\frac{\Psi(R')}{\Psi(R)}\right|^2$. Therefore, in the acceptance/rejection test, the probability to accept a particle move is determined by $\min\left(1, \quad \left|\frac{\Psi(R')}{\Psi(R)}\right|^2\right)$.

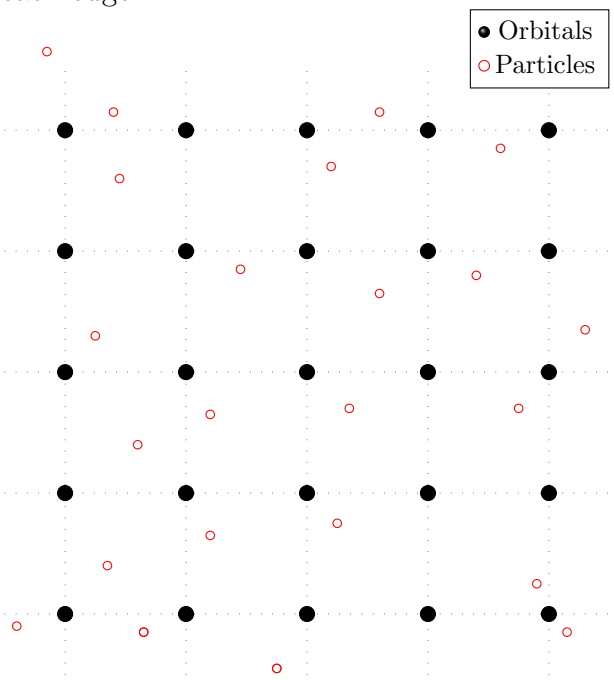
Computation of the Acceptance Probability

Because the transition probability (the probability to accept a particle move) is given by $\min\left(1, \quad \left|\frac{\Psi(R')}{\Psi(R)}\right|^2\right)$, we need to calculate $\left|\frac{\Psi(R')}{\Psi(R)}\right|^2$ at every Monte Carlo step. By (2.1), we have

$$\left|\frac{\Psi(R')}{\Psi(R)}\right|^2 = \left|\frac{\det(A')}{\det(A)}\right|^2.$$

So the computation of the transition probability is converted to calculate the squared determinant ratio, $\left|\frac{\det(A')}{\det(A)}\right|$.

Figure 2.1: Partial particles and orbitals in a 2D system. The domain is a periodic Cartesian grid with 5 orbitals on each edge.



In every step, we attempt to move one particle. Suppose at at Monte Carlo step k, we attempt to move particle i_k from position r_{i_k} to r'_{i_k} and the attempt is accepted.

$$A_k = \begin{pmatrix} \phi_1(r_1) & \phi_1(r_1) & \cdots & \phi_N(r_1) \\ \cdots & \cdots & & \cdots \\ \phi_1(r_{i_k}) & \phi_2(r_{i_k}) & \cdots & \phi_N(r_{i_k}) \\ \cdots & \cdots & & \cdots \\ \phi_1(r_N) & \phi_2(r_N) & \cdots & \phi_N(r_N) \end{pmatrix};$$

$$A_{k+1} = \begin{pmatrix} \phi_1(r_1) & \phi_1(r_1) & \cdots & \phi_N(r_1) \\ \cdots & \cdots & & \cdots \\ \phi_1(r'_{i_k}) & \phi_2(r'_{i_k}) & \cdots & \phi_N(r'_{i_k}) \\ \cdots & \cdots & & \cdots \\ \phi_1(r_N) & \phi_2(r_N) & \cdots & \phi_N(r_N) \end{pmatrix}.$$

The only difference between A_k and A_{k+1} is row i_k. Let u_k be a column vector such that

$$(u_k)_j = \phi_j(r'_{i_k}) - \phi_j(r_{i_k}), \quad \text{for} \quad j = 1, \ldots, N.$$

Then we have

$$\begin{aligned} A_{k+1} &= A_k + e_{i_k} u_k^T \\ &= A_k(I + A_k^{-1} e_{i_k} u_k^T) \end{aligned}$$

Therefore the determinant ratio satisfies

$$\begin{aligned} \left| \frac{\det(A_{k+1})}{\det(A_k)} \right| &= \left| \det(I + A_k^{-1} e_{i_k} u_k^T) \right| \\ &= \left| 1 + u_k^T A_k^{-1} e_{i_k} \right|. \end{aligned}$$

So now, we must find $A_k^{-1} e_{i_k}$ at every Monte Carlo step. We compute $A_k^{-1} e_{i_k}$ by solving

$$A_k z_k = e_{i_k} \tag{2.2}$$

with preconditioned GMRES rather than using an explicit inverse of A_k.

Algorithm 1 shows the MCMC process.

2.2 Properties of the Slater Matrix

In this section, we analyze some properties of Slater matrices occurring in the QMC method for our setup. We store Slater matrix as a sparse matrix and a typical sparsity pattern of a

Algorithm 1 Markov Chain Monte Carlo with Metropolis algorithm

1: Evaluate the energy $E(R)$ with density $\Psi^2(R)$.
2: **for** $k = 1$ to $totalSteps$ **do**
3: Generate a random step d, $R' = R + d$.
4: Calculate the squared ratio $\left|\frac{\Psi(R')}{\Psi(R)}\right|^2$ and generate a random number $p \in (0, 1)$.
5: **if** $(\left|\frac{\Psi(R')}{\Psi(R)}\right|^2 > p)$ **then**
6: Accept the new configuration R' and evaluate $E(R')$ with new configuration R'
7: **else**
8: Reject the particle move. $E(R)$ is counted again.
9: **end if**
10: **end for**

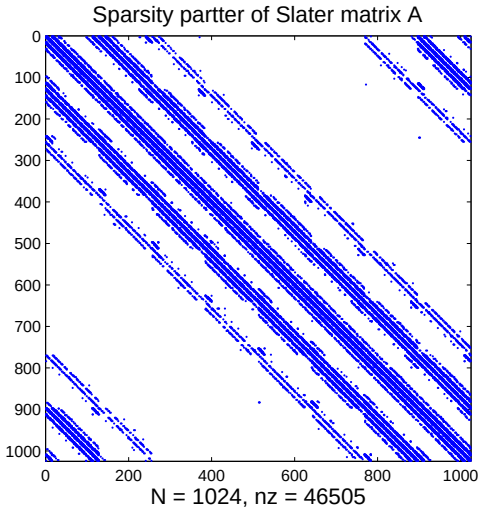
specific Slater matrix ordered to have each orbital paired with a nearby particle is shown in Figure 2.2.

The Slater matrices in our MCMC process are generally nonsingular. If a proposed particle move would result in the Slater matrix being singular, the acceptance probability would be zero because the determinant ratio would be zero. We always reject this move attempt, and thus the Slater matrix is always nonsingular.

The Slater matrix after appropriate reordering is close to a circulant matrix [39, 58, 40]. Each row of the Slater matrix has large elements around the orbital where the Gaussian is centered at. Any circulant matrix is normal, but the Slater matrix is generally not normal.

The spectrum or pseudospectrum are important in analyzing the iterative methods [88]. Figure 2.3 shows the spectrum of a Slater matrix. Notice that this spectrum is under our reordering algorithm that improves the diagonal dominance (see Section 5.2). The singular values are also important. For example, the bounds on the largest and smallest singular values are important for the analysis of convergence of Krylov subspace methods. So, we will analyze the bounds on smallest and largest singular values of the Slater matrix. In general,

Figure 2.2: The sparsity pattern of a Slater matrix in a specific Monte Carlo step for a 3D system with N = 1024, K = 1. The Slater matrix is reordered by our reordering algorithm that improves the diagonal dominance (see Section 5.2).



for smallest singular value, it is hard to obtain a practical lower bound. However, due to the acceptance/rejection test in QMC, Slater matrices are ill-conditioned with very low probability and singular with probability zero. Therefore, we try to bound the smallest singular value from below based on the distribution of singular values in equilibrium configuration and the transition probability.

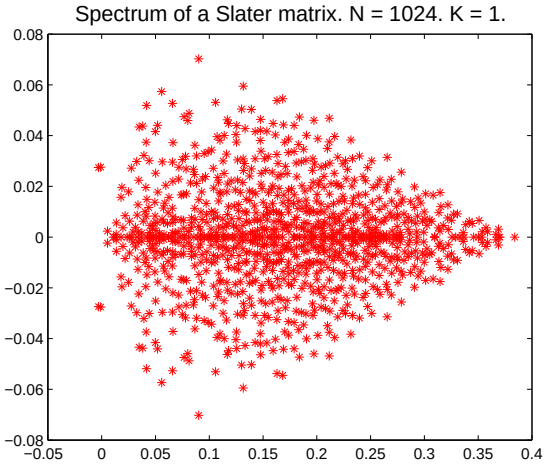
Largest Singular Value

Assume a Slater matrix A has singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n > 0$. A simple upper bound of $\sigma_1(A)$ is given in

$$\sigma_1(A) \leq \sqrt{\|A\|_1 \|A\|_\infty}. \tag{2.3}$$

On the right side of the inequality (2.3), $\|A\|_1$ is the largest column sum. Since each column relates to an orbital, the column with largest sum corresponds the orbital which has most

Figure 2.3: The spectrum of a Slater matrix in a specific Monte Carlo step for a 3D system with N = 1024, K = 1. The Slater matrix is reordered by our reordering algorithm that improves the diagonal dominance (see Section 5.2).



particles near it. $\|A\|_\infty$ is the largest row sum. Since each row relates to a particle and the orbitals are evenly distributed, we can obtain a bound on the row sum.

Let $\phi_j(\mathbf{r}) = e^{-K\|\mathbf{r}-\mathbf{O}_j\|^2}, j = 1, \cdots, N$, and thus $a_{ij} = e^{-K\|r_i-\mathbf{O}_j\|^2}$. Let I be the domain of all particles and orbitals. Then

$$\begin{aligned} \|A\|_\infty &= \max_i \sum_{j=1}^{n} |a_{ij}| = \max_i \sum_{j=1}^{n} e^{-K\|r_i-\mathbf{O}_j\|^2} \\ &= \max_{r_i} \sum_{j=1}^{n} e^{-K\|r_i-\mathbf{O}_j\|^2} \leq \max_{r\in I} \sum_{j=1}^{n} e^{-K\|\mathbf{r}-\mathbf{O}_j\|^2}. \end{aligned} \tag{2.4}$$

Because the orbitals are evenly distributed in the entire domain I, the maximum does not change if we restrict r from I into a smaller subinterval I_0 which covers the distance between two adjacent orbitals. We can even restrict r onto a smaller subinterval because Gaussian functions are symmetric and the domain is periodic. For example, in 1D case, if $I = [0, (n-1)h]$ where h is the distance between two adjacent orbitals, then we can take $I_0 = [0, h]$. In 2D case, if $I = [0, (n-1)h]^2$, then we can take $I_0 = [0, h]^2$. Generally, if $I = [0, (n-1)h]^n$,

then $I_0 = [0, h]^n$. Hence,

$$\begin{aligned} \|A\|_\infty &\leq \max_{r \in I_0} \sum_{j=1}^{n} e^{-K\|\mathbf{r}-\mathbf{O}_j\|^2} \\ &= \max_{r \in I_0} \sum_{j=1}^{n} \phi_j(\mathbf{r}). \end{aligned} \tag{2.5}$$

We don't normally have a similar bound on $\|A\|_1$. However, when the system is in equilibrium, we expect particles are evenly distributed and the row sums should not be much different. So the bound for the largest singular value is

$$\sigma_1(A) \leq \sqrt{\|A\|_1 \max_{r \in I_0} \sum_{j=1}^{n} \phi_j(r)}.$$

In our experiments for a 3D system with $N = 1024$, $K = 1$, we calculate the bound of largest column sum is $\|A\|_\infty \leq 1.4632$. For $\|A\|_1$, it is always less than 2.6587 for all Monte Carlo steps. Therefore, the resulting bound for the largest singular value is $\sigma_1(A) \leq \sqrt{\|A\|_1 \|A\|_\infty} = \sqrt{2.6587 * 1.4632} = 1.9724$. In our experiments, $\sigma_1(A)$ is always less than 1.2, so 1.9724 is not very accurate but still a good bound.

In our experiments, we have $\sigma_1 \approx 1$.

Smallest Singular Values

For an arbitrary matrix, there are various lower bounds on the smallest singular values [15, 50, 84, 89]. For example, the following bound is based on Frobenius norm and determinant:

$$\sigma_n(A) \geq \left(\frac{n-1}{\|A\|_F^2}\right)^{(n-1)/2} |\det(A)|. \tag{2.6}$$

By (2.4) and (2.5), we can rewrite the Frobenius norm. Then

$$\begin{aligned}
\|A\|_F^2 &= \sum_{i=1}^{n}\sum_{j=1}^{n}|a_{ij}|^2 = \sum_{i=1}^{n}\sum_{j=1}^{n}(e^{-K\|r_i-\mathbf{O}_j\|^2})^2 \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n}e^{-2K\|r_i-\mathbf{O}_j\|^2} \leq n\cdot\max_i\sum_{j=1}^{n}e^{-2K\|r_i-\mathbf{O}_j\|^2} \\
&= n\cdot\max_{r\in I_0}\sum_{j=1}^{n}e^{-2K\|r-\mathbf{O}_j\|^2} = n\cdot\max_{r\in I_0}\sum_{j=1}^{n}\phi_j(r)^2.
\end{aligned}$$

Substituting this back into (2.6), we get

$$\begin{aligned}
\sigma_n(A) &\geq \left(\frac{n-1}{n\cdot\max_{r\in I_0}\sum_{j=1}^{n}\phi_j(r)^2}\right)^{\frac{n-1}{2}}|\det(A)|. \\
&\approx \left(\frac{1}{\max_{r\in I_0}\sum_{j=1}^{n}\phi_j(r)^2}\right)^{\frac{n-1}{2}}|\det(A)|. \\
&= \beta|\det(A)|,
\end{aligned}$$

where $\beta = (\max_{r\in I_0}\sum_{j=1}^{n}\phi_j(r)^2)^{-\frac{n-1}{2}}$ is a constant. This is a new lower bound on the smallest singular values. However, it is not a practical bound because the parameter β and the determinant of A are not cheap to compute and may also be very small. Since it is hard to get a practical lower bound on the smallest singular value a priori, we can examine how much a smallest singular value can change by analyzing the ratio of two successive smallest singular values.

Lower Bounds on Smallest Singular Values

In this section, we try to develop a lower bound on the smallest singular value by making use of the distribution properties of singular values and the acceptance/rejection test in our QMC problem. The acceptance/rejection test tends to accept those particle moves with high probability. Therefore, we expect the smallest singular value would not become smaller and smaller.

Now we explain how we obtain the bound on the smallest singular value. We first introduce a well-known result in the book of *Topics in matrix analysis* by Horn and Johnson [48]. Notice that $M_{m,n}$ represents the set of $m \times n$ matrices.

Theorem 1. *Let $A \in M_{m,n}$ be given, and let A_r denote a submatrix of A obtained by deleting a total of r rows and/or columns from A. Then*

$$\sigma_k(A) \geq \sigma_k(A_r) \geq \sigma_{k+r}(A), \quad k = 1, \ldots, \min\{m, n\},$$

where for $X \in M_{p,q}$, we set $\sigma_j(X) = 0$ if $j > \min\{p, q\}$.

Proof. See the proof of Corollary 3.1.3 in Horn and Johnson [48]. □

In QMC method, the Slater matrix A is updated by cumulative rank-1 updates. Assume at some Monte Carlo step k, we attempt to move the particle j. Therefore, the j-th row of A_k will be updated if the move is accepted. Let $\widetilde{A}_k$ be the submatrix of A_k obtained by deleting the j-th row. Let A_{k+1} be the matrix with updated j-th row when the move is accepted.

Suppose the singular values of A_k, $\widetilde{A}_k$, A_{k+1} are $\sigma_i^{(k)}$, $\tilde{\sigma}_i$, $\sigma_i^{(k+1)}$, $i = 1, 2, \ldots, n$, respectively. It follows from theorem 1 that

$$\sigma_1^{(k)} \geq \tilde{\sigma}_1 \geq \sigma_2^{(k)} \geq \tilde{\sigma}_2 \geq \cdots \geq \sigma_{n-1}^{(k)} \geq \tilde{\sigma}_{n-1} \geq \sigma_n^{(k)}, \tag{2.7}$$

$$\sigma_1^{(k+1)} \geq \tilde{\sigma}_1 \geq \sigma_2^{(k+1)} \geq \tilde{\sigma}_2 \geq \cdots \geq \sigma_{n-1}^{(k+1)} \geq \tilde{\sigma}_{n-1} \geq \sigma_n^{(k+1)} \tag{2.8}$$

With (2.3) and (2.5), we have an upper bound on the largest singular values of the Slater matrices

$$\sigma_1(A) \leq \sqrt{\|A\|_1 \|A\|_\infty} \leq \sqrt{\|A\|_1 \max_{r \in I_0} \sum_{j=1}^{n} \phi_j(r)}.$$

Let the upper bound be $\sigma_{\max}$. Then (2.7) and (2.8) can be rewritten as

$$\tilde{\sigma}_1 \leq \sigma_1^{(k)}, \sigma_1^{(k+1)} \leq \sigma_{\max}, \tag{2.9}$$

$$\tilde{\sigma}_{i+1} \leq \sigma_i^{(k)}, \sigma_i^{(k+1)} \leq \tilde{\sigma}_i, \quad i = 2, 3, \ldots, n-1. \tag{2.10}$$

Further, if we look at the ratio of $\frac{\sigma_i^{(k+1)}}{\sigma_i^{(k)}}, i = 1, 2, \ldots, n-1$, we get

$$\frac{\tilde{\sigma}_1}{\sigma_{\max}} \leq \frac{\sigma_1^{(k+1)}}{\sigma_1^{(k)}} \leq \frac{\sigma_{\max}}{\tilde{\sigma}_1} \tag{2.11}$$

$$\frac{\tilde{\sigma}_{i+1}}{\tilde{\sigma}_i} \leq \frac{\sigma_i^{(k+1)}}{\sigma_i^{(k)}} \leq \frac{\tilde{\sigma}_i}{\tilde{\sigma}_{i+1}}, \quad i = 2, 3, \ldots, n-1. \tag{2.12}$$

By the property of the determinants, we have

$$\left|\frac{\det(A_{k+1})}{\det(A_k)}\right| = \frac{\sigma_1^{(k+1)}}{\sigma_1^{(k)}} \frac{\sigma_2^{(k+1)}}{\sigma_2^{(k)}} \cdots \frac{\sigma_{n-1}^{(k+1)}}{\sigma_{n-1}^{(k)}} \frac{\sigma_n^{(k+1)}}{\sigma_n^{(k)}}.$$

Define $\gamma_i := \frac{\sigma_i^{(k+1)}}{\sigma_i^{(k)}}, i = 1, 2, \ldots, n-1$. Then

$$\left|\frac{\det(A_{k+1})}{\det(A_k)}\right| = \prod_{i=1}^{n-1} \gamma_i \frac{\sigma_n^{(k+1)}}{\sigma_n^{(k)}}. \tag{2.13}$$

By (2.11) and (2.12), we have

$$\begin{aligned} & \frac{\tilde{\sigma}_1}{\sigma_{\max}} \frac{\tilde{\sigma}_2}{\tilde{\sigma}_1} \frac{\tilde{\sigma}_3}{\tilde{\sigma}_2} \cdots \frac{\tilde{\sigma}_{n-1}}{\tilde{\sigma}_{n-2}} \leq \textstyle\prod_{i=1}^{n-1} \gamma_i \leq \frac{\sigma_{\max}}{\tilde{\sigma}_1} \frac{\tilde{\sigma}_1}{\tilde{\sigma}_2} \frac{\tilde{\sigma}_2}{\tilde{\sigma}_3} \cdots \frac{\tilde{\sigma}_{n-2}}{\tilde{\sigma}_{n-1}}, \\ \Leftrightarrow \quad & \frac{\tilde{\sigma}_{n-1}}{\sigma_{\max}} \leq \textstyle\prod_{i=1}^{n-1} \gamma_i \leq \frac{\sigma_{\max}}{\tilde{\sigma}_{n-1}}. \end{aligned} \tag{2.14}$$

It follows from (2.13) and (2.14) that

$$\frac{\tilde{\sigma}_{n-1}}{\sigma_{\max}} \frac{\sigma_n^{(k+1)}}{\sigma_n^{(k)}} \leq \left|\frac{\det(A_{k+1})}{\det(A_k)}\right| \leq \frac{\sigma_{\max}}{\tilde{\sigma}_{n-1}} \frac{\sigma_n^{(k+1)}}{\sigma_n^{(k)}}.$$

Suppose the acceptance probability is $p_k = \left|\frac{\det(A_{k+1})}{\det(A_k)}\right|^2$. Then we have

$$\frac{\tilde{\sigma}_{n-1}}{\sigma_{\max}} \frac{\sigma_n^{(k+1)}}{\sigma_n^{(k)}} \leq \sqrt{p_k} \leq \frac{\sigma_{\max}}{\tilde{\sigma}_{n-1}} \frac{\sigma_n^{(k+1)}}{\sigma_n^{(k)}},$$

$$\Rightarrow \quad \sqrt{p_k}\frac{\tilde{\sigma}_{n-1}}{\sigma_{\max}} \leq \frac{\sigma_n^{(k+1)}}{\sigma_n^{(k)}} \leq \sqrt{p_k}\frac{\sigma_{\max}}{\tilde{\sigma}_{n-1}}. \tag{2.15}$$

Also it follows from (2.13) that

$$\sqrt{p_k} = \prod_{i=2}^{n-1}\gamma_i \frac{\sigma_n^{(k+1)}}{\sigma_n^{(k)}},$$

$$\Rightarrow \frac{\sigma_n^{(k+1)}}{\sigma_n^{(k)}} = \frac{\sqrt{p_k}}{\prod_{i=2}^{n-1}\gamma_i}. \tag{2.16}$$

Finally, we get to the bounds on $\frac{\sigma_n^{(k+1)}}{\sigma_n^{(k)}}$, where $\frac{\sigma_n^{(k+1)}}{\sigma_n^{(k)}}$ is the ratio between two consecutive smallest singular values. It tells us how the smallest singular values changes. By looking into the values of $\sqrt{p_k}\frac{\tilde{\sigma}_{n-1}}{\sigma_{\max}}$, $\sqrt{p_k}\frac{\sigma_{\max}}{\tilde{\sigma}_{n-1}}$, and $\frac{\sqrt{p_k}}{\prod_{i=2}^{n-1}\gamma_i}$, we could know how much the smallest singular values changes and its lower bound. In our experiments, for example, Figure 2.4 and 2.5 show that the value of $\frac{\tilde{\sigma}_{n-1}}{\sigma_{\max}}$ is between 4.5e-3 and 8e-3 for a system with N = 1024, K = 1. Let $\alpha = \frac{\tilde{\sigma}_{n-1}}{\sigma_{\max}}$. Therefore, with (2.15) we have

$$\sqrt{p_k}\alpha \leq \frac{\sigma_n^{(k+1)}}{\sigma_n^{(k)}} \leq \sqrt{p_k}\frac{1}{\alpha},$$

where $\alpha \in [4.5e-3, \quad 8e-3]$.

For the bound (2.16), we consider two cases, one is when γ_i's are uniformly distributed, and the other is when $\sigma_i^{(k)}$'s, $\sigma_i^{(k+1)}$'s are uniformly distributed.

Figure 2.4: The value of $\frac{\tilde{\sigma}_{n-1}}{\sigma_{\max}}$ with row i removed for $i = 1, 2, \cdots, N$, for a Slater matrix with N = 1024, K = 1. The x coordinate is the row index.

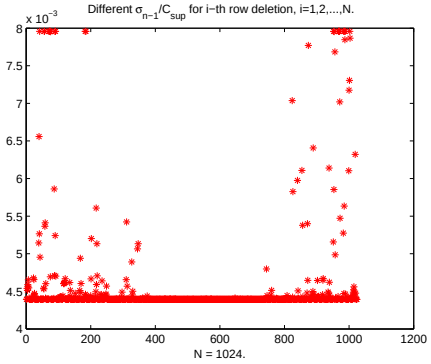
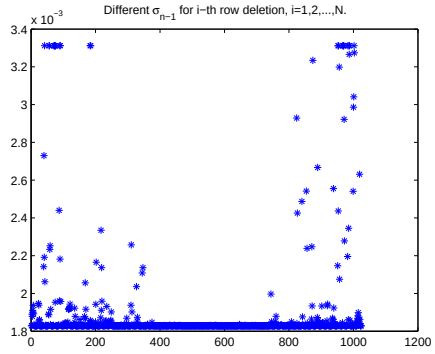


Figure 2.5: The value of $\tilde{\sigma}_{n-1}$ with row i removed for $i = 1, 2, \cdots, N$, for a Slater matrix with N = 1024, K = 1.



Bounds when γ_i's are Uniformly Distributed

Suppose each γ_i is uniformly distributed on the subinterval of $\left[\frac{\tilde{\sigma}_{i+1}}{\tilde{\sigma}_i}, \frac{\tilde{\sigma}_i}{\tilde{\sigma}_{i+1}}\right]$, $i = 1, 2, \ldots, n-1$. For a 3D system with N = 1024 and K = 1, the product of $\prod_{i=2}^{n-1} \gamma_i$ and its distribution are shown in Figure 2.6 and 2.7. The mean value of $\prod_{i=2}^{n-1} \gamma_i$ is very close to 1. The range in this experiment is $[0.3, 5.5]$. Therefore, the ratio between two consecutive smallest singular values is not expected to change much because of (2.16). p_k would have to be very small.

Figure 2.6: The value of the product, $\prod_{i=2}^{n-1} \gamma_i$, when γ_i's are uniformly distributed. Every single blue dot is a value at a specific Monte Carlo step. The total steps is 10000.

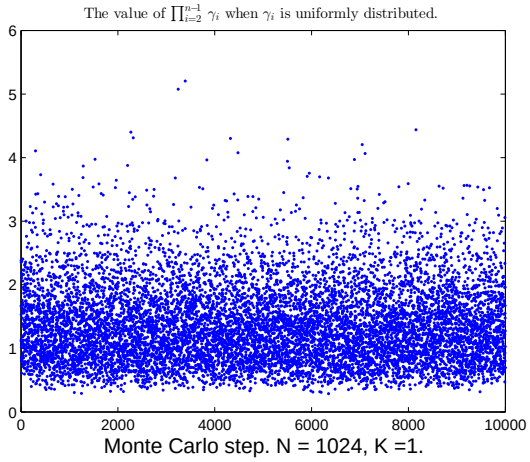
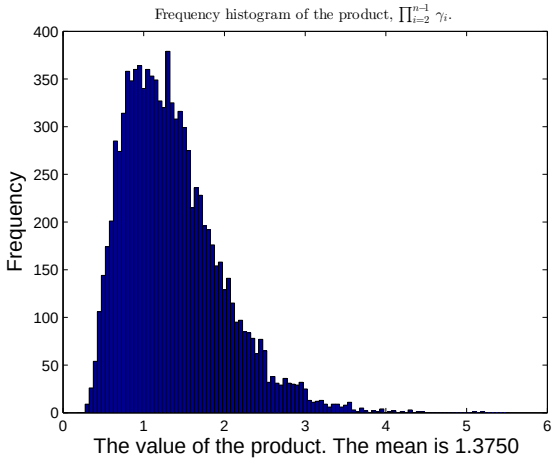


Figure 2.7: Frequency histogram of the product, $\prod_{i=2}^{n-1} \gamma_i$,when γ_i's are uniformly distributed. The mean of the product is 1.3750 and the total Monte Carlo steps is 10000.



Bounds when $\sigma_i^{(k)}$'s, $\sigma_i^{(k+1)}$'s are Uniformly Distributed

Suppose as an alternative $\sigma_i^{(k)}, \sigma_i^{(k+1)}$, $i = 2, 3, \ldots, n-1$, are independent and uniformly distributed on the subinterval $[\tilde{\sigma}_{i+1}, \tilde{\sigma}_i]$. For the same system with N = 1024 and K = 1, the product of $\prod_{i=2}^{n-1} \gamma_i$ and its distribution are shown in Figure 2.8 and 2.9. Similarly, $\prod_{i=2}^{n-1} \gamma_i$ is in the range of $[0.35, 3.5]$, which is even better than the case when γ_i's are uniformly distributed. As a result, the ratio between two consecutive smallest singular values can only be small with very low probability p_k.

Figure 2.10 and 2.11 shows the statistical result of smallest singular values of the Slater matrix in these experiments. The smallest singular value is around $1e-3$ and does not change much when the system is in equilibrium. The statistical result has justified our assumption.

Figure 2.8: The value of the product, $\prod_{i=2}^{n-1}\gamma_i$, when $\sigma_i^{(k)}, \sigma_i^{(k+1)}$ is uniformly distributed.

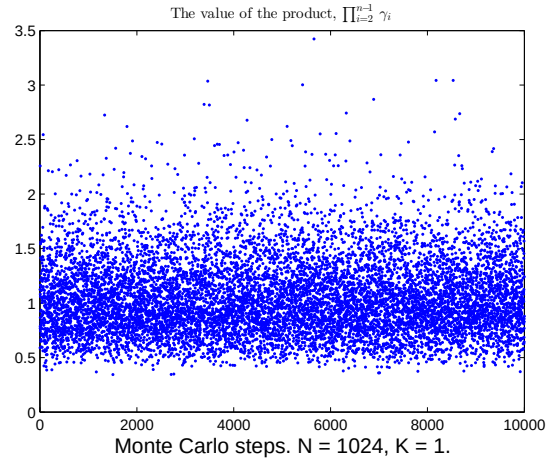


Figure 2.9: Frequency histogram of the product, $\prod_{i=2}^{n-1}\gamma_i$, when $\sigma_i^{(k)}, \sigma_i^{(k+1)}$ is uniformly distributed. The mean of the product is 1.3750 and the total Monte Carlo steps is 10000.

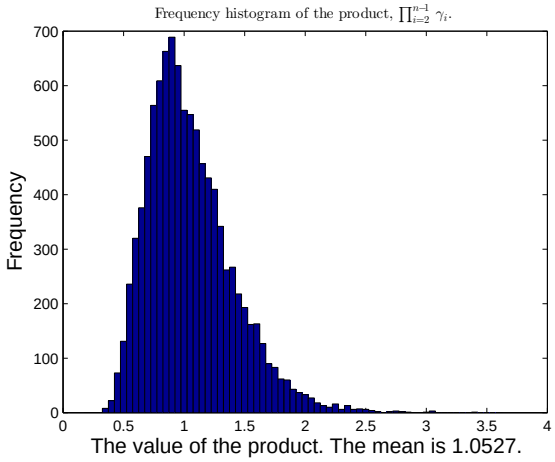


Figure 2.10: Smallest singular values in 600 Monte Carlo steps for a 3D system with N = 1024, K = 1.

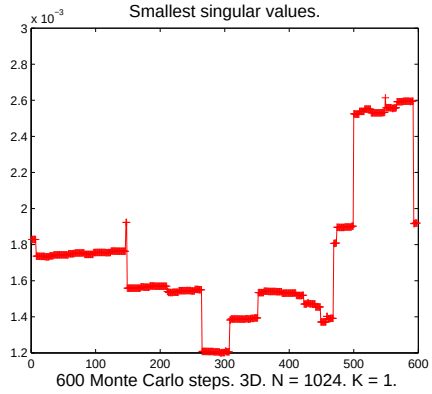
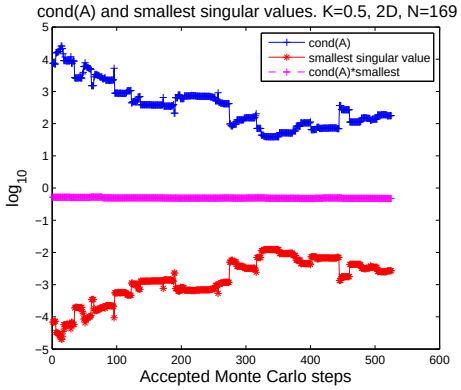


Figure 2.11: Condition number and smallest singular values of Slater matrix for a 2D system with N = 169, K = 0.5. Total Monte Carlo steps is 1000 and acceptance ratio is 54%.



Chapter 3

Preconditioned GMRES for QMC

The convergence of Krylov subspace methods depends on the spectrum of the system matrix, and can be significantly improved with preconditioners. In order to be efficient and robust, Krylov subspace methods need to be preconditioned. Preconditioners modify the spectrum of the system matrix in order to reduce the number of iterative steps required for convergence. In this dissertation, we employ generalized minimal residual (GMRES) method. We choose full GMRES because it is an optimal Krylov subspace method for nonsymmetric matrices, and its effect is simple to analyze. It is easy to interpret the number of iterations to converge. If we change preconditioners, the change in iterations counts tells us if we are doing well. We can also use simple convergence bounds to present an analysis of convergence choices, and to present potential users with a "framework" to understand how to make preconditioning choices.

In Section 3.1, we introduce the preconditioned GMRES and provide some well-know results on its convergence. Three preconditioners are tested and analyzed on the application to the QMC method presented in Chapter 2. Section 3.2 discusses ILU(0) and ILUTP preconditioners. A domain decomposition preconditioner is introduced and tested in Section 3.3. The last section introduces preconditioning for Schur complement system based domain decomposition method with inexact matrix-vector products. All experiments are done in

Matlab.

3.1 Preconditioned GMRES

GMRES [83, 82, 72] is an iterative method for solving linear systems. It is a Krylov subspace method that generates a sequence of orthogonal vectors. While CG and MINRES are suitable for symmetric systems, GMRES is suitable for nonsymmetric systems [83, 75].

Suppose we need to solve

$$Ax = b.$$

Let r_0 be the initial residual. The Krylov subspace associated with A and r_0 is $K^m(A, r_0) = \text{span}\{r_0, Ar_0, A^2r_0, \cdots, A^{m-1}r_0\}$. At each iteration, GMRES seeks the unique solution that minimizes the least squares problem

$$\min_{x \in x_0 + K^m} \|b - Ax\|_2.$$

Since the basis vectors $r_0, Ar_0, A^2r_0, \cdots, A^{m-1}r_0$ may be almost linearly dependent, we use the Arnoldi recurrence to find an basis V_m for K^m [57],

$$AV_m = V_{m+1}\underline{H}_m,$$

where $\underline{H}_m \in \mathbb{R}^{(m+1)\times m}$ is an upper Hessenberg matrix. The Arnoldi recurrence uses the (modified) Gram-Schmidt orthogonalization. The iterate x_m is defined as

$$x_m = x_0 + V_m y,$$

where column vectors of V_m are basis of K^m, and y is a column vector. Let $\beta = \|r_0\|_2$ and $e_1 = (1, 0, \ldots, 0)^T \in R^{m+1}$. We have

$$\begin{aligned} \|r_m\|_2 &= \|r_0 - A(x_m - x_0)\|_2 \\ &= \|V_{m+1}(\beta e_1 - \underline{H}_m y)\|_2 \\ &= \|\beta e_1 - \underline{H}_m y\|_2. \end{aligned}$$

Setting

$$y_m = \arg\min_y \|\beta e_1 - \underline{H}_m y\|_2$$

yields the minimal residual. The minimizer y_m is inexpensive to compute, since it requires the solution of an $(m+1)\times m$ least squares problem, where m is typically small (for restarted GMRES, m is chosen).

GMRES with Preconditioner

Any preconditioner P is a nonsingular matrix that approximates A in some way such that $P^{-1}A$ or AP^{-1} is better conditioned than A. There are a few different ways to apply the preconditioner, left preconditioning, right preconditioning, and split preconditioning.

GMRES with Right Preconditioning

Instead of solving the original system, we solve the right preconditioned system

$$AP^{-1}Px = b$$

via solving $AP^{-1}y = b$ for y first, and then $Px = y$ for x. The algorithm is shown in Algorithm 2.

GMRES with Left Preconditioning

Alternatively, we can solve the left preconditioned system

$$P^{-1}Ax = P^{-1}b.$$

Algorithm 2 GMRES wiht left preconditioning

Choose an initial guess x_0. $r_0 = b - Ax_0$. $\beta = \|r_0\|_2$. $a = \beta$. $k = 0$.
$v_1 = r_0/\|r_0\|_2$.
while $a > \epsilon\|b\|_2$ and $k < kmax$ **do**
  $k = k + 1$
  Compute $z = P^{-1}v_j$
  Compute $w = Az$
  for $j = 1, \ldots, k$ **do**
    $h_{jk} = w^H v_j$
    $w = w - \sum_{j=1}^{k} h_{jk} v_j$
  end for
  $h_{k+1,k} = \|w\|_2$
  $v_{k+1} = \dfrac{w}{\|w\|_2}$
  $e_1 = (1, 0, \ldots, 0)^T \in R^{m+1}$.
  Minimize $\|\beta e_1 - \underline{H}_m y\|$ over R^m to obtain y_m.
  $a = \|\beta e_1 - \underline{H}_m y_m\|$.
end while
$x_k = x_0 + P^{-1}V_k y_k$.

Convergence Results on GMRES

Before we discuss preconditioners in detail and analyze their effect on convergence, we introduce some (well-known) GMRES convergence results that will be used in this dissertation.

The first result is when a nonsingular matrix A has decomposition $A = I - C$. The following theorem shows that when $\|C\| \leq \rho < 1$, GMRES converges for $Ax = b$. Further, GMRES converges fast if ρ is small, for instance, $\rho < \dfrac{1}{2}$. Related analysis and results can be found in Kerkhoven and Saad [54], Gmati [38], and Nevanlinna [64].

Theorem 2. *For any matrix A, suppose $Ax = b$ is being solved by GMRES. If $A = I - C$ and $\|C\| \leq \rho < 1$, then the residual r_k at k-th step satisfies*

$$\|r_k\| \leq \rho^k \|r_0\|,$$

where r_0 is the initial residual.

Proof. Consider the polynomial $p_k(A) = (I - A)^k$, which is in the set of all k-degree poly-

nomial, $\mathcal{P}_k$, and $p_k(\mathbf{0}) = (I - \mathbf{0})^k = I$. Therefore p_k is a residual polynomial for Krylov subspace methods. By the optimality of GMRES (see also Theorem 3.1.1 in Kelley [52]), the residual r_k^{GMRES} satisfies

$$\|r_k^{\text{GMRES}}\| = \min_{p \in \mathcal{P}_k, p(0)=1} \|p(A)r_0\| \leq \|p_k(A)r_0\|. \tag{3.1}$$

Because $A = I - C$, we get $I - A = C$ and thus $p_k(A) = C^k$. Substitute C^k into (3.1):

$$\|r_k^{\text{GMRES}}\| \leq \|C^k r_0\| \leq \|C^k\|\|r_0\| \leq \|C\|^k \|r_0\| \leq \rho^k \|r_0\|.$$

□

The second bound is obtained by the field of values. When the field of values of a nonsingular matrix A is contained in an ellipse or disk that excludes the origin, GMRES converges for $Ax = b$. Greenbaum [41] gives an error bound though the bound can be pessimistic. In this dissertation, $\mathcal{F}(A)$ is used to represent the field of values of A.

Lemma 3. *Suppose the field of values of A is contained in the disk $\mathbf{D} = \{z \in \mathbb{C} : |z - c| \leq s, s < |c|\}$, which excludes the origin. Let $\|E\| \leq \varepsilon$ such that $s + \varepsilon < |c|$. Then the field of values of $A + E$ is contained in a new disk $\mathbf{D}_\varepsilon = \{z \in \mathbb{C} : |z - c| \leq s + \varepsilon, \quad s + \varepsilon < |c|\}$, and $\mathbf{D}_\varepsilon$ excludes the origin.*

Proof. Since $\varepsilon < |c| - s$, we have $s + \varepsilon < |c|$ and thus $\mathbf{D}_\varepsilon$ excludes the origin. For an arbitrary z in $\mathcal{F}(A + E)$, it follows from the definition of field of values that there exists a unit vector x such that $z = x^H(A + E)x$. Since $\mathcal{F}(A) \subset \mathbf{D}$, $\|x\| = 1$ and $\|E\| \leq \varepsilon$, we get

$$\begin{aligned} |z - c| &= |x^H(A + E)x - c| \\ &= |x^H Ax + x^H Ex - c| \\ &\leq |x^H Ax - c| + |x^H Ex| \\ &\leq s + \varepsilon. \end{aligned}$$

Therefore $z \in \mathbf{D}_\varepsilon$. Since z is an arbitrary value in $\mathcal{F}(A + E)$, it follows that $\mathcal{F}(A + E) \subset \mathbf{D}_\varepsilon$. □

Theorem 4. *Let* $\mathcal{F}(A) \subset \mathbf{D} = \{z \in \mathbb{C}^2 : |z - c| \leq s, \quad s < |c|\}$ *and* E *with small 2-norm such that* $\|E\| = \varepsilon < |c| - s$. *If we are solving* $(A + E)x = b$ *with GMRES, then the residual* r_k^{GMRES} *at* k*-th iteration satisfies*

$$\|r_k^{\text{GMRES}}\|/\|r_0\| \leq 2\left(\frac{s+\varepsilon}{|c|}\right)^k. \tag{3.2}$$

Proof. Since A is perturbed by a matrix E such that $\|E\|_2 = \varepsilon < |c| - s$, it follows from lemma 3 that $\mathcal{F}(A+E) \subset \mathbf{D}_\varepsilon = \{z \in \mathbb{C}^2 : |z - c| \leq s + \varepsilon, \quad s + \varepsilon < c\}$. Then for the system $(A + E)x = b$, by (3.18) in p.56 in Greenbaum [41], the GMRES residual r_k^{GMRES} at k-th iteration satisfies (3.2). □

Another similar result is when the spectrum of A is contained in a disk that excludes the origin. Such a disk with small radius guarantees fast convergence of GMRES. Actually, if the center and radius are c and r, the polynomial

$$p_k(z) = \left(\frac{c - z}{c}\right)^k$$

can be used to bound the residual. For any z in the disk, there exists

$$|p_k(z)| \leq \left|\frac{r}{c}\right|^k.$$

This gives similar bound as in the field of values case:

$$\|r_k\|/\|r_0\| \leq \kappa(V)\left|\frac{r}{c}\right|^k, \tag{3.3}$$

where $\kappa(V) = \|V\|\|V^{-1}\|$ and $A = V\Lambda V^{-1}$ is the spectral decomposition. For more information about GMRES, see Saad [72, 73]. In the next subsections, we discuss our two truncation approaches and provide some theoretical analysis.

3.2 ILU Preconditioner

For large sparse system matrices, incomplete LU (ILU) factorizations are among of the most useful and popular preconditioners. An incomplete LU factorization of a matrix is a sparse

approximation to the LU factorization [72]. For example, given any Slater matrix A_k. Let P_k be its incomplete LU factorization. Then $P_k = L_k U_k \approx A_k$, where L_k and U_k are lower and upper triangular matrices.

In Ahuja et al. [2], incomplete LU decompositions with threshold and pivoting (ILUTP) preconditioners are used [72, 73]. Further, the preconditioner is updated by cumulative rank-one updates until a new preconditioner needs to be computed due to the instability. In our case, we still employ the cumulative rank-one updates approach. However, instead of ILUTP, we think about using incomplete LU factorization with no/zero fill-in, denoted by ILU(0). ILU(0) is cheaper to compute and apply than ILUTP. The reason we could use ILU(0) is because our reordering algorithm makes the Slater matrix close to diagonal dominant (see Section 5.2), and thus ILU(0) algorithm is stable. ILU(0) factorization can be defined in general as follows: any pair of matrices L and U such that the elements of $A - LU$ are zeros in the locations of $NZ(A)$, where $NZ(A)$ is the set of nonzero elements of A.

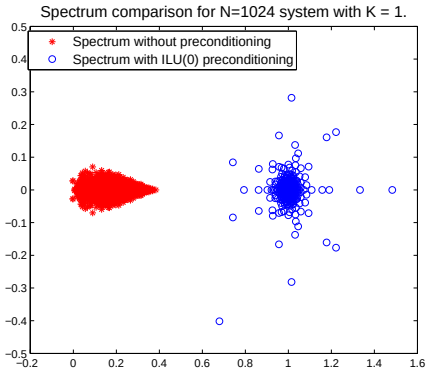
Numerical Results

ILU(0) is cheaper than ILUTP, and turns out to be effective as well. However, for some large system with $K = 0.5$, ILU(0) may fail because of zeros in the diagonal. If that is the situation, we need either use ILUTP to replace zeros in the diagonal by pivoting, or reorder the Slater matrix to make it closer to diagonal dominant. This is one of the reasons why we need to reorder the Slater matrix to improve its diagonal dominance.

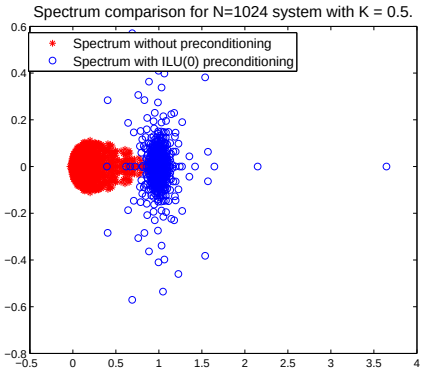
We look at the spectrum of preconditioned system, number of GMRES iterations, and the accuracy of the acceptance/rejection test. When $K = 0.5$, the convergence is not as good as that in the case when $K = 1$. This is expected because the Slater matrix is more ill-conditioned when $K = 0.5$. The spectrum of preconditioned system and number of GMRES iterations tell us how good our preconditioner is for the convergence. The accuracy of the acceptance and rejection test tells us how accurate our approximation is compared to the standard algorithm that uses exact inverse of Slater matrix.

First, we look at the spectrum with and without ILU preconditioning for Slater matrices. Figure 3.2(a) and 3.2(b) demonstrate the effectiveness of the ILU(0) preconditioner. The spectrum is very much improved by ILU(0) preconditioners. Because the spectrum of preconditioned system is around $(1, 0)$, we know the preconditioned system is of the form $I + C$. Therefore, it follows from theorem 2 that the preconditioned GMRES will convergence fast. This is demonstrated in Figure 3.3(a) and 3.3(b). The number of GMRES iterations is very small for the system with K = 1.

Figure 3.1: Spectrum comparison for Slater matrix with or without preconditioning. The preconditioner is ILU(0). This Slater matrix is related to a 3D system with N = 1024.



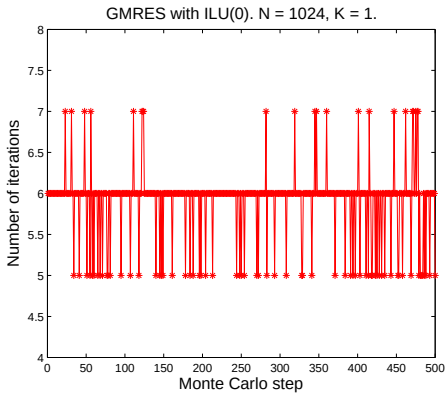
(a) K = 1



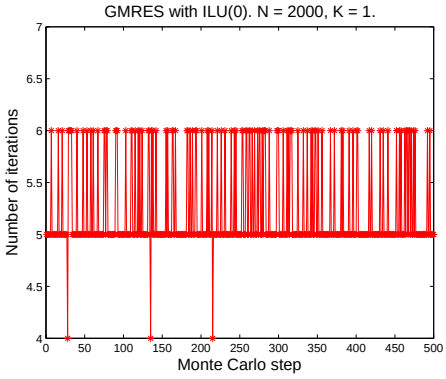
(b) K = 0.5

Next, we evaluate the stability of our ILU(0) preconditioner [9, 10, 20], which can be represented by $\|AU^{-1}L^{-1} - I\|_F$. Figure 3.3 shows the stability of ILU(0). This is another reason why GMRES converges fast.

Figure 3.2: Number of GMRES iterations with ILU(0) preconditioner and reordering. See Chapter 5 for detailed information about the reordering.

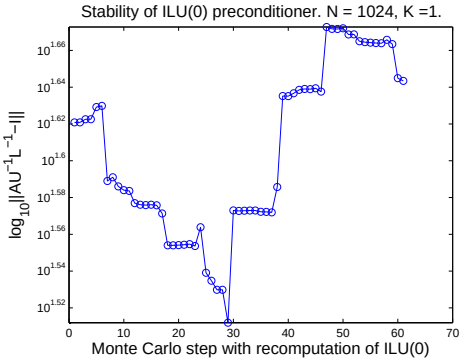


(a) Number of GMRES iterations for a system with N = 1024, K = 1



(b) Number of GMRES iterations for a system with N = 2000, K = 1

Figure 3.3: Stability of ILU(0) preconditioners for a system with N = 1024, K = 1.



Finally, we evaluate the accuracy of the acceptance and rejection test in the MCMC simulation. Table 3.1 shows that the iterative methods end up with very good approximation for different systems. The results for all systems are 100% good and at least 99.15% extremely good.

This method is based on Ahuja [2]. It produces a value f that tells how close the determinant ratio computed by our process is to the exact value by using exact inverses of Slater matrices. This value f also gives the probability of a wrong decision in the acceptance/rejection test. Let q be the square of the exact determinant ratio and q_z is the approximate one. Then

$f = |\min(q, 1) - \min(q_a, 1)|$ gives the probability of a wrong decision. The average value of f over a random walk gives the expected number of errors in the test. The approximation is defined as

- extremely good: $f < 1e - 04$
- very good: $f < 1e - 03$
- good: $f < 1e - 02$

Table 3.1: Results of using the average expected number of errors in the test to define the effectiveness of approximations. For all three systems, K = 1.

System size	1024	1458	2000
Extremly Good %	99.35	99.15	99.39
Very Good %	99.99	99.99	99.98
Good %	100	100	100
Acceptance Ratio %	0.5780	0.5878	0.5980

3.3 Domain Decomposition Preconditioner

Originally, the domain decomposition method is proposed to prove the existence of solutions of partial differential equations for non-standard domains [74, 77, 81]. The idea is to divide the entire domain into many possibly overlapping simpler subdomains, and to solve the partial differential equation alternatively on those subdomains. The boundary condition for each subdomain is based on the latest available approximation of the global solution and every new local solution is used to update the approximation of the global solution.

Now this method becomes popular in many fields and one application is as a preconditioner for Krylov subspace methods. The domain decomposition preconditioner is still based on the decomposition of the entire domain. The first step is to decompose the domain into

a couple of subdomains. Suppose the domain is Ω. We first decompose Ω into k non-overlapping subdomains $\Omega_1, \Omega_2, \ldots, \Omega_k$. Notice that $\Omega = \cup_{i=1}^{k} \Omega_i$ and $\Omega_i \cap \Omega_j = \emptyset$, for any $1 \leq i, j \leq k$. The next step is to decompose the matrix accordingly. This can be done by defining restriction operators R_i's for each subdomain. Formally, R_i can be expressed as

$$R_i = [I_i | O] \, \pi_i,$$

where I_i is the identity matrix that has the same dimension as the cardinality of Ω_i, π_i is the permutation on Ω_i. Another way to define the same R_i is to use delta function:

$$(R_i)_{ij} := \delta_{\omega_i, j} = \begin{cases} 1, & \omega_i = j \\ 0, & \text{otherwise} \end{cases} \quad \text{for } 1 \leq i, j \leq m,$$

where $\omega_i \in \Omega_i$. We view R_i's as restriction operators and R_i^T's as prolongations. Then we can define a restriction of the operator A on Ω_i as

$$A_i = R_i A R_i^T.$$

Now we can obtain domain decomposition preconditioners [74, 77, 33]. One option is additive Schwarz preconditioner P_{AS}, that is

$$P_{AS}^{-1} = \sum_{i=1}^{k} R_i^T A_i^{-1} R_i. \tag{3.4}$$

Another popular option is multiplicative Schwarz preconditioner [81, 33]. Define $Q_0 = I$ and $Q_i = (I - R_i^T A_I^{-1} R_i A) Q_{i-1}$, for $i = 1, \ldots, k$. Then the multiplicative Schwarz preconditioner P_{MS} is

$$P_{MS}^{-1} = (I - Q) A^{-1}. \tag{3.5}$$

In our experiments, we need to solve $A_k z_k = e_{i_k}$ iteratively by preconditioned GMRES at every Monte Carlo step. When the Slater matrix is updated by a rank-one update, only one row of the Slater matrix changes. The change is a small and local change to both physical system and the Slater matrix. So, we think about dividing the domain into two subdomains and employ domain decomposition preconditioner [77, 81]. Since we attempt to move a

particle at every Monte Carlo step, the first subdomain consists of the neighbor particles and orbitals of the moving particle. The second subdomain is the remaining particles and orbitals.

Two-level Multiplicative Schwarz Preconditioner

We will focus on one specific Monte Carlo step in this section and thus we skip the subscript k in A_k and z_k for simplicity. We are solving

$$Az = e_{i_k} \tag{3.6}$$

at a Monte Carlo step, where A is the Slater matrix and e_{i_k} is the unit vector with 1 at the i_k-th component. Notice that we attempt to move particle i_k at this Monte Carlo step.

We build and test two-level multiplicative Schwarz preconditioner (see (3.5)) on the application to the QMC method. The preconditioner P is $P_1 + P_2 - P_2AP_1$, where P_1 and P_2 are derived in below.

We now explain how to obtain the preconditioner. First, divide the domain (all particles and orbitals) into two subdomains. One is the local domain around the moving particle and the other consists of remaining orbitals and particles. A straightforward greedy algorithm is used to divide the domain. Following is the greedy algorithm.

1. Pick all those orbitals near the moving particle i_k within a cutoff radius. In details, we make the coordinates of the moving particle i_k as the center, and then find all orbitals that are within the preset radius. The moving particle moves from an old position to a new position and we use the old position because A^{-1} is the inverse of the Slater matrix for current system not the next system.

2. For each orbital that's found in step 1, pick the closest particle from remaining particles.

3. Check if the moving particle i_k is among the selected particles which forms the local domain. If not, simply replace any particle with the moving particle.

4. All the remaining particles and orbitals form the second domain.

Let ℓ_p be the set of local particles, ℓ_q be the set of local orbitals. Suppose the number of particles (orbitals) is m. So, the cardinality of ℓ_p and ℓ_q is m. Construct R_1 and I_1 to be row and column permutation matrix. R_1 permutes rows and I_1 permutes columns. Since particles and orbitals are corresponding to rows and columns respectively, R_1 is related to local particles ℓ_p and I_1 is related to ℓ_q.

$$R_1(i, \ell_{p_j}) = \begin{cases} 1, & i = \ell_{p_j} \\ 0, & \text{otherwise} \end{cases} \quad \text{for } 1 \leq i, j \leq m;$$

$$I_1(\ell_{o_i}, j) = \begin{cases} 1, & \ell_{o_i} = j \\ 0, & \text{otherwise} \end{cases} \quad \text{for } 1 \leq i, j \leq m. \tag{3.7}$$

For instance, if local particles are $1, 3, 4, 6$ within a $2 \times 2 \times 2$ system(8 particles in total), then R_1 would be

$$R_1 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}.$$

We define R_2 and I_2 similarly for the second domain of the remaining particles and orbitals. Then we can decompose A by multiplying A from the left by the row permutation matrix $\begin{pmatrix} R_1 \\ R_2 \end{pmatrix}$, and from the right by the column permutation matrix $\begin{pmatrix} I_1 & I_2 \end{pmatrix}$.

$$\begin{pmatrix} R_1 \\ R_2 \end{pmatrix} A \begin{pmatrix} I_1 & I_2 \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}.$$

So, we can rewrite system (3.6)

$$\begin{aligned} & Az = e_{i_k} \\ \Leftrightarrow \quad & \begin{pmatrix} R_1 \\ R_2 \end{pmatrix} A \begin{pmatrix} I_1 & I_2 \end{pmatrix} \begin{pmatrix} I_1^T \\ I_2^T \end{pmatrix} z = \begin{pmatrix} R_1 \\ R_2 \end{pmatrix} e_{i_k} \\ \Leftrightarrow \quad & \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \tilde{z} = \tilde{e}_{i_k}, \end{aligned} \tag{3.8}$$

where $A_{ij} = R_i A I_j$, for $1 \leq i, j \leq 2$, $\tilde{z} = \begin{pmatrix} I_1^T \\ I_2^T \end{pmatrix} z$, and $\tilde{e}_{i_k} = \begin{pmatrix} R_1 \\ R_2 \end{pmatrix} e_{i_k}$.

Define $P_i = I_i(R_i A I_i)^{-1} R_i = I_i (A_{ii})^{-1} R_i, i = 1, 2$. Then the iteration process with two-level domain decomposition preconditioner is

1. $z^{(k+\frac{1}{2})} = z^{(k)} + I_1 A_{11}^{-1} R_1 (e_{i_k} - A z^{(k)})$,

2. $$\begin{aligned} z^{(k+1)} &= z^{(k+\frac{1}{2})} + I_2 A_{22}^{-1} R_2 (e_{i_k} - A z^{(k+\frac{1}{2})}) \\ &= z^{(k)} + (I_1 A_{11}^{-1} R_1 + I_2 A_{22}^{-1} R_2 - I_2 A_{22}^{-1} R_2 A I_1 A_{11}^{-1} R_1)(e_{i_k} - A z^{(k)}) \\ &= z^{(k)} + (P_1 + P_2 - P_2 A P_1)(e_{i_k} - A z^{(k)}) \end{aligned}$$.

Let $P = P_1 + P_2 - P_2 A P_1$. Then P is the preconditioner.

Numerical Results

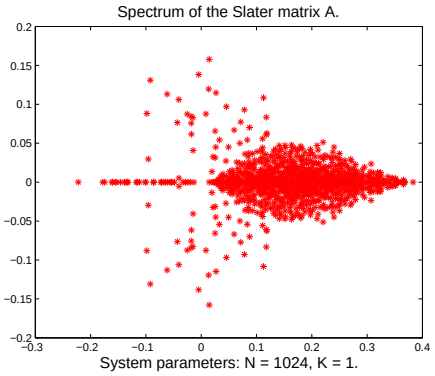
Before we present detailed numerical results, we have to say that the two-level domain decomposition preconditioner is expensive because a new preconditioner has to be computed for each moving particle. This is because the domain decomposition is closely related to the particle that is moving, which makes the recomputation of preconditioners necessary. We cannot apply cumulative rank-one updates for the two-level domain decomposition preconditioner.

We use left preconditioning, that is, we solve $PAz = Pe_{i_k}$ instead of solving $Az = e_{i_k}$. There comes a problem in respect to A_{22}^{-1} in P_2. In the following experiments, we use incomplete

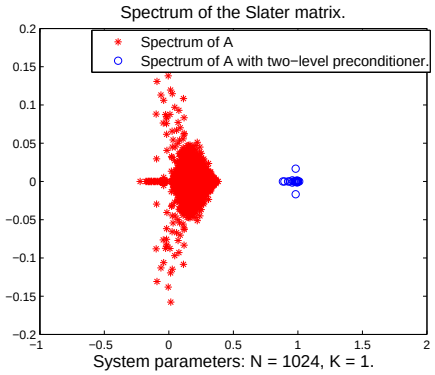
LU factorization of A_{22}^{-1}. The ILUTP produces good approximation of A_{22}. This is relatively expensive and we consider replacing ILUTP with an iterative solver in section 3.4. We run tests for systems with different size N and k. The values of N are 686, 1024, 1458, 2000, and 2662. For parameters of GMRES, the maximum number of iteration is set to be equal to the system size and tolerance is $1e-6$ for $\|r\|/\|r_0\|$.

As an example when $N = 1024$ and $K = 1$, Figure 3.4 demonstrates the effect of the preconditioner on the spectrum of a specific Slater matrix. The spectrum of preconditioned system is very well improved. The effectiveness is also demonstrated by the low number of the iterations of GMRES. In Figure 3.5, the average number of iterations is constantly around 5 after the system transits into equilibrium.

Figure 3.4: Spectrum of a Slater matrix with and without two-level multiplicative Schwarz preconditioner. The Slater matrix is picked randomly from the sequence of linear systems and the system is with $N = 1024$ and $K = 1$ in three dimensional space.

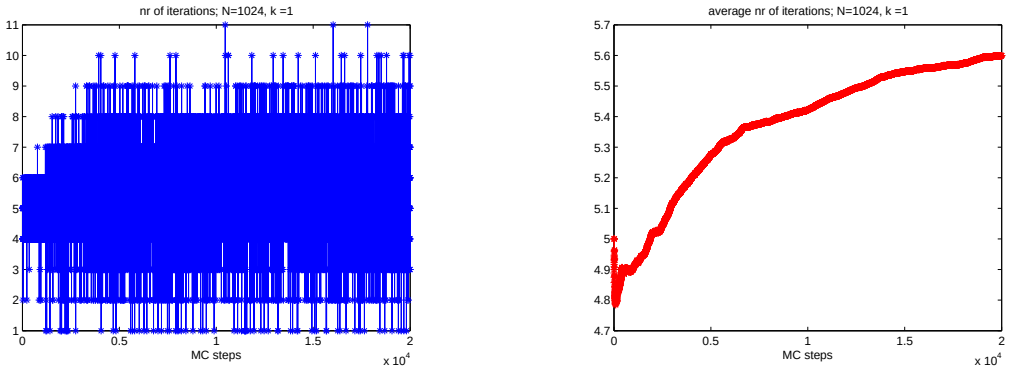


(a) Spectrum of the Slater matrix at some step



(b) Spectrum of the same Slater matrix with two-level multiplicative Schwarz preconditioner

Figure 3.5: Number of GMRES iterations



(a) Number of iterations

(b) Average number of iterations

For more systems and parameters, we use the same value in Section 3.2 to evaluate the accuracy of the simulation results. Table 3.4 shows the effectiveness for systems of different size with K = 1. For K = 0.5, the result of simulation is still good, though not as good as that with K = 1. The 'Extremely good' percentages are not very satisfactory when K = 0.5. This is because the Slater matrix becomes more ill-conditioned with small Gaussian parameter K.

Table 3.2: Test results of 50K Monte Carlo steps for different systems with K = 1

Size of system	1024	1458	2000	2662	3456
Extr. Good %	98.71	94.00	95.46	93.46	93.66
Very Good %	99.10	97.69	96.35	95.56	94.87
Good %	99.87	99.26	99.33	98.75	98.18
Acc. Ratio	0.5925	0.5732	0.6275	0.5620	0.6237

Table 3.3: Test results of 50K Monte Carlo steps for different systems with K = 0.5

Size of system	1024	1458	2000	2662
Extr. Good %	83.25	81.49	71.46	73.66
Very Good %	91.04	89.71	91.56	84.87
Good %	99.78	97.82	98.02	96.18
Acc. Ratio	0.6753	0.5879	0.5982	0.4877

3.4 Inexact Matrix-vector Products by Inner-outer GMRES

In Section 3.3, we divide the entire domain of particles and orbitals into two subdomains. The Slater matrix A is divided into $\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}$ accordingly. In this section, we solve a Schur complement system with an inexact preconditioning by inner-outer GMRES.

At every Monte Carlo step, we need to compute the squared determinant ratio

$$\left|\frac{\det A_{k+1}}{\det A_k}\right|^2 = \left|1 + u_k^T A_k^{-1} e_{i_k}\right|^2,$$

where $A_k^{-1} e_{i_k} = z_k$. For simplicity, we skip the index k that stands for the Monte Carlo step. Consider I_1 and I_2 given in (3.7). Because $\begin{pmatrix} I_1 & I_2 \end{pmatrix} \begin{pmatrix} I_1^T \\ I_2^T \end{pmatrix} = I$, we can rewrite the determinant ratio as

$$\begin{aligned} \left|1 + u^T A^{-1} e_{i_k}\right| &= \left|1 + u^T \begin{pmatrix} I_1 & I_2 \end{pmatrix} \begin{pmatrix} I_1^T \\ I_2^T \end{pmatrix} z\right| \\ &= \left|1 + \begin{pmatrix} u^T I_1 & u^T I_2 \end{pmatrix} \begin{pmatrix} I_1^T z \\ I_2^T z \end{pmatrix}\right| \\ &= \left|1 + u_1^T z_1 + u_2^T z_2\right|, \end{aligned} \tag{3.9}$$

where $u_1 = u^T I_1, u_2 = u^T I_2, z_1 = I_1^T z, z_2 = I_2^T z$. By properties of Gaussian functions, we know the component of u becomes smaller when the corresponding orbital is further from the moving particle. Therefore, we can truncate out the last term in (3.9) by setting a cutoff radius. In other words, we can select our local domain such that the product of $u_2^T z_2$ is neglectable. For example when $\|u_2\| \leq 1e-8$, we have $|u_2^T z_2| \leq 1e-5$. Because the acceptance probability is the squared determinant ratio and the square of $1e-5$ is $1e-10$, the MCMC simulation is rarely affected (the probability error due to truncation of $|u_2^T z_2|$ is $1e-10$ per Monte Carlo step). In practice, we choose our local domain such that $\|u_2\| \leq 1e-6$.

So, we only concern about u_1 and z_1 because $\left|1 + u_1^T z_1\right|$ is a practical approximation to $\left|1 + u^T A^{-1} e_{i_k}\right|$ with the truncation. By (3.8), the new system related to the partitioned Slater matrix is

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \tilde{z} = \tilde{e}_{i_k}, \tag{3.10}$$

where $\tilde{z} = \begin{pmatrix} I_1^T \\ I_2^T \end{pmatrix} z$, and $\tilde{e}_{i_k} = \begin{pmatrix} R_1 \\ R_2 \end{pmatrix} e_{i_k}$.

Let $\tilde{z} = \begin{pmatrix} I_1^T z \\ I_2^T z \end{pmatrix} = \begin{pmatrix} \tilde{z}_1 \\ \tilde{z}_2 \end{pmatrix}$, and $\tilde{e}_{i_k} = \begin{pmatrix} R_1 e_{i_k} \\ R_2 e_{i_k} \end{pmatrix} = \begin{pmatrix} e_i \\ 0 \end{pmatrix}$. Then we can rewrite (3.10) into

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} \tilde{z}_1 \\ \tilde{z}_2 \end{pmatrix} = \begin{pmatrix} e_i \\ 0 \end{pmatrix}. \tag{3.11}$$

Instead of solving for $\begin{pmatrix} z_1 \\ z_2 \end{pmatrix}$, we only look for the first component z_1, which equals $I_1 \tilde{z}_1$. This can be done by solving the Schur complement system with respect to $\tilde{z}_1$, which is

$$(A_{11} - A_{12} A_{22}^{-1} A_{21}) \tilde{z}_1 = -A_{21} A_{11}^{-1} e_i.$$

Denote $S := (A_{11} - A_{12} A_{22}^{-1} A_{21})$ and $b := -A_{21} A_{11}^{-1} e_i$. Then the Schur complement system can be rewritten as

$$S \tilde{z}_1 = b. \tag{3.12}$$

We solve this system with GMRES, which is an outer loop. To approximate A_{22}^{-1}, we utilize GMRES again, which is an inner loop. The inner-outer GMRES method is very cheap. If we use ILU or ILUTP to deal with A_{22}^{-1}, it is more expensive (especially if we need to compute a new ILUTP for A_{22} at each Monte Carlo step). Further, because A_{22} may or may not be close to diagonal dominant, ILU and ILUTP may not be a good preconditioner.

Analysis and Discussion

The outer GMRES in (3.12) yields Arnoldi relation

$$SV_m = V_{m+1}\underline{H}_m, \tag{3.13}$$

where $V_m = [v_1, v_2, \ldots, v_m]$, v_i's are basis vectors of the Krylov subspace $K^m(S, b)$, and $\underline{H}_m$ is an $(m+1) \times m$ upper Hessenberg matrix. Inexact preconditioning involves inner GMRES iterations to approximate the inverse of A_{22}. This produces inexact matrix-vector products at every iteration of outer GMRES. In other words, an error matrix E_i is generated, $i = 1, \ldots, m$.

$$\begin{aligned} & [(S+E_1)v_1 \quad (S+E_2)v_2 \quad \cdots \quad (S+E_m)v_m] = V_{m+1}\underline{H}_m \\ \Leftrightarrow \qquad & SV_m + [E_1v_1 \quad E_2v_2 \quad \cdots \quad E_mv_m] = V_{m+1}\underline{H}_m. \end{aligned}$$

Instead of real (exact) residual r_i, we obtain computed residual $\widetilde{r}_i$, $i = 0, 1, \ldots, m$. Suppose $V_m y_m$ is the approximate solution to (3.12). Then

$$\begin{aligned} \widetilde{r}_m &= r_0 - SV_m y_m \\ &= (r_0 - V_{m+1}H_m y_m) + [E_1v_1, E_2v_2, \ldots, E_mv_m]\, y_m \\ &= r_m + [E_1v_1, E_2v_2, \ldots, E_mv_m]\, y_m. \end{aligned}$$

Before we continue, we introduce some results on the bounds of residual and convergence of inexact GMRES. Theorem 5 and 6 are from Simoncini and Szyld [76].

Theorem 5. *Assume we solve $S\tilde{z}_1 = b$ with inexact inner-outer GMRES. $V_m y_m$ is the approximate solution and where $V_m = [v_1, v_2, \ldots, v_m]$, v_i's are basis of the Krylov subspace $K^m(S, b)$. Let δ_m be the difference between $\widetilde{r}_m$ and r_m. At k-th outer iteration of GMRES, an error matrix E_k is introduced. This can be represented by Arnoldi relation*

$$SV_m + [E_1v_1 \quad E_2v_2 \quad \cdots \quad E_mv_m] = V_{m+1}\underline{H}_m.$$

Suppose $y_m = \left[\eta_1^{(m)}, \eta_2^{(m)}, \ldots, \eta_m^{(m)}\right]$. Then δ_m satisfies

$$\delta_m = \|r_m - \widetilde{r}_m\| \leq \sum_{k=1}^{m} |\eta_k^{(m)}| \|E_k\|.$$

Proof. See Simoncini and Szyld [76] for proof. □

In practice, our goal is to make δ_m small. Theorem 6 provides conditions that makes δ_m meet our need.

Theorem 6. *Let $\varepsilon > 0$. Let r_m be the GMRES residual after m iterations of the inexact Arnoldi method. Let $\sigma_m(H_m)$ be the smallest singular value of H_m. Under the same hypotheses and notation of Theorem 5, if for every $k \leq m$,*

$$\|E_k\| \leq \frac{\sigma_m(H_m)}{m} \frac{1}{\|\widetilde{r}_{k-1}\|} \varepsilon, \tag{3.14}$$

then the difference between real residual r_m and computed residual $\widetilde{r}_m$ is less than ε, that is

$$\|r_m - \widetilde{r}_m\| \leq \varepsilon. \tag{3.15}$$

Proof. See Simoncini and Szyld [76] for proof. □

Therefore, to make $\delta_m \leq \varepsilon$, we need $\|E_k\|$ to satisfy (3.14). Let $\ell_m = \dfrac{\sigma_m(H_m)}{m}$. Then (3.14) becomes

$$\|E_k\| \leq \ell_m \frac{1}{\|\widetilde{r}_{k-1}\|} \varepsilon.$$

According to Simoncini and Szyld [76], setting $\ell_m = 1$ is usually not harmful.

Consider our Schur system $S\tilde{z}_1 = b$, where $S = (A_{11} - A_{12}A_{22}^{-1}A_{21})$. At k-th iteration of outer GMRES, the matrix-vector product is computed as

$$\begin{aligned} Sv_k &= (A_{11} - A_{12}A_{22}^{-1}A_{21})v_k && (3.16)\\ &= A_{11}v_k - A_{12}A_{22}^{-1}(A_{21}v_k) && (3.17)\\ &= A_{11}v_k - A_{12}z_{inner}^{(k)}. && (3.18) \end{aligned}$$

where $z_{inner}^{(k)}$ is the approximation solution to $A_{22}z = A_{21}v_k$ with inner GMRES. The computed residual of inner GMRES is $r_{inner}^{(k)} = A_{21}v_k - A_{22}z_{inner}^{(k)}$. The exact solution is $A_{22}^{-1}(A_{21}v_k)$. Therefore

$$z_{inner}^{(k)} = A_{22}^{-1}(A_{21}v_k) - A_{22}^{-1}r_{inner}^{(k)}.$$

Then the computed matrix-vector product of outer GMRES is

$$\begin{aligned} A_{11}v_k - A_{12}z_{inner}^{(k)} &= A_{11}v_k - A_{12}A_{22}^{-1}(A_{21}v_k) + A_{12}A_{22}^{-1}r_{inner}^{(k)} \\ &= Sv_k + A_{12}A_{22}^{-1}r_{inner}^{(k)} \\ &= \left(S + A_{12}A_{22}^{-1}r_{inner}^{(k)}\frac{v_k^T}{\|v_k\|^2}\right)v_k. \end{aligned}$$

Therefore at k-th iteration of outer GMRES, the backward error matrix generated by inexact inner GMRES is

$$E_k = A_{12}A_{22}^{-1}r_{inner}^{(k)}\frac{v_k^T}{\|v_k\|^2}.$$

From Arnoldi recurrence, $\|v_k\| = 1$. To obtain (3.15), we need $\|E\|$ to satisfy

$$\|E_k\| \leq \frac{\sigma_m(H_m)}{\|A_{12}A_{22}^{-1}\|m}\frac{1}{\|\widetilde{r}_{k-1}\|}\varepsilon,$$

where $\widetilde{r}_{k-1}$ is the computed residual from $(k-1)$-th iteration of outer GMRES.

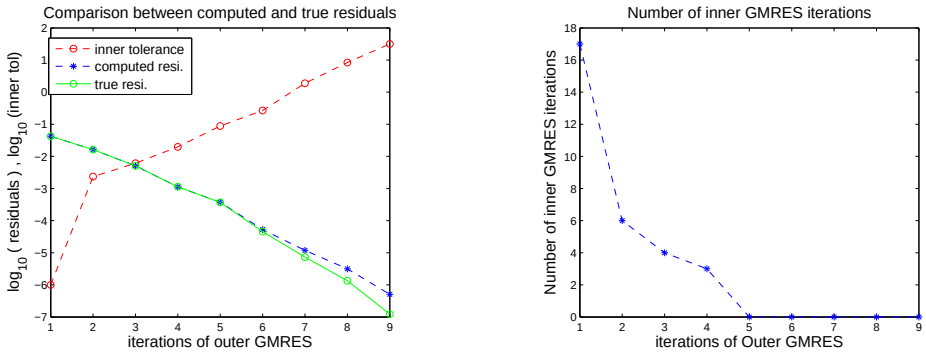
Therefore, similarly to set $\ell_m = 1$ in Simoncini and Szyld [76], we can assume $\frac{\sigma_m(H_m)}{\|A_{12}A_{22}^{-1}\|m} = 1$. We can also use an approximation of $\frac{\sigma_m(H_m)}{\|A_{12}A_{22}^{-1}\|m}$ to obtain tolerance on the residual of inner GMRES. In our experiments, we choose the former option.

Numerical Results

Although the convergence for the outer GMRES iteration is rapid, the number of iterations for inner GMRES is higher than it is with ILU preconditioned GMRES for almost the same size system as the original Slater matrix. Therefore, this inexact preconditioning is not a very cost efficient preconditioner.

As we can see from Figure 3.8, the number of total iterations for inner GMRES isn't satisfactory. This is one result when A_{22} may not be close to diagonal dominance. Figure 3.6 and 3.7 show the number of iterations for inner GMRES and the relationship between inner tolerance and residuals.

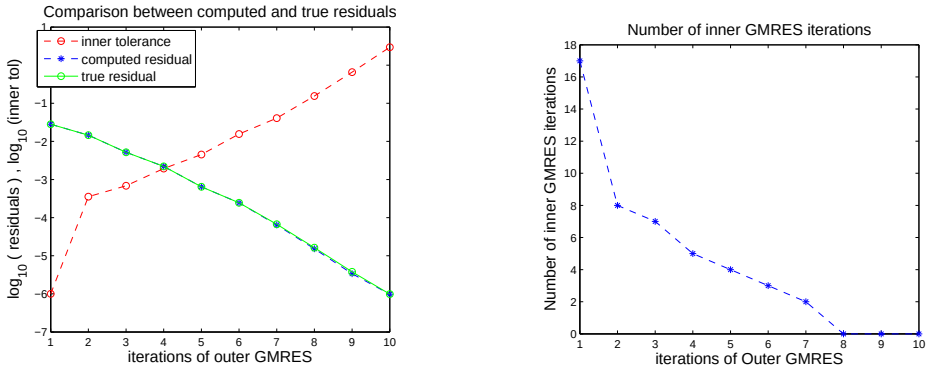
Figure 3.6: (1) Number of iterations for a system with N = 1024, K = 1 at a representative Monte Carlo step. Figure (a) is using dynamic inner tolerance.



(a) Comparison between computed and true residuals

(b) Number of inner GMRES iterations for each outer GMRES iteration

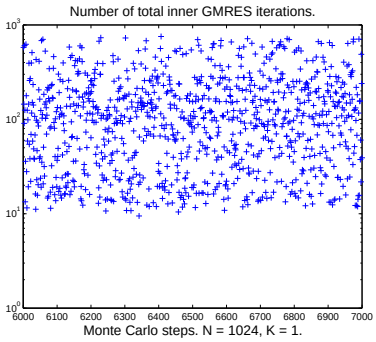
Figure 3.7: (2) Number of iterations for a system with N = 1024, K = 1 at a representative Monte Carlo step. Figure (a) is using dynamic inner tolerance.



(a) Comparison between computed and true residuals

(b) Number of inner GMRES iterations for each outer GMRES iteration

Figure 3.8: Number of total inner GMRES iterations per Monte Carlo step for a system with N = 1024, K = 1.



Chapter 4

Truncation of Preconditioner Updates

Many applications require the solution of a sequence of linear systems of the form

$$A_i x_i = b_i \quad i = 0, 1, 2, \cdots , \tag{4.1}$$

where A_i's are nonsingular matrices, b_i's are right-hand sides. To solve the sequence of linear systems, Krylov subspace methods especially with recycling preconditioners have become popular techniques and people might adapt the Krylov solver according to the problem at hand [76, 68]. In order to be efficient and robust, Krylov subspace methods need to be preconditioned. This causes the problem of computing a sequence of preconditioners corresponding to the sequence of linear systems. For example, when $i = 0$, an initial preconditioner P_0 needs to computed for solving $A_0 x_0 = b_0$. In general, for system i, a preconditioner P_i needs to be obtained for solving $A_i x_i = b_i$.

However, computing preconditioners $P_0, P_1, P_2, \ldots$ for every single system separately can be very expensive and impractical. There is a strong need to reduce the overall cost of preconditioning these linear systems. Therefore, most effective approaches adopt a middle ground by updating the existing preconditioner and reusing it for successive linear systems [36, 11, 2, 66], which we can call recycling preconditioners.

In Ahuja et al. [2, 1], preconditioners are multiplied by cumulative inverses of rank-one

updates. Sherman–Morrison formula is used to obtain the inverse of rank-one or low-rank updates. Similarly in Bergamaschi et al. [11], Broyden-type rank-one updates are used to update the preconditioners. These are two common ways to update the existing preconditioner and reuse it for successive linear systems. The advantage of these approaches is that they are cheaper than computing a new preconditioner for each system and more effective than using the same preconditioner for all linear systems. However, due to the cumulative updates to the preconditioner, the cost of applying the preconditioner increases. At some point, the cost of applying a preconditioner is even higher than computing a new preconditioner. Therefore, in Ahuja et al. [2, 1] and Bergamaschi et al. [11], a new preconditioner is computed from scratch periodically.

However, as the number of linear systems increases, the cost of applying the preconditioner becomes too expensive. In this chapter, we propose to overcome this shortcoming by effectively truncating the preconditioner updates and hence reducing the cost for performing preconditioner updates. We will propose two approaches for truncating the accumulated updates. Details of these procedures follow in the subsequent Sections.

This chapter is organized as follows. In Section 4.1, we explain the aforementioned truncation methods in detail. Section 4.2 demonstrates the effectiveness of these methods for the application in discretized nonlinear partial differential equations, such as convection-diffusion problems. Section 4.3 presents the results for a large number of steps in the quantum Monte Carlo problem.

4.1 Introduction

In this thesis we will consider sequences of nonsymmetric linear systems. We are mainly interested in addressing two issues associated with recycling the preconditioners: First, how can we make the updated preconditioner as powerful as the originally computed one? Second, how can we update and apply the preconditioners inexpensively?

Consider a sequence of linear systems (4.1). Suppose initially we compute a preconditioner P_0 for A_0. Notice that A_0, $P_0 \in \mathbb{R}^{n\times n}$. The system matrices change by rank-1 updates [26, 64]. For example, the matrix A_0 is modified by the rank-one update $p_1 q_1^T$,

$$\begin{aligned} A_1 &= A_0 + p_1 q_1^T \\ &= A_0(I + A_0^{-1} p_1 q_1^T) \\ &= A_0(I + z_1 q_1^T), \end{aligned}$$

where $z_1 := A_0^{-1} p_1$ and I is the identity matrix that has the same dimension as A_0. Note that we aim to maintain preconditioned matrices such that $A_0 P_0 = A_1 P_1 = \cdots$. As a result, for the next system $A_1 x_1 = b_1$, we want to update P_0 accordingly. This produces a new preconditioner P_1.

$$P_1 = (I + z_1 q_1^T)^{-1} P_0. \tag{4.2}$$

Using Sherman-Morrison formula, we get

$$(I + z_1 q_1^T)^{-1} = I - (1 + q_1^T z_1)^{-1} z_1 q_1^T.$$

Then we could rewrite P_1 as

$$P_1 = (I - (1 + q_1^T z_1)^{-1} z_1 q_1^T) P_0 = (I - w_1 q_1^T) P_0,$$

where $w_1 = (1 + q_1^T z_1)^{-1} z_1$. The preconditioned system for A_1 is $A_1 P_1 \tilde{x}_1 = b_1$. After m rank-1 updates to the matrix, we have the following relationship:

$$A_m = A_0(I + z_1 q_1^T)(I + z_2 q_2^T) \cdots (I + z_m q_m^T). \tag{4.3}$$

Correspondingly, the new preconditioner P_m is obtained by multiplying P_0 with inverses of rank-one updates to the identity as

$$P_m := (I - w_m q_m^T) \cdots (I - w_2 q_2^T)(I - w_1 q_1^T) P_0. \tag{4.4}$$

The preconditioner update is a product of inverses of rank-one updates to the identity which we can denote as T_m.

$$\begin{aligned} T_m &:= (I - w_m q_m^T) \cdots (I - w_2 q_2^T)(I - w_1 q_1^T) \\ &= I + W_m B_m Q_m^T. \end{aligned} \tag{4.5}$$

where B_m is a rank-m $m \times m$ lower triangular matrix, $W_m = [w_1, w_2, \ldots, w_m]_{n\times m}$, and $Q_m = [q_1, q_2, \ldots, q_m]_{n\times m}$. For simplicity, let

$$X_m = W_m B_m Q_m^T. \tag{4.6}$$

X_m is a low-rank matrix that contains the information of the rank-one updates to the preconditioner.

So $P_m = T_m P_0$ and the preconditioned system after m rank-one updates is $A_m P_m \widetilde{x}_m = b_m$.

The newly obtained preconditioners $P_1, P_2, \ldots, P_m$ are as good as the originally computed preconditioner P_0 because the relation (1.3) is obtained. This approach maintains the quality of the original preconditioned matrix. However, applying these preconditioners is not cost effective. In every linear system or nonlinear step, multiplying a vector with the preconditioner becomes increasingly expensive because of the cumulative cost of multiplying by the preconditioner updates. In other words, the cost of multiplying a vector by a preconditioner P_m with m preconditioner updates (the rank of X_m is m) is $O(4mN)$ plus the cost of multiplying P_0 [53]. Over m Monte Carlo or nonlinear steps, the cost is $O(m^2 N)$. At some point the cost of multiplying vectors by the preconditioner update T_m is higher than that of computing a new preconditioner. For that reason, in Ahuja et al. [2] and Bergamaschi et al. [11] a new preconditioner is computed from scratch periodically. However, computing a (very) good new preconditioner from scratch is quite expensive too; therefore, we propose two methods for truncating the accumulated preconditioner updates. Therefore the cost of applying the preconditioner is reduced or even bounded to $O(N)$. Further, the computation of a new preconditioner is postponed or even avoided. This can be fulfilled by making truncation whenever the rank of T is larger than or equal to a preset threshold.

However, due to truncation, the number of iterations of GMRES may not be as low as that without truncation. Assume we discard some directions from X_m. This produces the truncated preconditioner $\widetilde{P}_i$, where $\widetilde{P}_i = P_i - \Delta P_i$, where ΔP_i is the part we discard from P_i. As a result, instead of (1.3), the preconditioned system matrix $A_m \widetilde{P}_m$ satisfies

$$A_0 P_0 = \cdots = A_{m-1} P_{m-1} \quad = \quad A_m P_m = A_m(\widetilde{P}_m + \Delta P_m) \approx A_m \widetilde{P}_m. \tag{4.7}$$

Our goal is to perform a truncation without drastically impacting the rate of convergence.

We consider two approaches to truncate X_m. The first approach is based on the SVD of X_m. This approach is effective if the SVD has relatively many small singular values. In other words, SVD based truncation will work well if the matrix can be approximated by using a small number of singular triplets. This turns out to be case for the nonlinear PDE problem discussed in Section 4.2. If there are few relatively small singular values, an SVD based truncation may lead to a substantial increase in the number of iterations. In that case, we consider an alternative strategy. Given $T_m = I + W_m B_m Q_m^T$, we consider the canonical angles between Range($Q_m P_0$) and a previous Krylov space (see below for more details), and we truncate directions that are associated with large canonical angles. The main idea behind this approach is that in an Arnoldi recurrence

$$A_m(I + W_m B_m Q_m^T) P_0 V_\ell = V_{\ell+1} \underline{\mathrm{H}}_\ell, \tag{4.8}$$

directions in Range(Q_m) that have a large (near $\pi/2$) angle with Range($P_0 V_\ell$) have little effect on the Arnoldi recurrence.

In the next subsections, we discuss our two truncation approaches and provide some theoretical analysis.

Truncation by SVD

This approach is used to obtain a rank-p approximation from a rank-m update of the preconditioner by using singular value decompositions, where $p < m$. For example, in the QMC problem in Section 4.3, we set $p = 30$ and $m = 50$. Consider the update matrix T_m of the preconditioner P_m after m rank-one updates

$$T_m = I + W_m B_m Q_m^T. \tag{4.9}$$

To simplify the notation, define $X_m := W_m B_m Q_m^T$. X_m is a rank-m matrix. We first obtain the singular value decomposition of X_m.

$$\begin{aligned} X_m &= \Phi\Omega\Psi^T \\ &= \sum_{i=1}^{m} \sigma_i u_i v_i^T, \end{aligned} \tag{4.10}$$

where u_i's and v_i's are the left and right singular vectors, and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_m > 0$. The idea of truncation by SVD is to truncate X_m into a smaller rank matrix. Therefore, we keep the first p largest singular values and their corresponding singular vectors, and discard the remaining. By doing so, we get rank-p approximation of X_m, which can be represented by

$$\widetilde{X}_m = \sum_{i=1}^{p} \sigma_i u_i v_i^T, \tag{4.11}$$

For simplicity, define

$$E := \sum_{i=p+1}^{m} \sigma_i u_i v_i^T = X_m - \widetilde{X}_m. \tag{4.12}$$

Then $\text{rank}(E) = m - p$ and $\|E\|_2 = \sigma_{p+1}$. By discarding E from X_m, we convert X_m into $\widetilde{X}_m$. Further, the new preconditioner update with $\widetilde{X}_m$ is

$$\widetilde{T}_m = I + \widetilde{X}_m, \tag{4.13}$$

and the new preconditioner after truncation equals

$$\widetilde{P}_m = \widetilde{T}_m P_0. \tag{4.14}$$

The resulting preconditioned system after truncation is

$$A_m \widetilde{P}_m \widetilde{x}_m = b_m. \tag{4.15}$$

Analysis of SVD-based Truncation

Consider the preconditioned system (4.15) after SVD based truncation at step m. From (4.9)-(4.14) it follows that

$$\begin{aligned} A_m\widetilde{P}_m &= A_m\widetilde{T}_m P_0 \\ &= A_m\left(T_m - \sum_{i=p+1}^{m}\sigma_i u_i v_i^T\right)P_0 \\ &= A_m T_m P_0 - A_m\left(\sum_{i=p+1}^{m}\sigma_i u_i v_i^T\right)P_0 \\ &= A_m P_m - A_m E P_0. \end{aligned} \tag{4.16}$$

In some cases, people compute one exact inverse for generating preconditioners, say $A_0P_0 = I$. For example, in Meerbergen and Bai [61], an initial preconditioner $K_\sigma^{-1} = (K - \sigma M)^{-1}$ is used to precondition a system $Ax = f$, where $A = K - \omega^2 M$ and K, M are large sparse symmetric matrices. Note that for matrices that change exactly by rank-one updates, starting with $A_0P_0 = I$ and no truncation leads a direct method [53, 65]. This becomes too expensive if we need to handle many (nonlinear) steps. Hence our truncation plus a cheap iteration can be a good alternative. For the system (4.15) after our truncation, the following theorem shows that GMRES will converge fast if the initial preconditioner P_0 is such that $A_0P_0 = I$.

Theorem 7. *Let P_m given by* (4.4)*, $\widetilde{P}_m$ given by* (4.14)*, and E given by* (4.12)*. Let $A_m \in \mathbb{R}^{N\times N}$ and $\widetilde{P}_m$ be nonsingular, and $A_0P_0 = I$. Then GMRES converges at most $d+1$ steps for the system* (4.15)*, where d is the rank of E.*

Proof. Since A_m and $\widetilde{P}_m$ are nonsingular, $A_m\widetilde{P}_m$ is nonsingular and thus GMRES converges for $A_m\widetilde{P}_m\widetilde{x}_m = b$ by Theorem 3.1.2. in Kelley [52]. $A_0P_0 = I$ implies P_0 is nonsingular and thus $\mathrm{rank}(P_0) = N$. Since A_m is also nonsingular and $\mathrm{rank}(E) = d < N$, it follows that

$$\begin{aligned} \mathrm{rank}(A_m E P_0) &\leq \min\{\mathrm{rank}(A_m), \mathrm{rank}(E), \mathrm{rank}(P_0)\} \\ &= \mathrm{rank}(E) = d. \end{aligned}$$

This further implies $\text{rank}((A_mEP_0)^\ell) \leq \text{rank}(E) = d \quad \text{for} \quad \ell = 1, 2, \cdots$. Consider the Krylov subspace $K^\ell(A_mEP_0, r_0) = \text{span}\left\{r_0, (A_mEP_0)r_0, \cdots, (A_mEP_0)^{\ell-1}r_0\right\}$.

Since $\text{rank}((A_mEP_0)^\ell) \leq d$ and r_0 may or may not be in Range(A_mE), it follows that

$$\dim K^\ell(A_mEP_0, r_0) \leq d + 1. \tag{4.17}$$

With (4.15), (4.16), and $A_0P_0 = I$, we have $A_m\widetilde{P}_m = A_mP_m - A_mEP_0 = I - A_mEP_0$. For any Krylov subspace $K^\ell(B, r)$, it follows from the definition that $K^\ell(I - B, r) = K^\ell(B, r)$. Therefore,

$$\begin{aligned} \dim K^N(A_m\widetilde{P}_m, r_0) &= \dim K^N(I - A_mEP_0, r_0) \\ &= \dim K^N(A_mEP_0, r_0) \\ &\leq d + 1. \end{aligned} \tag{4.18}$$

So $K^N(A_m\widetilde{P}_m, r_0) \subset K^{d+1}(A_m\widetilde{P}_m, r_0)$ (see Proposition 6.1 in Saad [72]). Because GMRES is guaranteed to converge, the solution lies in $K^N(A_m\widetilde{P}_m, r_0) \subset K^{d+1}(A_m\widetilde{P}_m, r_0)$. It follows that GMRES converges within at most $d + 1$ steps for the system (4.15). □

In most cases, it is sufficient to compute a preconditioner such that $A_0P_0 = I + C$ with $\|C\|$ small (see Theorem 2). In this situation, since $A_mP_m = A_0P_0$ and (4.16), truncation by SVD produces

$$A_m\widetilde{P}_m = I + C - A_mEP_0. \tag{4.19}$$

Because $\|C\|$ is determined by the choice and quality of the preconditioner, we can make $\|C\|$ small such that $\|C\| \leq \delta < 1$. By (4.12), we have $\|E\| \leq \sigma_{p+1}$. If $\delta + \sigma_{p+1} < 1$, then theorem 2 applies and GMRES is guaranteed to converge for the system (4.15).

When $A_0P_0 = I + C$, we could also make use of theorem 3 and error bound (3.2) to prove the convergence of GMRES. Still, let $\|C\| \leq \delta < 1$. Then the field of values of A_mP_m is included in a small disk, $\mathcal{F}(A_mP_m) \subset \{z \in \mathbb{C} : |z - 1| \leq \delta, \quad \delta < 1\}$. Based on (4.19): $A_m\widetilde{P}_m = (I + C) - A_mEP_0$, if $\|A_mEP_0\| = \varepsilon < 1 - \delta$, it follows from theorem 3 that $\mathcal{F}(A_m\widetilde{P}_m) \subset$

$\{z \in \mathbb{C} : |z-1| \leq \delta + \varepsilon, \quad \delta + \varepsilon < 1\}$. This guarantees the convergence of GMRES and gives a bound on the residual r_k at k-th iteration:

$$\|r_k\|/\|r_0\| \leq 2\left(\delta + \varepsilon\right)^k. \tag{4.20}$$

$\|A_m E P_0\|$ is closely related to the singular values we discard. If $\|A_m E P_0\|$ follows directly from the singular values, truncation by SVD is expected to work very well. This is demonstrated by the following application of nonlinear partial differential equations in Section 4.2.

Truncation Based on Canonical Angles

In some applications, the singular values decrease drastically and we discard the small ones. So the truncation by SVD works well. Unfortunately, this is not always the case. For example, in the Quantum Monte Carlo problem in Section 4.3, the singular values of X_m do not change much and we have very few small singular values to discard (see Figure 4.2). This is why we propose the second truncation approach by canonical angles.

Consider the system we need to truncate: $A_m P_m \widetilde{x}_m = b_m$. With Krylov subspace methods (GMRES), we have the Arnoldi relationship

$$A_m P_m V_\ell = V_{\ell+1} H_\ell. \tag{4.21}$$

The residual r_ℓ of GMRES is $r_0 - A_m P_m V_\ell y_\ell$, where y_ℓ is a column vector such that $x_\ell - x_0 = V_\ell y_\ell$. With (4.4) (4.5), and (4.10), r_ℓ can be rewritten as

$$\begin{aligned} r_\ell &= r_0 - A_m\left(I + W_m B_m Q_m^T\right) P_0 V_\ell y_\ell \\ &= r_0 - A_m\left(I + \sum_{i=1}^{m} \sigma_i u_i v_i^T\right) P_0 V_\ell y_\ell. \end{aligned} \tag{4.22}$$

Consider when $v_{i_0}^T P_0 V_\ell = 0$. We have

$$A_m\left(I + \sum_{i=1}^{m} \sigma_i u_i v_i^T\right) P_0 V_\ell = A_m\left(I + \sum_{i=1, i\neq i_0}^{m} \sigma_i u_i v_i^T\right) P_0 V_\ell. \tag{4.23}$$

If we discard σ_{i_0}, u_{i_0}, and v_{i_0}, it follows from (4.23) that

$$\begin{aligned} r_\ell &= r_0 - A_m \left(I + \sum_{i=1, i\neq i_0}^{m} \sigma_i u_i v_i^T \right) P_0 V_\ell y_\ell \\ &= \widetilde{r}_\ell, \end{aligned} \tag{4.24}$$

where $\widetilde{r}_\ell$ is the residual with respect to the truncated system (4.15). Therefore the residual does not change with the truncation of σ_{i_0}, u_{i_0}, and v_{i_0}. This example motivates our second truncation approach.

If we look into the angles between each direction of Range(Q_m) and Range($P_0 V_l$), those directions orthogonal or nearly orthogonal to $P_0 V_\ell$ have less impact on the residual. This suggests us discard those directions as an alternate truncation approach. In other words, those directions with the angle $\angle(\cdot, P_0 V_\ell)$ equal or close to $\pi/2$ are discarded.

Now we explain the second truncation step by step. Our goal is to truncate rank-m $W_m B_m Q_m^T$ into rank-p $\widetilde{W}_m \widetilde{U}_m^T$. We first orthonormalize Q_m and $P_0 V_l$ by QR-factorization, $Q_m = \hat{U}_m \hat{R}_m$, $P_0 V_\ell = \hat{Y}\hat{S}$. Then $Q_m^T P_0 V_\ell = \hat{R}_m^T \hat{U}_m^T \hat{Y}\hat{S}$ and we compute the singular value decomposition of $\hat{U}_m^T \hat{Y}$.

$$\hat{U_m}^T \hat{Y} = \Phi \Omega \Psi^T, \tag{4.25}$$

where the diagonal of Ω consists of all singular values $\omega_1 \geq \omega_2 \geq \cdots \geq \omega_m > 0$. We keep p dominant singular values and their corresponding vectors and thus the truncation of $\hat{U}_m$ is $\widetilde{U}_m = \hat{U}_m \Phi_p$, where $\widetilde{U}_m \in \mathbb{R}^{N\times p}$ and Φ_p corresponds to the vectors we keep. Let Φ_c be the part we throw away in Φ. Then it follows that $\Phi_p \Phi_p^T + \Phi_c \Phi_c^T = I_m$ and $\|\Phi_c^T \hat{U}_m^T \hat{Y}\| \leq \omega_{p+1}$.

Therefore $\|W_m B_m Q_m^T P_0 V_\ell\|$ can be written as

$$\begin{aligned} \|W_m B_m Q_m^T P_0 V_\ell\| &= \|W_m B_m \hat{R}_m^T \left(\Phi_p \Phi_p^T + \Phi_c \Phi_c^T \right) \hat{U}_m^T \hat{Y}\hat{S}\| \\ &= \|W_m B_m \hat{R}_m^T \Phi_p \widetilde{U}_m \hat{Y}\hat{S} + W_m B_m \hat{R}_m^T \Phi_c \Phi_c^T \hat{U}_m^T \hat{Y}\hat{S}\| \qquad (4.26) \\ &\leq \|W_m B_m \hat{R}_m^T \Phi_p \widetilde{U}_m \hat{Y}\hat{S}\| + \omega_{p+1} \|W_m B_m \hat{R}_m^T \Phi_c\|. \qquad (4.27) \end{aligned}$$

By discarding the second term in (4.27), $W_m B_m \hat{R}_m^T \Phi_c \Phi_c^T \hat{U}_m^T$, the new truncation of $W_m B_m Q_m^T$ is $\left(W_m B_m \hat{R}_m^T \Phi_p \right) \widetilde{U}_m^T$, where $\widetilde{U}_m = \hat{U}_m \Phi_p$. Define $\widetilde{W}_m := W_m B_m \hat{R}_m^T \Phi_p$ and thus $\widetilde{W}_m \widetilde{U}_m^T$ is a

rank-p update. At last, the new preconditioner after truncation is

$$\widetilde{P}_m = (I + \widetilde{W}_m \widetilde{U}_m^T) P_0. \tag{4.28}$$

The resulting preconditioned system is the same as (4.15) with the SVD-based truncation:

$$A_m \widetilde{P}_m \widetilde{x}_m = b_m. \tag{4.29}$$

Analysis of Truncation by Canonical Angles

We know that $W_m B_m Q_m^T$ is truncated into $\widetilde{W}_m \widetilde{U}_m^T$. Let $W_m B_m Q_m^T = \widetilde{W}_m \widetilde{U}_m^T + E$. By (4.28), we have

$$A_m P_m = A_m \widetilde{P}_m + A_m E P_0. \tag{4.30}$$

As we mentioned before (see (4.23)), if $EP_0V_\ell = 0$, our truncation produces the same convergence because the Arnoldi recurrence and the residual do not change. If $EP_0V_\ell \neq 0$, our truncation introduces approximate Arnoldi recurrence and residual. However, it is reasonable to assume EP_0V_ℓ is small due to our truncation by canonical angles. We consider using the same basis vectors of the Krylov subspaces from the original system, that is V_ℓ.

$$(A_m \widetilde{P}_m + A_m E P_0) V_\ell = V_{\ell+1} H_\ell \tag{4.31}$$

$$\Leftrightarrow \quad A_m \widetilde{P}_m V_\ell = V_{\ell+1} H_\ell - A_m E P_0 V_\ell \tag{4.32}$$

$$\Leftrightarrow \quad A_m \widetilde{P}_m V_\ell = V_{\ell+1} H_\ell - (I - V_{\ell+1} + V_{\ell+1}) A_m E P_0 V_\ell \tag{4.33}$$

$$\Leftrightarrow \quad A_m \widetilde{P}_m V_\ell = V_{\ell+1}(H_\ell - A_m E P_0 V_\ell) + (I - V_{\ell+1}) A_m E P_0 V_\ell \tag{4.34}$$

Suppose $(I - V_{\ell+1}) A_m E P_0 V_\ell = 0$. Then for the truncated system, we have

$$A_m \widetilde{P}_m V_\ell = V_{\ell+1} \widetilde{H}_\ell, \tag{4.35}$$

and

$$\widetilde{H}_\ell = H_\ell - A_m E P_0 V_\ell. \tag{4.36}$$

Let $\widetilde{E} := A_m E P_0 V_\ell$. Then $H_\ell = \widetilde{H}_\ell + \widetilde{E}$ and $\widetilde{E}$ can be seen as a perturbation matrix to the Hessenburg matrix H_ℓ because of the truncation. Based on the same notation and assumption, we could get the following theorem, which gives bound between the two residuals. The bound also provides the condition for GMRES convergence.

Theorem 8. *Let $\varepsilon > 0$. Let r_ℓ be GMRES residuals for the system without truncation. Let $\widetilde{r}_\ell$ be the residual for the truncated system under same Krylov subspaces. Let $\sigma(H_\ell)$ be the smallest singular value of H_ℓ and $\|A_m P_m\| \leq 1 + \delta$. If*

$$\|E P_0 V_\ell\| \leq \frac{\sigma(H_\ell)}{\|A_m\| \|r_0\|} \varepsilon,$$

then $\|r_\ell - \widetilde{r}_\ell\| \leq \gamma\varepsilon$, where $\gamma = (1+\delta)\alpha + 1 + \dfrac{\sigma(H_\ell)}{\|r_0\|}\varepsilon$.

Proof. By (4.21) and (4.24), we have

$$\begin{aligned} r_\ell - \widetilde{r}_\ell &= (r_0 - A_m P_m V_\ell y_\ell) - \left(r_0 - A_m \widetilde{P}_m V_\ell \widetilde{y}_\ell\right) && (4.37)\\ &= \left(A_m \widetilde{P}_m + A_m E P_0\right) V_\ell y_\ell - A_m \widetilde{P}_m V_\ell \widetilde{y}_\ell && (4.38)\\ &= A_m \widetilde{P}_m V_\ell \left(y_\ell - \widetilde{y}_\ell\right) + A_m E P_0 V_\ell y_\ell, && (4.39) \end{aligned}$$

where y_ℓ minimizes $\|e_1\beta - H_\ell y\|$, $\beta = \|r_0\|$, and $\widetilde{y}_\ell$ minimizes $\|e_1\beta - \widetilde{H}_\ell \widetilde{y}\|$. $y_\ell = \left(H_\ell^T H_\ell\right)^{-1} H_\ell e_1 \beta$. It follows from (4.36) that

$$\left(H_\ell^T H_\ell + H_\ell^T \widetilde{E} + \widetilde{E}^T H_\ell + \widetilde{E}^T \widetilde{E}\right) \widetilde{y}_\ell = \left(H_\ell + \widetilde{E}\right) e_1 \beta. \tag{4.40}$$

If we subtract $\left(H_\ell^T H_\ell\right) y_\ell = H_\ell e_1 \beta$ from (4.40), we have

$$H_\ell^T H_\ell \left(y_\ell - \widetilde{y}_\ell\right) = \left(-H_\ell^T \widetilde{E} - \widetilde{E}^T H_\ell - \widetilde{E}^T \widetilde{E}\right) \widetilde{y}_\ell + \widetilde{E} e_1 \beta. \tag{4.41}$$

By Lemma 5.1 in Simoncini and Szyld [76], we have $\|y_\ell\| \leq \dfrac{1}{\sigma(H_\ell)} \|r_0\|$ and $\|\widetilde{y}_\ell\| \leq \dfrac{1}{\sigma(\widetilde{H}_\ell)} \|r_0\|$. As a result, if $\|E P_0 V_\ell\| \leq \dfrac{\sigma(H_\ell)}{\|A_m\| \|r_0\|}\varepsilon$, it follows that

$$\|y_\ell - \widetilde{y}_\ell\| \leq \left(\frac{1}{\sigma(H_\ell)} + \frac{2}{\sigma(H_\ell)^2 \sigma(\widetilde{H}_\ell) \|r_0\|}\right) \varepsilon. \tag{4.42}$$

Let $\alpha := \dfrac{1}{\sigma(H_\ell)} + \dfrac{2}{\sigma(H_\ell)^2\sigma(\widetilde{H}_\ell)\|r_0\|}$. Then $\|y_\ell - \widetilde{y}_\ell\| \le \alpha\varepsilon$. Therefore,

$$\begin{aligned}
\|r_\ell - \widetilde{r}_\ell\| &\le \|A_m\widetilde{P}_m V_\ell\|\|y_\ell - \widetilde{y}_\ell\| + \|A_m E P_0 V_\ell y_\ell\| \\
&= \|A_m P_m V_\ell + A_m E P_0 V_\ell\|\|y_\ell - \widetilde{y}_\ell\| + \|A_m E P_0 V_\ell y_\ell\| \\
&\le \left(1 + \delta + \|A_m\|\frac{\sigma(H_\ell)}{\|A_m\|\|r_0\|}\varepsilon\right)\alpha\varepsilon + \|A_m\|\frac{\sigma(H_\ell)}{\|A_m\|\|r_0\|}\frac{1}{\sigma(H_\ell)}\|r_0\|\varepsilon \\
&= ((1+\delta)\alpha + 1)\varepsilon + \frac{\sigma(H_\ell)}{\|r_0\|}\varepsilon^2.
\end{aligned}$$

Let $\gamma = (1+\delta)\alpha + 1 + \dfrac{\sigma(H_\ell)}{\|r_0\|}\varepsilon$. Then $\|r_\ell - \widetilde{r}_\ell\| \le \gamma\varepsilon$. □

Further, it follows from (4.27) that $\|EP_0V_\ell\| \le \omega_{p+1}\|W_m B_m \hat{R}_m^T \Phi_c\|$. Let $\widetilde{\omega}_{p+1} := \omega_{p+1}\|W_m B_m \hat{R}_m^T \Phi_c\|$. Then as long as $\widetilde{\omega}_{p+1} \le \dfrac{\sigma(H_\ell)}{\|A_m\|\|r_0\|}\varepsilon$, we have $\|r_\ell - \widetilde{r}_\ell\| \le \gamma\varepsilon$.

Theorem 8 shows that when $\|EP_0V_\ell\|$ is small, the truncation by canonical angles has competitive convergence as the original system without truncation. For example, in the quantum Monte Carlo problem in Section 4.3, the spectrum of the Hessenburg matrix does not change much (see Figure 4.1) and thus the truncation by canonical angles works well.

4.2 Application to a Nonlinear Convection-diffusion Problem

We consider a convection-diffusion equation on $[0,1]^2$:

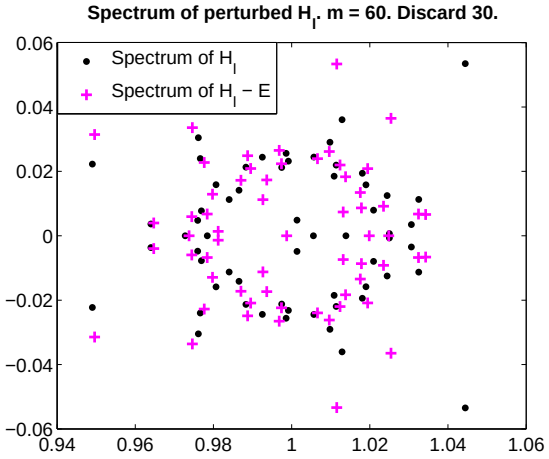
$$-\nabla\cdot(d(\eta + \gamma u^2)\nabla u) + ru_x + su_y + tu = f,$$

where d is the diffusion coefficient, f is the source (or sink). The boundary condition is

$$u(0,y) = 0.2 + y(1-y^2); \quad u(1,y) = u(x,0) = u(x,1) = 0.$$

This problem is a combination of the diffusion and convection equations. It describes physical phenomena where particles, energy, or other physical quantities are transferred inside a

Figure 4.1: Spectrum of H_ℓ and $H_\ell - \widetilde{E}$ in a specific truncation step for a system with $N = 1024$, $K = 1$, and $m = 60$. H_ℓ is the Hessenburg matrix. $\widetilde{E}$ is the perturbation matrix to H_ℓ, $\widetilde{E} = V_\ell^{\mathrm{T}} A_m E P_0 V_\ell$. In this figure, we discard 40 vectors that are orthogonal or close to orthogonal to $P_0 V_\ell$.



physical system due to two processes: diffusion and convection. We use finite difference method to get the system matrix $A(u)$ and apply Newton-Line search method to solve the discretized equation.

$$F(u) = A(u) \cdot u - f = 0.$$

The diffusion part $A(u) \cdot u$ is nonlinear. Therefore, the preconditioner needs to be updated when solving every Newton equation iteratively. Work in [11, 23, 29] has showed that Broyden-type rank-one update for quasi-Newton preconditioners is useful. However, there is still a need to reduce the cost of applying preconditioners if there are many nonlinear steps.

Truncation by SVD

Initially we compute a preconditioner P_0. Every Newton step we solve

$$J_k(x_{k+1} - x_k) = -F(x_k), \quad k = 0, 1, \ldots,$$

iteratively with a preconditioner P_k, where J_k is the Jacobian at this step. Let $s_k = x_{k+1} - x_k$ be the update to the solution and $y_k = F(x_{k+1}) - F(x_k)$ be the change of values of F.

Based on the Broyden-type rank-1 updates and Sherman-Morrison formula [11, 29], P_k comes from multiplying P_0 by Broyden-type rank-one updates

$$P_k = (I - w_{k-1}v_{k-1}^T)(I - w_{k-2}v_{k-2}^T)\cdots(I - w_0 v_0^T)P_0, \tag{4.43}$$

where $v_k = \dfrac{s_k}{\|s_k\|}$, $w_k = \dfrac{P_k^{-1}y_k - s_k}{\|s_k\| + v_k^T(P_k^{-1}y_k - s_k)}$, and $(I - w_i v_i^T)$'s are rank-1 updates to the preconditioner. Assume we truncate whenever the number of rank-1 updates hits m. Then the first time we truncate is at m-th nonlinear step. Similar to (4.5), the preconditioner update is

$$T_m = I + X_m, \tag{4.44}$$

where $X_m \in \mathbb{R}^{N\times N}$ is a rank-m matrix. We apply the truncation by SVD on X_m. After computing singular value decomposition $X_m = \sum_{i=1}^m \sigma_i u_i v_i^T$, we keep p dominant singular values and X_m is truncated into $\widetilde{X}_m$.

$$\widetilde{X}_m = \sum_{i=1}^{p} \sigma_i u_i v_i^T. \tag{4.45}$$

The newly truncated preconditioner is $\widetilde{P}_m = \widetilde{T}_m P_0 = (I + \widetilde{X}_m)P_0$, and the m-th Newton step after truncation is

$$J_m \widetilde{P}_m \widetilde{s}_m = -F(x_m).$$

Analysis

First, we introduce a bound from Bergamaschi et al. [11]. This bound is on the preconditioner with Broyden-type rank-one updates and is given in the following theorem.

Theorem 9. *Suppose $J(x)$ is Lipschitz continuous and $J(x^*)$ is nonsingular. Let $\alpha = 1/\|J(x^*)^{-1}\|$. Fixed constants $0 < \delta < \alpha$ and $0 < \delta_1$. Then there exist ε, such that $\|P_0 J_0\| < \varepsilon$*

and

$$\|I - J_k P_k\| < \frac{\delta_1 \alpha}{1 - \delta\alpha}. \tag{4.46}$$

Theorem 9 shows that the spectrum of $J_k P_k$ is included in a disk that is around $(1, 0)$ in the complex plane. The radius is $\frac{\delta_1 \alpha}{1 - \delta\alpha}$. $\alpha = 1/\|J(x^*)^{-1}\|$ and δ depends on the choice of initial preconditioner. By (4.46), the field of values of $J_k P_k$ is also included in a disk around $(1, 0)$.

With (4.45), we have $X_m - \tilde{X}_m = \sum_{i=p+1}^{m} \sigma_i u_i v_i^T$. Let $E := \sum_{i=p+1}^{m} \sigma_i u_i v_i^T$. Then $E = X_m - \tilde{X}_m$ and $\text{rank}\,(E) = m - p$. Since p dominant singular values are kept in $\tilde{X}_{m,p}$, we expect the low-rank matrix E is a small perturbation to X_m.

With $E = X_m - \tilde{X}_m$ and $\widetilde{P}_m$ given in Section 4.2, we have

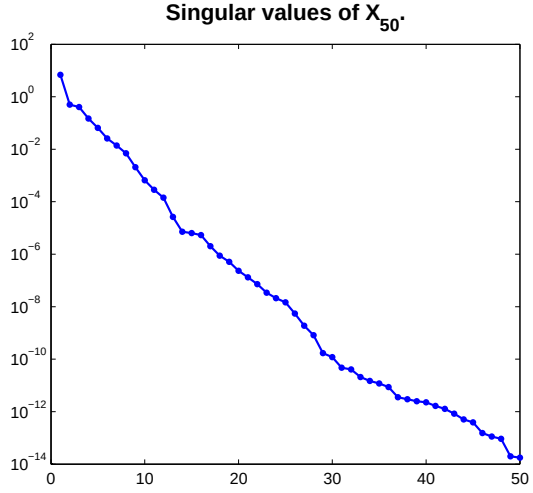
$$\begin{aligned} J_m \widetilde{P}_m &= J_m(I + \widetilde{X}_m)P_0 \\ &= J_m(I + X_m)P_0 - J_m E P_0 \\ &= J_m P_m - J_m E P_0. \end{aligned}$$

By theorem 9, the field of values of $J_m P_m$ is included in a disk that excludes the origin and the radius is $\frac{\delta_1 \alpha}{1 - \delta\alpha}$. It follows from theorem 3 that if $\|J_m E P_0\|_2 < 1 - \frac{\delta_1 \alpha}{1 - \delta\alpha}$, the field of values of $J_m \widetilde{X}_{m,p} P_0$ is included in a disk that excludes the origin and thus GMRES converges.

Numerical Results

In our experiments, $\eta = 0.1$, $\gamma = 1$, and $r = s = 1$. The Newton-Line Search method takes about 210 steps to converge. Figure 4.2 shows the singular values of X_{50}. The singular values decay very fast and SVD-based truncation turns out to work very well. Figure 4.3 demonstrates the norm of the difference between X_m and $\tilde{X}_m$, namely, $\|E\|$. Therefore, if we truncate 50 rank-1 updates into 20, $\|E\|$ is always less than $1e-4$. We could further obtain $\|A_m E P_0\| \leq 4e-3$ by simple computation. Therefore by theorem 2, the bound on

Figure 4.2: Singular values of the first 50 preconditioner updates product.



the residual is

$$\|r_k\|/\|r_0\| \leq 2\left(\delta + (4e-3)\right)^k, \tag{4.47}$$

where δ is determined by the initial ILU(0) preconditioner. So, GMRES is expected to converge very fast, which is proved in Figure 4.4.

In comparison, we test several ways to handle the preconditioner. First, people compute an initial ILU preconditioner with zero fill-in [72] and use this preconditioner for the remaining Newton steps. Secondly, people compute a new ILU preconditioner for the recomputed Jacobian at every Newton step. Thirdly, based on the initial ILU preconditioner, people update the preconditioner with continuous rank-1 updates for the remaining Newton steps. Last, when the number (size) of the rank-1 updates hits a threshold, we apply SVD-based truncation. As Newton method progresses, whenever the number of rank-1 updates hits the threshold, we truncate the preconditioner update accordingly.

Figure 4.4 demonstrates the comparison between various preconditioning. The SVD-based truncation maintains good number of iterations of GMRES. While cheaper to apply the preconditioner, the truncation approach arrives at similar and competitive convergence.

We also test different threshold to truncate the preconditioner update. It turns out that he

Figure 4.3: The norm of difference between X and $\widetilde{X}_m$ every time we truncate. $E = X_m - \widetilde{X}_m$. In this experiment, we use SVD-based truncation to discard 30 singular values out of 50.

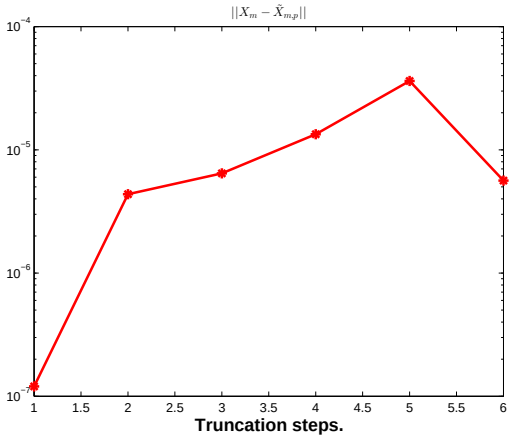


Figure 4.4: Comparison of different preconditioners. $ILUJ_0$ means we compute an initial ILU(0) preconditioner and use it for all the remaining Newton steps. $ILUJ_k$ means we compute a new ILU(0) preconditioner for each new Jacobian every Newton step. In 'Cumulative Rank-1', the preconditioner is updated with rank-1 update every Newton step. In SVD-T, the preconditioner update is truncated into 20 vectors whenever the number of rank-1 updates hits 50 and the SVD-based truncation approach is used.

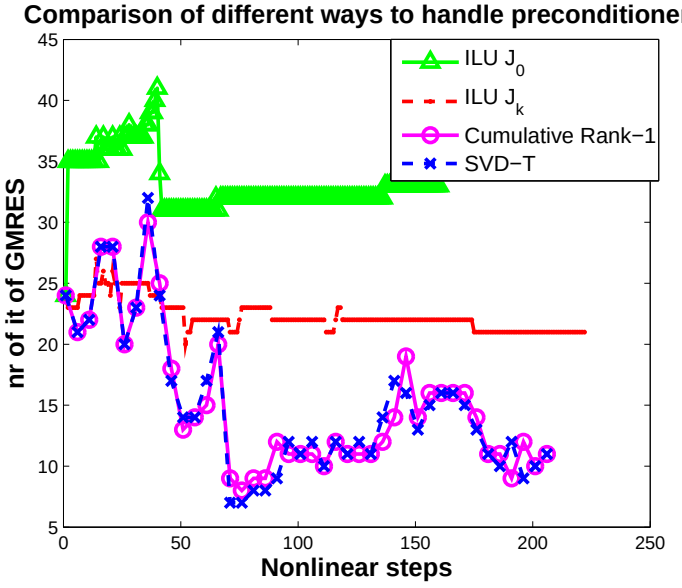
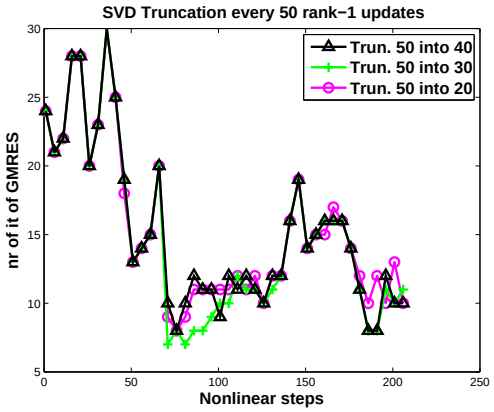


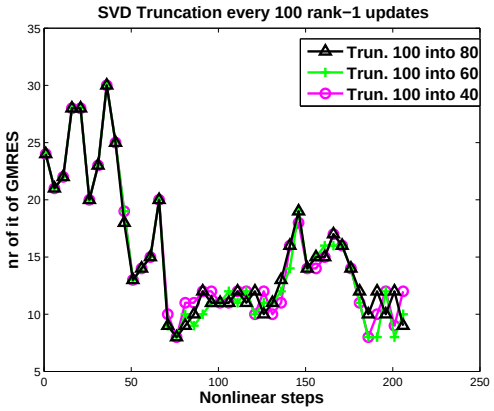
Figure 4.5: Comparison of different truncation size when the threshold of number of rank-1 updates is 50. 'Trun. 50 into 40' means we keep 40 rank-1 updates out of 50 when we truncate the preconditioner update.



number of iterations of GMRES is not very sensitive to the number of vectors we discard during truncation. This is mainly because most of the singular values are very small (see Figure 4.2). The results are in Figure 4.5 and 4.6.

Figure 4.5 shows the result when the threshold of truncation is 50, and Figure 4.6 is when the threshold is 100. In both cases, the truncation approach works well and the convergence is not very sensitive to the number of vectors we discard. As shown in the figures, the preconditioner update can be truncated into 40% of its own size with still good approximation (from 50 to 20, or 100 to 40). In further tests, the preconditioner update can be truncated into 20% of its own size with almost the same approximation.

Figure 4.6: Comparison of different truncation size when the threshold of number of rank-1 updates is 100. 'Trun. 100 into 80' means we keep 80 rank-1 updates out of 100 when we truncate the preconditioner update.



4.3 Application to the Quantum Monte Carlo Method

At every step, we need to evaluate

$$\left|\frac{\det A_{k+1}}{\det A_k}\right| = \left|1 + u_k^T A_k^{-1} e_{i_k}\right|.$$

We compute $A_k^{-1} e_{i_k}$ by solving $A_k z_k = e_{i_k}$ iteratively.

$$A_{k+1} = (I + z_k u_k^T) A_k,$$

where $z_k u_k^T$ is the rank-one update. The preconditioner is

$$P_k = (I + X_k)\, P_0,$$

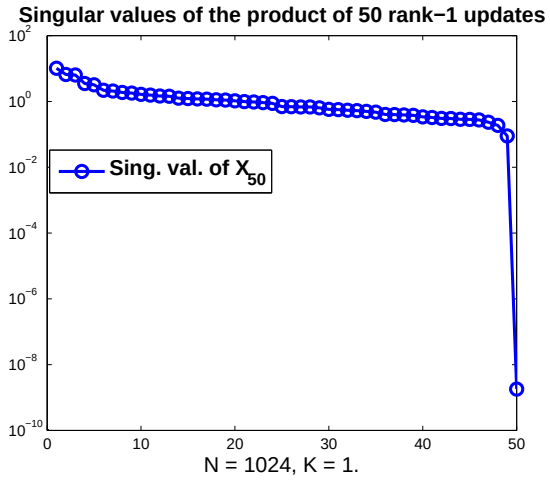
where X_k is given as before. We apply our truncation approaches onto X_i.

Numerical Results

Figure 4.7 shows the singular values of X_{50} in a specific linear system with $N = 1024$ and $K = 1$. The singular values of preconditioner update does not decay fast so we have very few singular values to discard. If we discard 30 vectors out of 50 vectors, we have $\|E\| \leq 0.32$ and $\|A_m E P_0\| \leq 4$, which ends up with a very large bound for the residuals by theorem 2. Therefore, the truncation by SVD may probably not produce a good approximation and it turns out that SVD-based truncation does not work well.

As we mentioned before, this is why we propose the truncation by canonical angles. Since we know in advance all the moving particles in these experiments, we happen to know the right-hand side vector of any future linear system. Therefore, when we apply the truncation by canonical angles, instead of using previous linear systems, we integrate the future ℓ right-hand side vectors together and then use it to obtain the space V_ℓ. Figure 4.8 shows the comparison of singular values for two truncation approaches in a same Monte Carlo step. As we can see, the truncation by canonical angles produces smaller singular values. Although

Figure 4.7: Singular values of X_{50}, the product of first 50 rank-1 updates for a 3D system with N = 1024, K = 1.



for example if we discard 30 vectors out of 50 vectors, we have $\|E\| \leq 8.72e-1$, which is still not a very small bound. Therefore, we expect the convergence is improved by canonical angles based truncation, but still not very fast. This is justified in our experiments. See Figure 4.9 and 4.10 for more details.

We test our two truncation approaches for several different systems. The preconditioner is ILU(0) or ILUTP [9, 10, 72]. Figure 4.9 shows the comparison between two truncation approaches. Truncation by canonical angles turns out to be better than that by SVD.

In a short run, both truncation approaches work well. For example, in Figure 4.9, the number of GMRES iterations changes from 4 to 17 during 250 Monte Carlo steps after we compute a new ILUTP preconditioner. In a long run, we need to recompute the ILUTP preconditioner when the number of iterations of GMRES hits some threshold. In both Figure 4.9 and 4.10, the truncation by canonical angles is better than the SVD-based truncation.

Figure 4.8: Singular values of X_{50}, the product of first 50 rank-1 updates for some system in QMC problem. The system size is 1024 and the Gaussian parameter $K = 1$.

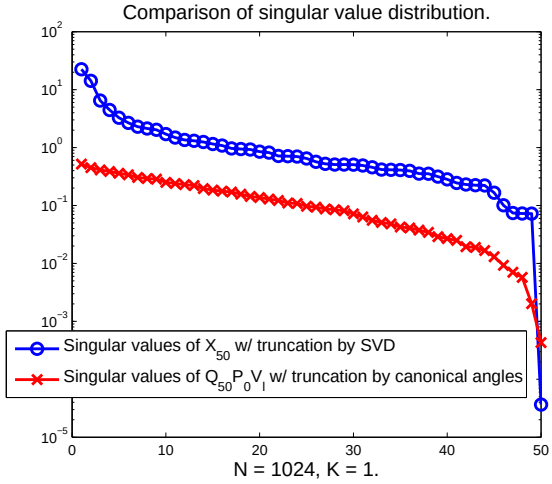


Figure 4.9: Comparison of number of iterations of GMRES for different truncation approaches. In both truncation, we discard 30 rank-1 updates out of 50. This is the result for $N = 1024$ system with $K = 1$.

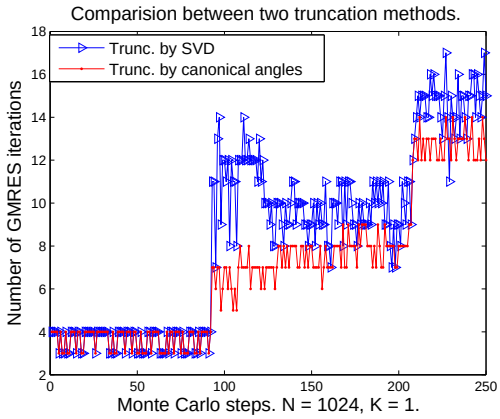
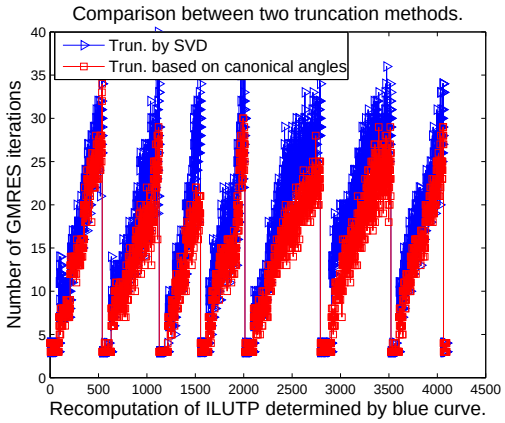


Figure 4.10: Comparison of number of iterations of GMRES in 4000 steps. This is for $N = 1024$ system with $K = 1$. Notice that we recompute ILUTP with respect to the blue curve. The mean of blue curve (SVD-based truncation) is 16.72 while the mean of red curve (truncation by canonical angles) is 13.57.



Chapter 5

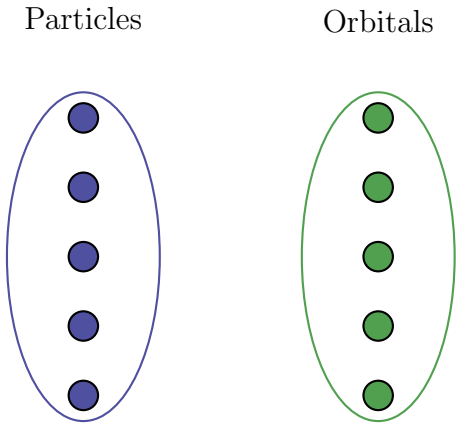
Slater Matrix Reordering

Some preconditioners require diagonal dominance of the system matrix. For example, the ILU preconditioner works well when the matrix is diagonally dominant or close to diagonal dominant. Further, if the matrix is far from diagonal dominant, the ILU algorithm is unstable [80, 72, 59, 18]. In this chapter, we develop and test our reordering schemes on Slater matrix. Because the Slater matrix keeps changing during the MCMC process, it is very important to prevent the diagonal dominance from deteriorating. We reorder the Slater matrix regularly by permuting rows or columns to improve its diagonal dominance.

In Section 5.1, we introduce a few popular methods and algorithms that improves the diagonal dominance. However, the disadvantage of these algorithms is that they cannot prevent zeros from the diagonal. We build an reordering algorithm with diagonal cutoff to avoid zeros in Section 5.2. In Section 5.3, we explain and test two reordering schemes, global and local reordering.

We demonstrate the effectiveness of our reordering algorithm to improve the diagonal dominance. Further, the cumulative local reordering turns out to be as competitive as global reordering. In this chapter, all algorithms are tested on Slater matrix but they could also be applied to improve the diagonal dominance of other matrices.

Figure 5.1: Demonstration of a bipartite graph of particles and orbitals



5.1 Slater Matrix and Bipartite Graph

The Slater matrix represents the relationship between particles and orbitals in QMC method. Each element of the Slater matrix relates to a particle and an orbital. If we see from the graph way, any element is a weighted edge that connects the particle and orbital. Because the system has same number of particles and orbitals, the counterpart of Slater matrix in graph theory is a bipartite graph [12, 85]. Slater matrix is the biadjacency matrix of the bipartite graph. Note that the graph is undirected because there is no direction or order in the physical system. Figure 2.1 shows the particles and orbitals in a 2D system. Figure 5.1 shows the bipartite graph that consists of two subsets, particles and orbitals.

Different Reordering

Our goal is to improve the diagonal dominance of Slater matrix by permuting rows and columns. There are two straightforward and general methods, one is to maximize the diagonal and the other is to maximize the trace. Both methods are optimization problems, that is, we want to maximize either the diagonal or the trace over all possible row/column permutations. For the bipartite graph, this is a combinatorial optimization problem [31, 42, 43, 51].

If we want to maximize the diagonal of the Slater matrix, it means to find the maximum matching for the bipartite graph [32, 55, 21]. Notice that the maximum matching may or may not be unique. Also, maximizing the trace is equivalent to find the maximum weighted matching [19, 47]. Therefore, we can make use of matching algorithms, obtain the matching information, and then reorder the Slater matrix.

We explain both methods as follows.

1. **Maximizing the diagonal**. This method aims to make the Slater matrix close to diagonal dominant as much as possible. Take one particular row for example. This method maximizes the diagonal element column permutations.

   This method is easy to implement with *Greedy algorithm* [67]. In practice, we start from one particular row, maximize the diagonal element, and then fix this row and its corresponding column. We keep maximizing the diagonal elements for the remaining rows and columns till a single diagonal element is left. The more rows and columns we have fixed, the less choices are left for the remaining steps. Sometimes the choices are so few that no permutation will prevent zero on the diagonal. *Greedy algorithm* is cheap to implement but the diagonal dominance may possibly deteriorate for the later rows.

   Because maximizing the diagonal means to find maximum matching for the bipartite graph, we can utilize the max-flow method. There are a few algorithms available. In our experiments, we use the push-relabel algorithm [19, 47].

2. **Maximizing the trace**. Another idea is to maximize the trace, that is, the sum of the absolute values of all diagonal entries. Because any element of the Slater matrix is positive, we can skip absolute values. As mentioned before, we convert the problem of maximizing the trace to finding the maximum weighted matching. There are quite a few options available, such as linear programming, the Hungarian algorithm, the Ford-Fulkerson algorithm, and the Edmonds-Karp algorithm. In practice, we choose the Hungarian algorithm [31, 42, 43, 51].

Algorithm 3 Greedy Algorithm to Maximize the Diagonal

1: **for** $i = 1$ to $n - 1$ **do**
2: Find the largest element among i-th to n-th elements in i-th row and column.
3: **if** the diagonal element is less than the largest element **then**
4: Swap the largest element and diagonal element by permuting corresponding rows or columns.
5: **end if**
6: **end for**

5.2 Our Reordering with Diagonal Cutoff

The diagonal and trace maximization both improves the diagonal dominance to some extent. However, it is possible that we get zero(s) on the diagonal for both methods. In our experiments on Slater matrices, we do often get zeros on the diagonal for both methods. Zero(s) in the diagonal will make ILU algorithm unstable or break down. Therefore, we need to fix the issue by making the minimum of diagonal entries nonzero. This can be done by maximizing the minimum of the diagonal. For Slater matrix, this is always applicable because physically no particle is isolated (so we can always find a matching to maximize the smallest diagonal element). In graph language, we find the optimal matching such that the longest path is minimized.

Maximizing the Minimum Absolute Value of Diagonal Entries

We use bisection method and the push-relabel algorithm to maximize the minimum of the diagonal. The optimal value is unique. However in practice, instead of finding the exact value that maximizes the smallest diagonal element, we always look for a larger cutoff value that allows much more matchings. This gives more freedom to operate other optimization algorithm, such as maximum weighted matching algorithm. Following are main steps of the algorithm.

1. Initialization. Convert A into a bipartite graph G (we want to convert the maximum matching problem to a max-flow problem). Construct a directed graph G' from G. We obtain G' by adding a source vertex s and a sink vertex v. Further, for any edge in G, we change it into a directed edge in G'. We store the biadjacency matrix of G' as a sparse matrix. The form of G' is

$$G' = \begin{pmatrix} 0 & 1\cdots 1 & 0\cdots 0 & 0 \\ \vdots & O & A & 0 \\ & 0\cdots 0 & 0\cdots 0 & 1 \\ \vdots & \cdots & \cdots & \vdots \\ & \cdots & \cdots & 1 \\ 0 & \cdots & \cdots & 0 \end{pmatrix}.$$

2. Suppose the maximal element in A is b, $b = \sup\{a_{ij} : a_{ij} \in A, 1 \leq i, j \leq N\}$. Notice that b is positive since all elements of A are positive. Implement bisection method on the interval $[a, b]$, where $a = 0$. Terminate with cutoff ε.

   - $c = \dfrac{a+b}{2}$. Change all elements of A smaller than c into zeros and update G' accordingly.
   - Use the push-relabel algorithm to see if there exists a perfect matching in G'.
     - If yes, $a = c$.
     - If no, $b = c$.

This algorithm reminds a method to optimize the diagonal. Every time we obtain an ε, we extract the max-flow matching information, reword it into row permutations, and then transform A accordingly. Next we fix rows and columns whose diagonal elements are equal to or very close to ε. We remove these rows and columns from A to obtain a smaller size matrix A'. We then apply the algorithm of maximizing the minimum of the diagonal to the new A'. We get a new ε' for A', extract matching information, and reword the information to fix more rows and columns in the original Slater matrix A. We keep doing this until enough

Algorithm 4 Maximizing the minimum of the diagonal

1: Initialization. Convert the Slater matrix A to a bipartite graph G. Construct a directed graph G' by adding a source vertex s and a sink vertex v to G.
2: Initial cutoff $\varepsilon = \dfrac{\text{start} + \text{terminal}}{2}$.
3: **for** $i = 1$ to max **do**
4: Use the push-relabel algorithm to find the maximum matching
5: **if** (the maximum matching exists) **then**
6: Start $= \varepsilon$;
7: **else**(the maximum matching does not exist)
8: Terminal $= \varepsilon$;
9: **end if**
10: **if** ($\varepsilon < tol_1$) or (|start - terminal| $< tol_2$) **then**
11: Break;
12: **end if**
13: **end for**

(even all) rows and columns are fixed. However, this method is too expensive. We need to implement $O(N)$ times bisection method and $O(N)$ times push-relabel algorithm.

Our Reordering with a Cutoff on the Diagonal

To reduce the cost, we propose a reordering which plays at middle ground. We mix both the trace maximization and the minimum of the diagonal maximization. We maximize the smallest element in the diagonal first, and then maximize the trace. This procedure prevents zeros on the diagonal, and only employs one time bisection method and push-relabel algorithm. The diagonal cutoff value ε we obtain from the bisection method prevents zero from the diagonal. We call this reordering with a cutoff on the diagonal.

1. Maximize the minimum of the diagonal elements. Use both bisection method and the

push-relabel algorithm to find a cutoff ε.

2. Extract the max-flow information and reword it into row permutations for A. Permute A.

3. Penalty step. Change elements of A that are smaller than ε into some negative value. Construct graph G.

4. Use the Hungarian Algorithm to find the maximum weighted matching for G. Reword the matching information and permute A accordingly.

Remark:

1. Notice that every time we swap two rows, we need to reindex (update) the particle that is not in the diagonal. For example, if (1, 3), (2, 4), (3, 5) are three matching pairs we obtain. Suppose we swap row 1 and row 3 first, which means we assign particle 1 to orbital 3. After we have swapped the two rows, we need change the pair (3, 5) into (1, 5). This is because particle 1 and 3 are swapped in the first row permutation.

2. Since the matching is in terms of rows (particles), only row permutations will be performed. Because the set of all row permutations is equivalent to the set of all column permutations, it does not make any difference if we allow column permutations.

3. For more information about max-flow method and the Hungarian algorithm, see [13, 28, 56, 87]. For detailed code package, see also Melin [62] and David Gleich [7, 37].

Numerical Results

In this section, we test different reordering algorithms on Slater matrices. The first two algorithms are to maximize the diagonal and the trace. The third is our reordering with a cutoff on the diagonal. To maximize the diagonal, we use *Greedy Algorithm*. To maximize the trace, we use the Hungarian algorithm.

It is shown that our reordering with a cutoff on the diagonal works best to improve both the diagonal dominance and the spectrum of Slater matrix. Figure 5.2 shows that our reordering with a diagonal cutoff improves the diagonal dominance significantly for most rows of the Slater matrix. In Figure 5.3, it is shown that our reordering with a diagonal cutoff produces much less ratios that are close to zero than the trace maximization reordering. This means our reordering with a diagonal cutoff produces better diagonal dominance. Notice that we order the ratios monotonic to make them more observable in Figure 5.2 and 5.2.

Figure 5.2: Comparison of diagonal dominance before and after our reordering with a diagonal cutoff for a 3D system with N =1024, K =1. The cutoff is 0.02. We compute the ratio between the diagonal entry and the absolute row sum minus diagonal entry for each row.

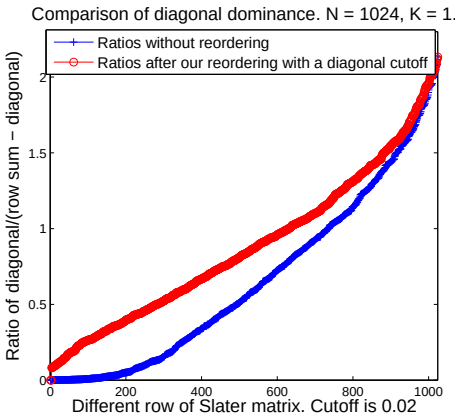
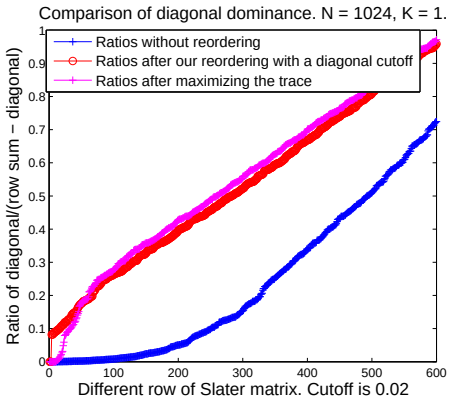


Figure 5.3: Comparison of diagonal dominance with different reordering for a 3D system with N =1024, K =1. The cutoff is 0.02. We compute the ratio between the diagonal entry and the absolute row sum minus diagonal entry for each row.

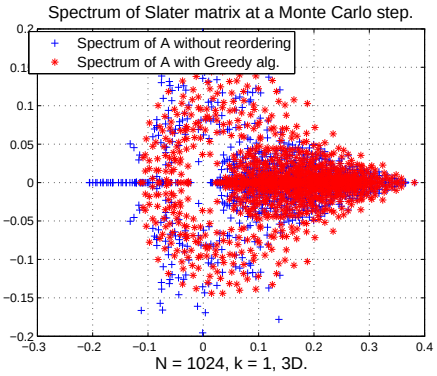


We also look into the effect of reordering on the spectrum. In Figure 5.4, we compare between the *Greedy Algorithm* and our algorithm. The spectrum of a Slater matrix at a specific Monte Carlo steps is shown. Figure 5.5 shows the result for a Slater matrix from different Monte Carlo step. All results demonstrate that our reordering works better than the *Greedy Algorithm*.

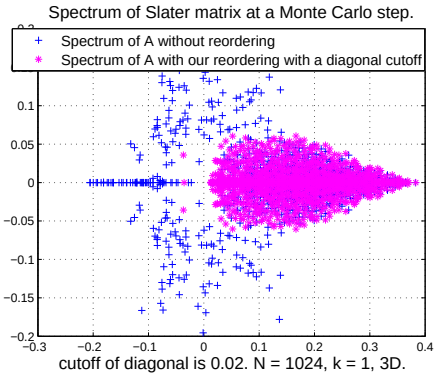
In Figure 5.6, we compare the trace maximization algorithm and our reordering algorithm with a diagonal cutoff. Because there is no minimum requirement on the diagonal, the trace maximization ends up with many eigenvalues around the origin. Therefore, our reordering with a diagonal cutoff is better with less eigenvalues around the origin.

We also test different cutoffs on the diagonal, such as 0.02, and 0.5 in Figure 5.5(b) and 5.7(b). The spectrum are similar and not sensitive to the cutoff. Therefore in practice, we are flexible on determining the cutoff of the diagonal.

Figure 5.4: Spectrum of a Slater matrix with different reordering algorithm (1).

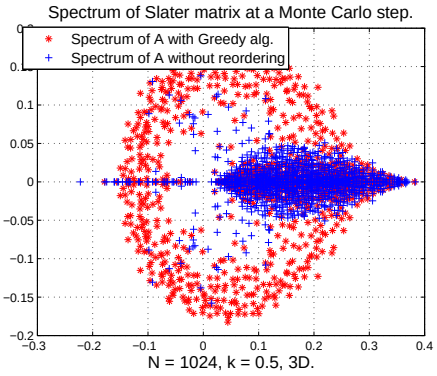


(a) Spectrum of a Slater matrix with greedy algorithm to maximize the diagonal. The system is in 3D and N = 1024, K = 1.

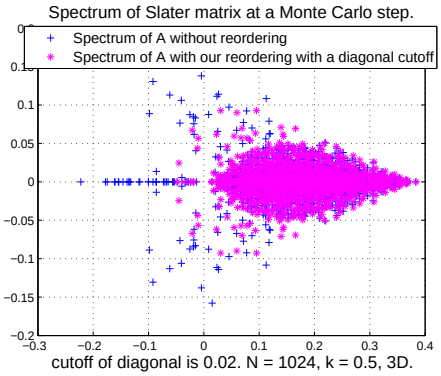


(b) Spectrum of a Slater matrix with our reordering algorithm with a diagonal cutoff 0.02. The system is in 3D and N = 1024, K = 1.

Figure 5.5: Spectrum of a Slater matrix with different reordering algorithm (2).

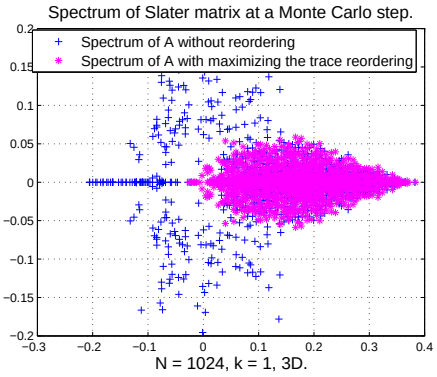


(a) Spectrum of a Slater matrix with greedy algorithm to maximize the diagonal. The system is in 3D and N = 1024, K = 0.5.

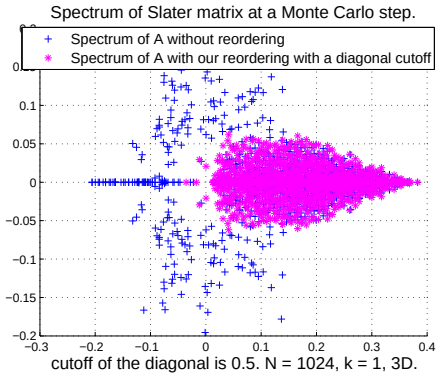


(b) Spectrum of a Slater matrix with our reordering algorithm with a diagonal cutoff 0.02. The system is in 3D and N = 1024, K = 0.5.

Figure 5.6: Spectrum of a Slater matrix with different reordering algorithm (3).



(a) Spectrum of a Slater matrix with the algorithm to maximize the trace. The system is in 3D and N = 1024, K = 1.



(b) Spectrum of a Slater matrix with our reordering algorithm with a diagonal cutoff 0.5. The system is in 3D and N = 1024, K = 1.

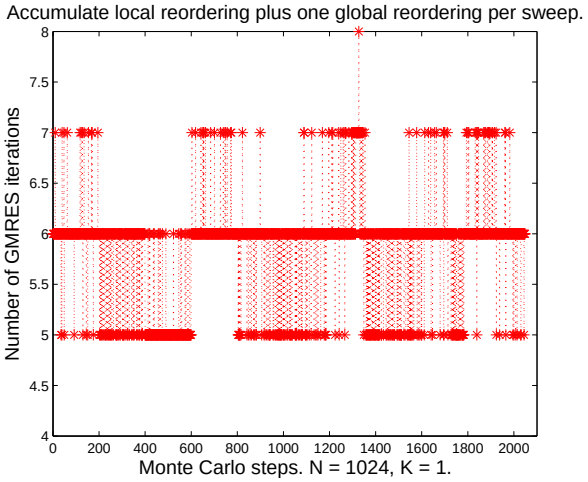
5.3 Global and Local Reordering

If we apply our reordering onto the entire Slater matrix, we call it global reordering. The cost of max-flow method and the Hungarian algorithm is $O(N^3)$, which is very expensive. When we implement global reordering, we observe that only a few rows need to be permuted. This suggests the possibility to reorder only a small part of the Slater matrix. As described in Section 3.3 of Chapter 3, the Slater matrix A can be divided into $\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}$ by domain decomposition methods. Therefore, we can apply our reordering with a diagonal cutoff onto the submatrix A_{11} based on the factorization of A. We call this local reordering. If we are using domain decomposition preconditioner or inexact preconditioning with inner-outer GMRES, the factorization is immediately available. If we are using other preconditioners, we need to first obtain A_{11} by domain decomposition method, and then apply our reordering. Because A_{11} has much smaller dimension, the local reordering reduces the cost significantly, that is, from $O(N^3)$ to $O(1)$ per reordering.

Since a particle always moves locally, the Slater matrix changes locally and thus we expect the local reordering is effective. Whenever a particle move is accepted, we perform local

reordering and store the matching information for the Slater matrix. When we need to recompute a preconditioner or perform global reordering, we extract all accumulate local reordering information and make the permutations. We call this accumulate local reordering. In Figure 5.7, the number of iterations stays low as we recompute a new ILUTP every 200 Monte Carlo steps. This demonstrates the effectiveness of accumulate local reordering.

Figure 5.7: Number of iterations with accumulate local reordering for a 3D system with N =1024 and K =1. We also perform global reordering once per sweep. The preconditioner is ILUTP, updated by cumulative rank-one updates, and computed every 200 Monte Carlo steps.



Algorithm 5 Local Reordering Algorithm

1: Maximize the minimum of the diagonal of A_{11}, a small part of the Slater matrix corresponds to the neighborhood of the moving particle. The solution give us the cutoff ε of the minimum on the diagonal.

2: Penalty: change elements of A_{11} that are smaller than ϵ into some negative value. Construct graph G.

3: Use the Hungarian Algorithm to find the maximum weighted matching for G. Permute A_{11} accordingly.

4: Recover the reordering information for A and store it.

Comparison of the Two Reordering Schemes

In our experiments, we always perform an initial global reordering because it produces a good initial matrix. Then we apply local reordering whenever there is a rank-one update to the Slater matrix. To reduce the cost and also for simplicity, we do not reorder the resulting Slater matrix instantly. Instead, we store the local reordering information till we do next global reordering. So, we do cumulative local reordering and reorder the matrix when necessary.

In general, when the smallest diagonal element becomes larger than the threshold, we perform a global reordering. In practice, the global reordering is performed a couple of times every sweep for all systems with K = 1. For systems with K = 0.5, it may take more times global reordering every sweep. For example, four times global reordering are necessary for a 3D system with N = 2000, and K = 0.5.

Table 5.3, Figure 5.8 and 5.9 show that the local reordering contains most information of the global reordering for every rank-one update in the Slater matrix. This demonstrates why we can perform cheaper local reordering with almost the same effect on the diagonal dominance.

Figure 5.8: Number of row permutations in global and local reordering for a 3D system with N =1024 and K =1.

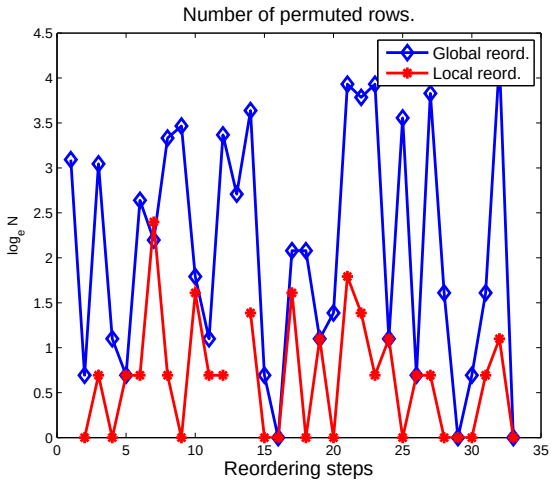
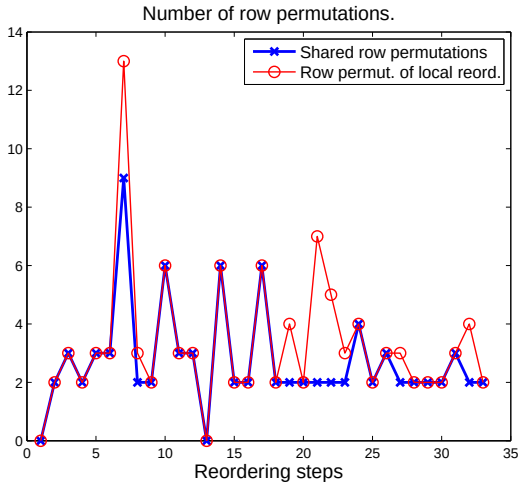


Table 5.1: Comparison between global and local reordering. The overlap means the number of shared permutations between global and local reordering at each reordering. The size of local or global reordering is the dimension of the squared matrix that we reorder.

	Number of permutations in each reordering						
Global reordering	4	24	5	3	20	10	14
Local reordering	2	3	2	3	3	11	2
Overlap percent.	2	3	2	3	3	9	2
	100%	100%	100%	100%	100%	81.8%	100%
Size of local reordering	58	60	57	62	58	70	58
Size of global reordering	1024	1024	1024	1024	1024	1024	1024

Figure 5.9: Number of row permutations for the 3D system with N =1024 and K =1. The red curve is the number of row permutations in the local reordering and the blue curve is the number of overlapped row permutations between global and local reordering.



Bibliography

[1] K. Ahuja. *Recycling Krylov subspaces and preconditioners*. PhD thesis, Department of Mathematics, Virginia Polytechnic Institute and State University, 2011. Advised by E. de Sturler.

[2] K. Ahuja, B. K. Clark, E. de Sturler, D. M. Ceperley, and J. Kim. Improved scaling for quantum Monte Carlo on insulators. *SIAM J. Sci. Comput.*, 33(4):1837–1859, 2011.

[3] M. A. Akgun, J. H. Garcelon, and R. T. Haftka. Fast exact linear and non-linear structural reanalysis and the Sherman-Morrison-Woodbury formulas. *International Journal for Numerical Methods in Engineering*, 50(7):1587–1606, 2001.

[4] D. Alfé and M. J. Gillan. An efficient localized basis set for quantum Monte Carlo calculations on condensed matter. *Phys. Rev. B*, 70:161101, 2004.

[5] D. Alfé and M. J. Gillan. Linear-scaling quantum Monte Carlo with non-orthogonal localized orbitals. *J. Phys.: Condensed Matter*, 16:L305–L311, 2004.

[6] Z. Bai, W. Chen, R. Scalettar, and I. Yamazaki. *Numerical Methods for Quantum Monte Carlo Simulations of the Hubbard Model, in MultiScale Phenomena in Complex Fluids*. Higher Education Press, China, 2009.

[7] M. Bayati, M. Gerritsen, D. F. Gleich, A. Saberi, and Y. Wang. Algorithms for large, sparse network alignment problems. *Ninth IEEE International Conference on Data Mining*, pages 705–710, 2009.

[8] I. Beichl and F. Sullivan. The Metropolis algorithm. *Computing in Science & Engineering*, 2000.

[9] M. Benzi. *Preconditioning techniques for large linear systems: A survey.* J. Comput. Phys. 182, 2002.

[10] M. Benzi, D. B. Szyld, and A. van Duin. Orderings for incomplete factorization preconditioning of nonsymmetric problems. *SIAM J. Sci. Comput.*, 20:1652–1670, 1999.

[11] L. Bergamaschi, R. Bru, A. Martinez, and M. Putti. Quasi-Newton preconditioners for the inexact Newton method. *Electron. Trans. Numer. Anal.*, 23:76–87, 2006.

[12] J. A. Bondy and U. S. R. Murty. *Graph Theory with Applications.* North-Holland, 1976.

[13] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision. *Pattern Analysis and Machine Intelligence*, 26:1124 – 1137, 2004.

[14] E. W. Brown, B. K. Clark, J. L. Dubois, and D. M. Ceperley. Path-integral Monte Carlo simulation of the warm-dense homogeneous electron gas. *Phys. Rev. Letts.*, 110(146405), 2013.

[15] J. F. Cai, E. J. Candes, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM J. Optim.*, 20:1956–1982, 2010.

[16] D. M. Ceperley and B. Alder. *Quantum Monte Carlo*, volume 231. American Association for the Advancement of Science, 1986.

[17] D. M. Ceperley, G. V. Chester, and M. H. Kalos. Monte Carlo simulation of a many fermion system. *Phys. Rev. B*, pages 3081–3099, 1977.

[18] T. Y. Chen. *Preconditioning Sparse Matrices for Computing Eigenvalues and Solving Linear Systems of Equations.* PhD thesis, MIT, 2001.

[19] B. V. Cherkassky and A. V. Goldberg. On implementing the push-relabel method for the maximum flow problem. *Algorithmica*, 19(4):390–410, 1997.

[20] E. Chow and Y. Saad. Experimental study of ILU preconditioners for indefinite matrices. *J. Comput. Appl. Math*, 86(2):387–414, 1997.

[21] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms, 3rd Edition*. MIT Press, 2009.

[22] E. de Sturler. Truncation strategies for optimal Krylov subspace methods. *SIAM J. Numer. Anal.*, 36(3):864–889, 1999.

[23] R. S. Dembo, S. C. Eisenstat, and T. Steihaug. Inexact Newton methods. *SIAM J. on Numerical Analysis*, 19(2):400–408, 1982.

[24] I. S. Duff and J. Koster. The design and use of algorithms for permuting large entries to the diagonal of sparse matrices. *SIAM J. Matrix Anal. Appl.*, 20:889–901, 1999.

[25] I. S. Duff and J. Koster. On algorithms for permuting large entries to the diagonal of a sparse matrix. *SIAM J. Matrix Anal. Appl.*, 22:973–996, 2001.

[26] T. Eirola and O. Nevanlinna. Accelerating with rank-one updates. *Linear Algebra and its Applications*, 121:511–520, 1989.

[27] E. Fermi and R. Richtmyer. Note on census-taking in Monte Carlo calculations. *Declassified report Los Alamos Archive*, 805, 1948.

[28] G. W. Flake, S. Lawrence, C. L. Giles, and F.M. Coetzee. Self-organization and identification of web communities. *Computer*, 35(4):66–70, 2002.

[29] D. R. Fokkema, G. L. G. Slejipen, and H. A. van der Vorst. Accelerated inexact Newton schemes for large systems of nonlinear equations. *SIAM J. Sci. Comput.*, 19:657–674, 1997.

[30] W. M. C. Foulkes, L. Mitas, R. J. Needs, and G. Rajagopal. Quantum Monte Carlo simulations of solids. *Rev. Mod. Phys.*, 506:73–89, 2001.

[31] A. Frank. On Kuhn's Hungarian method-A tribute from Hungary. *Naval Research Logistics*, 2005.

[32] A. M. Frieze. An algorithm for algebraic assignment problems. *Discrete Applied Mathematics*, 1979.

[33] D Fritzsche, A. Frommer, S. D. Shank, and D. B. Szyld. Overlapping blocks by growing a partition with applications to preconditioning. *SIAM J. Sci. Comput.*, 35(1):A453–A473, 2013.

[34] P. E. Gill and W. Murray. *Modification of Matrix Factorizations after a Rankone Change*. Number 55-83 in The State of the Art in Numerical Analysis,. Academic Press, New York, 1977.

[35] P. E. Gill, W. Murray, and M. A. Saunders. Methods for computing and modifying the LDV factors of a matrix. *Math. Comp.*, 29:1051–1077, 1975.

[36] L. Giraud, S. Gratton, and E. Martin. Incremental spectral preconditioners for sequences of linear systems. *Applied Numerical Mathematics*, 57:1164–1180, 2007.

[37] D. Gleich. MatlabBGL. *Mathworks file exchange*, 2006.

[38] N. Gmati and B. Philippe. Comments on the GMRES convergence for preconditioned systems. *Large-Scale Scientific Computing Lecture Notes in Computer Science*, 4818:40–51, 2008.

[39] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, MD, 2nd edition, 1989.

[40] R. M. Gray. Toeplitz and circulant matrices: A review. *Now Publishers Inc*, 2006.

[41] A. Greenbaum. *Iterative Methods for Solving Linear Systems.* SIAM, Philadelphia, 1997.

[42] D. M. Greig, B. T. Porteous, and A. H. Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society*, 51(2):271–279, 1989.

[43] M. Grotschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981.

[44] W. W. Hager. Updating the inverse of a matrix. *SIAM Review*, 31(2):221–239, 1989.

[45] B. Hammond, W. A. Lester, and P. J. Reynolds. Monte Carlo methods in ab initio quantum chemistry. *World Scientific, Singapore*, 1994.

[46] J. H. Hetherington. Observations on the statistical iteration of matrices. *Phys. Rev. A*, 30, 1984.

[47] J. E. Hopcroft and R. M. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM J. Comput.*, 2(4):225–231, 1973.

[48] R. A. Horn and C. R. Johnson. *Topics in Matrix Analysis.* Cambridge University Press, 1994.

[49] K. W. Gildersleeve J. P. Boyd. Numerical experiments on the condition number of the interpolation matrices for radial basis functions. *Applied Numerical Mathematics*, 61:443–459, 2011.

[50] C. R. Johnson. Further lower bound for the smallest singular value. *Linear Algebra and its Applications*, 272:169–179, 1998.

[51] R. Jonker and T. Volgenant. Improving the Hungarian assignment algorithm. *Operations Research Letters*, 5(4):171–175, 2003.

[52] C. T. Kelley. *Iterative Methods for Linear and Nonlinear Equations.* SIAM, Philadelphia, 1995.

[53] C. T. Kelley. *Solving Nonlinear Equations with Newtons Methods*. SIAM, Philadelphia, 2003.

[54] T. Kerkhoven and Y. Saad. On acceleration methods for coupled nonlinear elliptic systems. *Numer. Mathematics.*, 60:525–548, 1991.

[55] S. Koziel and X. S. Yang. *Computational Optimization, Methods and Algorithms*, volume 356 of *Studies in Computational Intelligence*. Springer, 2011.

[56] H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics quarterly*, 1955.

[57] R. B. Lehoucq, D. C. Sorensen, and C.Yang. ARPACK users' guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods. *SIAM, Philadelphia*, 1998.

[58] H. Lütkepohl. *Handbook of Matrices*. John Wiley&Sons, 1996.

[59] J. Mayer. Alternative weighted dropping strategies for ILUTP. *SIAM J. Sci. Comput.*, 27:1424–1437, 2006.

[60] W. McMillan. Ground state of liquid helium. *Physical Review*, 138(2A), 1965.

[61] K. Meerbergen and Z. J. Bai. The Lanczos method for parameterized symmetric linear systems with multiple right-hand sides. *SIAM. J. Matrix Anal. & Appl*, 31:1642–1662, 2010.

[62] A. Melin. Hungarian algorithm: An algorithm to find the minimum edge weight matching for an arbitrary bipartite graph. *Mathworks file exchange*, 2006.

[63] R. Myers. *Classical and Modern Regression with Applications*. Duxbury Press, 1986.

[64] O. Nevanlinna. *Convergence of Iterations for Linear Equations*. Springer, 1993.

[65] P. K. V. V. Nukala and P. R. C. Kent. A fast and efficient algorithm for Slater determinant updates in quantum Monte Carlo simulations. *J. Chem. Phys.*, 130(20):204105, 2009.

[66] D. Osei-Kuffuor and Y. Saad. Preconditioning Helmholtz linear systems. *Applied Numerical Mathematics*, 60:420–431, 2010.

[67] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization Algorithms and Complexity*. Dover publications, 1998.

[68] M. L. Parks, R. Sampath, and P. Nukala. Efficient simulation of large-scale 3D fracture networks via Krylov subspace recycling (in preparation).

[69] K. S. Riedel. A Sherman-Morrison-Woodbury identity for rank augmenting matrices with application to centering. *SIAM. J. Matrix Anal. & Appl.*, 13:659–662, 1992.

[70] M. Rosenbluth and A. Rosenbluth. Monte Carlo calculations of the average extension of macromolecular chains. *J. Chem. Phys.*, 23, 1955.

[71] A. N. Rubtsov, V. V. Savkin, and A. I. Lichtenstein. Continuous-time quantum Monte Carlo method for fermions. *Phys. Rev. B*, B 72(3):035122, 2005.

[72] Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, Philadelphia, 2nd edition, 2003.

[73] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual method for solving nonsymmetric linear systems. *SIAM J.on Scientific and Statistical Computing*, pages 856–869, 1986.

[74] H. A. Schwarz. Ueber einen Grenzübergang durch alternirendes Verfahren. 1870.

[75] J. R. Shewchuk. An introduction to the conjugate gradient method without the agonizing pain. *Carnegie Mellon University*, 1994.

[76] V. Simoncini and D. B. Szyld. Theory of inexact Krylov subspace methods and applications to scientific computing. *SIAM J. Sci. Comput.*, 25(2):454–477, 2003.

[77] B. Smith, P. Bjorstad, and W. Gropp. *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge University Press, 2004.

[78] J. D. Tebbens and M. Tuma. Efficient preconditioning of sequences of nonsymmetric linear systems. *SIAM J. Sci. Comput.*, 29:1918–1941, 2007.

[79] J. Thijssen. *Computational Physics*. Cambridge University Press, 1999.

[80] E. Toczylowski. A perfect matching algorithm for sparse bipartite graphs. *Discrete Applied Mathematics*, 9:263–268, 1984.

[81] A. Toselli and O. Widlund. *Domain Decomposition Methods-Algorithms and Theory*. Springer, 2005.

[82] H. A. van der Vorst. GMRESR: A family of nested GMRES methods. *Technische Universiteit Delft, Faculteit der Technische Wiskunde en Informatica*, 1991.

[83] H. A. van der Vorst. *Iterative Krylov Methods for Large Linear Systems, 1st Edition*. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 2009.

[84] J. M. Varah. A lower bound for the smallest singular value of a matrix. *Linear Algebra and its Applications*, 11:3–5, 1975.

[85] D. B. West. *Introduction to Graph Theory, 2nd Edition*. University of Illinois, 2001.

[86] A. J. Williamson, R. Q. Hood, and J. C. Grossman. Linear-scaling quantum Monte Carlo calculations. *Phys. Rev. Lett.*, 87:246–406, 2001.

[87] W. L. Winston and J. A. B. Goldberg. *Operations Research: Applications and Algorithms*. Philadelphia University, 2004.

[88] T. G. Wright. Eigtool. *http://www.comlab.ox.ac.uk/pseudospectra/eigtool/*, 2002.

[89] Y. S. Yu and D. H. Gu. A note on a lower bound for the smallest singular value. *Linear Algebra and its Applications*, 253:25–28, 1997.