Anderson Acceleration of Fixed-point Iteration with Applications to Electronic Structure Computations

by

Peng Ni

A Dissertation

Submitted to the Faculty

of

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Doctor of Philosophy

in

Mathematical Sciences

by

November 13, 2009

APPROVED:

Professor Homer F. Walker
Dissertation Advisor
Department of Mathematical Sciences
Worcester Polytechnic Institute

Professor Mayer Humi
Department of Mathematical Sciences
Worcester Polytechnic Institute

Professor L. Ramdas Ram-Mohan
Department of Physics
Worcester Polytechnic Institute

Professor Roger Lui
Department of Mathematical Sciences
Worcester Polytechnic Institute

Professor Suzanne L. Weekes
Department of Mathematical Sciences
Worcester Polytechnic Institute

Abstract

In electronic structure computations, it is necessary to set up and solve a certain nonlinear eigenvalue problem to identify materials. In this dissertation, we first introduce the nonlinear eigenvalue problem and the currently prevailing Self-Consistent Field (SCF) method accelerated by the Anderson acceleration method. We then compare the Anderson acceleration method with the well-known Generalized Minimal Residual (GMRES) method and show that they are essentially equivalent when applied to linear systems. After that, we study a linearly constrained least-squares problem embedded in the Anderson procedure. We use numerical experiments to illustrate the convergence properties. Finally, we give a summary of our work and an outline of future research.

Acknowledgments

It has been five years since I first came to pursue my PhD study in WPI, but when I look back, I can still see the young “me” from behind. If my life is viewed as a residual norm plot, I would not say it is quadratic convergence, not even monotone, but I do converge. I learned a lot during the five years here in WPI, and I am thankful to all those who have ever helped me, supported me, and shared the happy life with me.

The first person I want to thank is my advisor, Dr. Homer F. Walker. He has been very patient in teaching me, very far-sighted in inspiring me, and very tolerant in forgiving my mistakes. Thank you very much for your time and energy spent on me.

I am pleased to thank my dissertation committee members Dr. Mayer Humi, Dr. L. Ramdas Ram-Mohan, Dr. Roger Y. Lui, and Dr. Suzanne L. Weekes, for the detailed instructions and suggestions to my dissertation.

Thanks to Dr. Chao Yang from Lawrence Berkeley National Laboratory for instructions on my PhD project, from which my dissertation subject is derived.

I would also like to thank Dr. Bogdan M. Vernescu, Dr. Marcel Blais, Dr. William W. Farr, Dr. Christopher J. Larsen , Dr. J. J. Malone, Dr. William J. Martin, and Dr. Balgobin Nandram, Dr. Brigitte Servatius for every piece of advice they have ever given me.

Thanks to the WPI Mathematical Sciences Department for the financial support. I wouldn’t have been able to start my PhD study without this.

Thanks to my office mate Casey Richardson, Emily Evans, Gerardo Hernandez, Criselda Toto and Xueying Huang for helping me on Latex compiling as well as my random questions, and all others for sharing the happy and meaningful life here.

At last, I want to thank my family for their warm and firm support to me over the last 25 years.

Contents

1 Introduction **1**
1.1 Overview . . . 1
1.2 Physical Problem Introduction . . . 2
1.3 Mathematical Problem Introduction . . . 4
1.4 Algorithm Introduction . . . 7
1.4.1 The SCF method . . . 8
1.4.2 The Anderson acceleration method . . . 9
1.4.3 The SCF method accelerated by Anderson acceleration . . . 12

2 Anderson Acceleration of General Fixed-Point Iteration **13**
2.1 Anderson Acceleration for General Fixed-Point Problems . . . 13
2.1.1 General fixed-point problems . . . 13
2.1.2 Algorithm description . . . 14
2.1.3 Algorithm comparison . . . 16
2.2 The Linearly Constrained Least-Squares Problem . . . 21
2.2.1 Lagrange multipliers . . . 22
2.2.2 Matrix calculation . . . 23
2.2.3 The null-space method . . . 24
2.2.4 The method of elimination . . . 25
2.2.5 Method comparison . . . 27

3 Numerical Experiments **29**
3.1 KSSOLV Introduction . . . 29
3.2 Method Description . . . 31
3.2.1 Generalized Broyden's method . . . 31
3.2.2 Pulay mixing . . . 32

3.2.3 Kerker mixing 32
3.3 Test of Convergence 33
3.4 Test of Varying Mixdim 38
3.5 Least-squares problem 65

4 Conclusions **71**
4.1 Summary 71
4.2 Future Research 72

List of Tables

2.1 Theoretical Method Comparison 27
2.2 Maximum Observed Condition Numbers 28

3.1 Size of the Test Problems 30

List of Figures

1.1 Example of Charge Density 2

3.1 Charge Density for SiH_4 & Silicon bulk 36
3.2 Charge Density for Quantum Dot & PtNiO 36
3.3 Charge Density for CO_2 & H_2O 37
3.4 Charge Density for HnCO & C_2H_6 37
3.5 Anderson Mixing for SiH_4 41
3.6 Broyden Mixing for SiH_4 41
3.7 Pulay Mixing for SiH_4 42
3.8 Kerker Mixing for SiH_4 42
3.9 Pulay + Kerker Mixing for SiH_4 43
3.10 Anderson Mixing for Silicon Bulk 44
3.11 Broyden Mixing for Silicon Bulk 44
3.12 Pulay Mixing for Silicon Bulk 45
3.13 Kerker Mixing for Silicon Bulk 45
3.14 Pulay + Kerker Mixing for Silicon Bulk 46
3.15 Anderson Mixing for Quantum Dot 47
3.16 Broyden Mixing for Quantum Dot 47
3.17 Pulay Mixing for Quantum Dot 48
3.18 Kerker Mixing for Quantum Dot 48
3.19 Pulay + Kerker Mixing for Quantum Dot 49
3.20 Anderson Mixing for PtNiO 50
3.21 Broyden Mixing for PtNiO 50
3.22 Pulay Mixing for PtNiO 51
3.23 Kerker Mixing for PtNiO 51
3.24 Pulay + Kerker Mixing for PtNiO 52

3.25 Anderson Mixing for CO_2 53
3.26 Broyden Mixing for CO_2 53
3.27 Pulay Mixing for CO_2 54
3.28 Kerker Mixing for CO_2 54
3.29 Pulay + Kerker Mixing for CO_2 55
3.30 Anderson Mixing for H_2O 56
3.31 Broyden Mixing for H_2O 56
3.32 Pulay Mixing for H_2O 57
3.33 Kerker Mixing for H_2O 57
3.34 Pulay + Kerker Mixing for H_2O 58
3.35 Anderson Mixing for HNCO 59
3.36 Broyden Mixing for HNCO 59
3.37 Pulay Mixing for HNCO 60
3.38 Kerker Mixing for HNCO 60
3.39 Pulay + Kerker Mixing for HNCO 61
3.40 Anderson Mixing for C_2H_6 62
3.41 Broyden Mixing for C_2H_6 62
3.42 Pulay Mixing for C_2H_6 63
3.43 Kerker Mixing for C_2H_6 63
3.44 Pulay + Kerker Mixing for C_2H_6 64

Chapter 1

Introduction

1.1 Overview

In electronic structure computations, people often need to set up nonlinear eigenvalue problems. The currently prevailing method for finding approximate solutions to these problems is the Self-Consistent Field (SCF) method accelerated by the Anderson acceleration method, described in [25], [24] and [5].

Acceleration methods are a group of methods that are often applied to iterative methods, to improve the rate of convergence, given that the convergence of many iterative methods are slow.

The Anderson acceleration method first appeared in [1] in 1965. It has since been applied to electronic structure computations and appears in the survey paper [5] in 2008. In electronic struture computations, the Anderson acceleration method and other acceleration methods are also known as "mixing" methods.

Some other acceleration methods have also been considered under the name of vector extrapolation, which fall into two categories: the polynomial methods and the ϵ-algorithms. The first category includes the reduced-rank extrapolation (RRE), minimal-polynomial extrapolation (MPE), and modified minimal-polynomial extrapolation (MMPE) methods; the second category includes topological ϵ-algorithm (TEA) and the scalar and vector ϵ-algorithms (SEA and VEA). See more details in [9] [22] [21] [8] [20].

In this dissertation, we outline some analytical as well as numerical studies on the Anderson acceleration method applied to iterative methods. In Chapter 1, we introduce a nonlinear eigenvalue problem encountered in electronic structure computations and the currently prevailing Self-Consistent Field (SCF) method accelerated by the Anderson acceleration method. New convergence results for the Anderson acceleration method on linear systems are shown in Section 2.1. A linearly constrained least-squares problem embedded in the Anderson routine is studied in Section 2.2. We do numerical experiments to study the convergence properties of the Anderson acceleration method in Chapter 3. Finally, we summarize our work and outline future work in chapter 4.

1.2 Physical Problem Introduction

In the field of nanomaterials and devices, one of the technical problems involves fabrication of the devices. Nanomaterials can be created in a laboratory environment, but researchers cannot determine at the outset exactly what structure they will get. Consequently, people often need to identify unknown electronic structures of materials using computational models.

Every material has a unique "ground state" — that is, the minimum-energy level where the material's electrons remain unless the material is perturbed by external sources. If it is possible to determine a material's ground state, then it is possible to identify the material itself. (See more in [23].)

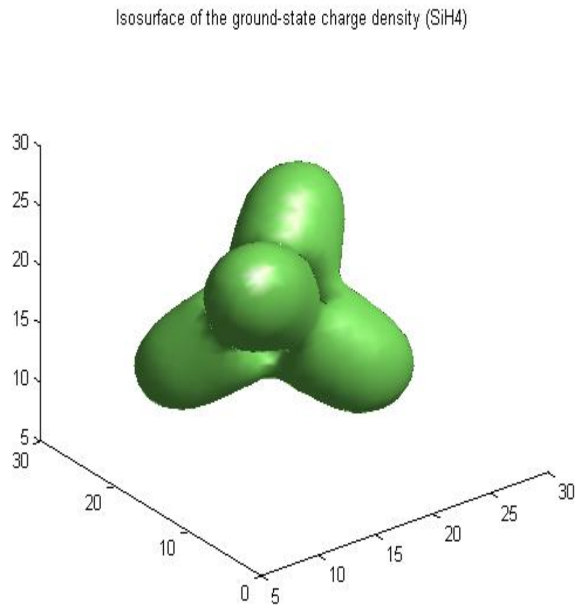

Figure 1.1: Example of Charge Density

A current approach is to find out the charge density associated with the ground state, and this involves setting up nonlinear eigenvalue problems in the process. Figure 1.1 shows the ground state charge density solved from a nonlinear eigenvalue problem.

For the charge density, we would like to introduce the density functional theory (DFT). In the past few decades, density functional theory has become one of the most widely used methods for the calculation of the properties of the complex electronic systems. The basic idea, introduced by Hohenberg, Kohn, and Sham in the 1960s, is to describe the system in terms of the electronic density without explicit reference to the many-body wave function. See details in [14], [7], [13], [11], [2] and [19].

The idea of DFT is to transform from the ground state wave function ψ_0 to the single-particle density function ρ_0. This transformation has obvious advantages for many-particle systems, it allows the knowledge, methods, and concepts developed for single-particle systems to be applied. For one-partical systems, it allows the exact ground state energy E_0 to be determined via the variational principle with respect to the single-particle density. For single-particle systems:

$$\rho_0(x) = |\psi_0(x)|^2, \tag{1.1}$$

where x denotes the position of the particle. The density function $\rho_0(x)$ must vanish at the boundaries of the region occupied by the particle and satisfies the normalization condition:

$$\int \rho_0(x)dx = 1. \tag{1.2}$$

The momentum operator in quantum mechanics is defined by

$$p_x \equiv -2\hbar \frac{\partial}{\partial x} \tag{1.3}$$

where $\hbar$ is the Planck's constant divided by 2π ($1.054571628 \times 10^{-34}$ Js). The wave function ψ_0 satisfies the time independent Schrödinger equation:

$$\left[-\frac{p_x^2}{2m} + V(x)\right] \psi_0(x) = E_0\psi_0(x), \tag{1.4}$$

or

$$\left[-\frac{\hbar^2}{2m}\frac{d^2}{dx^2} + V(x)\right] \psi_0(x) = E_0\psi_0(x), \tag{1.5}$$

where $V(x)$ is the external potential energy for the particle of mass m. We consider systems where ψ_0 is real, so that the energy may be written as

$$E_0 = \int \sqrt{\rho_0(x)} \left[-\frac{\hbar^2}{2m}\frac{d^2}{dx^2} + V(x) \right] \sqrt{\rho_0(x)} dx. \tag{1.6}$$

We now introduce the energy density functional

$$E[\rho] = T[\rho] + \int \rho(x)V(x)dx, \tag{1.7}$$

where the kinetic energy functional $T[\rho]$ is

$$T[\rho] = \int \sqrt{\rho(x)}(-\frac{\hbar^2}{2m}\frac{d^2}{dx^2})\sqrt{\rho(x)}dx. \tag{1.8}$$

The trial density ρ must satisfy the same boundary and normalization conditions as ρ_0. According to the Hohenberg-Kohn theorem, the external potential $V(x)$ is determined uniquely by $\rho_0(x)$, and the energy density functional, $E(\rho)$, satisfies the condition

$$E(\rho) \geq E_0. \tag{1.9}$$

In other words,

$$\min E(\rho) = E_0 = E[\rho_0]. \tag{1.10}$$

Note that we are going to continue our introduction to the specific nonlinear eigenvalue problem with discretized functions (i.e matrices), and thus use a different set of notations in the next section.

1.3 Mathematical Problem Introduction

In most situations in chemistry, it is legitimate to consider the nuclei as classical objects and as point-like particles with charges (λ_1,λ_2,...,λ_p) at positions (x_1,x_2,...,x_p), while treating the electrons as quantum particles. This is the so-called Born-Oppenheimer approximation. In view of this approximation, determining the structure of the ground state (that is, the state of minimum energy) of a molecular system consisting of p nuclei and n electrons amounts to solving a minimization problem. We will describe details in the following paragraphs (also see [4]).

With an appropriate discretization scheme, a single electron wave function can be approximated by a vector $x_i \in \mathbb{R}^n$, where n is the spatial degrees of freedom, i.e., the number of real space grid points. These vectors satisfy the orthonormality constraints:

$$x_i^T x_j = \delta_{i,j}, \quad i, j = 1, 2, \ldots, p,$$

where p is the number of occupied states. If we let $X = (x_1, x_2, \ldots, x_p)$, the matrix

$$D(X) = XX^T \tag{1.11}$$

is often known as the density matrix, and the charge density associated with the p occupied states can be expressed by

$$\rho(X) = \mathrm{diag}(XX^T), \tag{1.12}$$

where the MATLAB notation diag(A) denotes a column vector consisting of diagonal entries of the matrix A.

The Kohn-Sham (KS) total energy function consists of several components:

$$E_{total}(X) = E_{kinetic}(X) + E_{ion}(X) + E_H(X) + E_{XC}(X), \tag{1.13}$$

where $E_{kinetic}$ is the kinetic energy and E_{ion}, E_H, and E_{XC} are potential energies introduced by the electro-ion interation (ionic potential), the electron-electron interaction (Hartree potential), and the exchange correlation potential, respectively.

Let $L \in \mathbb{R}^{n \times n}$ be a Hermitian matrix representing a discretized Laplacian operator. The kinetic energy is then defined by:

$$E_{kinetic}(X) = \frac{1}{2} trace(X^T L X). \tag{1.14}$$

The ionic potential energy consists of a local a nonlocal term. If we let D_{ion} be a real diagonal matrix representing a discretized local ionic potential function, then the local ionic potential energy is defined by:

$$E_{ion(local)}(X) = trace(X^T D_{ion} X). \tag{1.15}$$

The contribution from the nonlocal ionic potential is defined by:

$$E_{ion(nonlocal)}(X) = \sum_i \sum_l |x_i^T \omega_l|^2 \tag{1.16}$$

where ω_l represents a discretized pseudopotential reference projection function.

In practice, L is nonsingular with appropriate boundary conditions. If we use L^{-1} to denote the inverse of the discrete Laplacian operator, then the Hartree potential energy, which is used to model the classical electrostatic average interation between electrons, can be expressed by

$$E_H(X) = \frac{1}{2}\rho(X)^T L^{-1}\rho(X). \tag{1.17}$$

The exchange correlation function ϵ_{xc} is used to model the nonclassical interation between electrons. The potential energy induced by this function is defined by

$$E_{XC}(X) = \vec{1}^T(\epsilon[\rho(X)]), \tag{1.18}$$

where $\vec{1} = (1, 1, \dots, 1)^T$.

Using the notation established above, we can state the KS total energy minimization problem as

$$\left\{ \begin{array}{l} \min E_{total}(X) \\ \text{s.t. } \ X^T X = I_p, \end{array} \right. \tag{1.19}$$

where I_p denotes the $p \times p$ identity matrix.

The Lagrangian associated with (1.19) is

$$\mathcal{L}(X) = E_{total}(X) - \text{trace}[\Lambda^T(X^T X - I_p)], \tag{1.20}$$

where Λ is a $p \times p$ matrix containing the lagrange multipliers associated with the constraints specified by $X^T X = I_p$.

The solution to (1.19) must satisfy the first order necessary conditions

$$\left\{ \begin{array}{l} \nabla_X \mathcal{L}(X) = 0 \\ X^T X = I_p \end{array} \right. \tag{1.21}$$

Here, $\nabla_X \mathcal{L}$ represents an $n \times p$ matrix whose (i, j)th entry is the partial derivative of $\mathcal{L}$ with respect to the i, jth entry of X.

It is easy to verify that

$$
\begin{aligned}
\nabla_X E_{kinetic} &= \frac{1}{2} L X, && (1.22)\\
\nabla_X E_{ion(local)} &= D_{ion} X, && (1.23)\\
\nabla_X E_{ion(nonlocal)} &= \sum_l (\omega_l \omega_l^T) X, && (1.24)\\
\nabla_X E_H &= \mathrm{Diag}(L^{-1}\rho(X)) X, && (1.25)\\
\nabla_X E_{XC} &= \mathrm{Diag}(\mu_{xc}(\rho)) X, && (1.26)\\
& && (1.27)
\end{aligned}
$$

where

$$\mu_{xc}(\omega) \equiv \frac{d\epsilon_{xc}(\omega)}{d\omega} \tag{1.28}$$

is the derivative of the exchange-correlation function. Here the MATLAB notation Diag(ρ) represents a diagonal matrix whose diagonal is determined by the vector ρ, and we have scaled (1.22)–(1.26) by $\frac{1}{2}$ to be consistent with the convention used in the electronic structure community.

Substituting (1.22)–(1.26) into (1.21), we obtain the Kohn-Sham equation

$$\left\{ \begin{array}{l} H(\rho)X = X\Lambda_p \\ X^T X = I_p \end{array} \right. , \tag{1.29}$$

where

$$H(X) = [\frac{1}{2}L + D_{ion} + \sum_l \omega_l \omega_l^T + \mathrm{Diag}(L^{-1}\rho) + \mathrm{Diag}(\mu_{xc}(\rho))]. \tag{1.30}$$

In the Khon-Sham equation, the columns of $X \in \mathbb{R}^{n \times p}$ $(p < n)$ are approximate electron wave functions, $\rho = diag(XX^T) \in \mathbb{R}^{n\times 1}$ is the charge density, $H \in \mathbb{R}^{n\times n}$ is the discrete Hamiltonian and is dependent on ρ, $\Lambda_p \in \mathbb{R}^{p\times p}$ is a diagonal matrix with the p smallest eigenvalues of H on the diagonal, and $I_p \in \mathbb{R}^{p\times p}$ is the identity matrix. We want to solve for the charge density ρ.

1.4 Algorithm Introduction

Current approaches to this problem include the Self-Consistent Field (SCF) method ([25]) and the SCF method accelerated by the Anderson acceleration method [5].

In each iteration of the SCF method, we fix the Hamiltonian H, find the p smallest eigenvalues and the corresponding eigenvectors to form X_+ and thus ρ_+, use ρ_+ to update H and enter next iteration. The idea of Anderson acceleration is to save these ρ_+'s from previous iterations, let ρ_+^{AA} be a linear combination of them, with some correction vector, and use ρ_+^{AA} to update H before next iteration. Here, the superscript "AA" stands for "Anderson Acceleration".

In the following, we will introduce the details of the two algorithms. Some MATLAB notations are used. In particular, for $A \in \mathbb{R}^{p\times p}$, $\text{diag}(A)$ denotes the vector in $\mathbb{R}^p$ the components of which are the diagonal entries of A, $\text{size}(A, 2)$ denotes the number of columns of A, $\text{cond}(A)$ denotes the condition number of A, and $[\,]$ denotes the "empty" matrix.

1.4.1 The SCF method

This method is called the Self-Consistent Field (SCF) method, because the calculational strategy adopted is to seek self-consistency in each iteration. It is formulated as follows:

Algorithm 1 The SCF Method

```
Given $X \in \mathbb{R}^{n\times p}$, and $tol > 0$, evaluate $\rho = \text{diag}(XX^T)$ and $H(\rho) \in \mathbb{R}^{n\times n}$.
for $iter = 1$ to $Max\_iter$ do
    Find the $p$ smallest eigenvalues and the corresponding orthonormal eigenvectors of
    $H(\rho)$ to form the columns of $X_+ \in \mathbb{R}^{n\times p}$.
    Evaluate $\rho_+ = \text{diag}(X_+X_+^T)$, and $\Delta\rho = \rho_+ - \rho$.
    Update $\rho \longleftarrow \rho_+$.
    if $||\Delta\rho|| < tol$ then
        Break.
    end if
    Update $H(\rho) \longleftarrow H(\rho_+)$.
end for
```

There is one essential process in the SCF method, which is updating ρ. In each iteration, we first have a value of ρ, then we evaluate the matrix $H(\rho)$. After that, we solve the eigenvalue problem for X_+ and evaluate ρ_+. We represent the process as follows:

$$\rho \longrightarrow H(\rho) \longrightarrow H(\rho)X_+ = X_+\Lambda_+ \longrightarrow \rho_+ = diag(X_+X_+^T)$$

Based on the SCF method, we define a function S such that:

$$\rho_+ = S(\rho) \tag{1.31}$$

Now the nonlinear eigenvalue problem becomes a fixed-point problem: $\rho_* = S(\rho_*)$. The SCF method becomes a fixed-point iteration:

Algorithm 2 Recast SCF Method

```
Given $\rho \in \mathbb{R}^{n\times 1}$, and $tol > 0$, and evaluate $H(\rho) \in \mathbb{R}^{n\times n}$.
for $iter = 1$ to $Max\_iter$ do
    Let $\rho_+ = S(\rho)$; let $\Delta\rho = S(\rho) - \rho$.
    Update $\rho \longleftarrow \rho_+$.
    if $||\Delta\rho|| < tol$ then
        Break.
    end if
end for
```

In practice, the SCF method may be less effective than desired: the convergence may be slow, and sometimes the iterates even diverge. So for the following parts of this dissertation, we will focus on the Anderson acceleration method, where SCF is augmented with a certain procedure to improve convergence.

1.4.2 The Anderson acceleration method

We will introduce this algorithm with mostly the same notation as [5] here. See [1] for the original formulation. The idea of Anderson acceleration is to make use of storages from previous iterations. On one hand, we need sufficient number of storages, so that we have enough information to predict a next iterate; on the other hand, we do not store all of the previous iterates, since the early iterates may contain less predictive informate.

Consider a procedure for solving a large nonlinear system of equations

$$f(x) = 0 \tag{1.32}$$

($x, f \in \mathbb{R}^n$) by an iterative process. The $m+1$ most recent iterates are denoted by $x_{k-m}, \ldots, x_k \in \mathbb{R}^n$ and the corresponding outputs $f_{k-m}, \ldots, f_k \in \mathbb{R}^n$. Assuming evaluating $f(x)$ is expensive and no explicit analytic form of $f(x)$ is available, the Anderson acceleration method determines the next estimate x_{k+1} in the following way.

Denote:

$$\bar{x}_k = x_k - \sum_{i=k-m}^{k-1} \gamma_i^{(k)} \Delta x_i = x_k - \mathscr{X}_k \gamma^{(k)} \tag{1.33}$$

$$\bar{f}_k = f_k - \sum_{i=k-m}^{k-1} \gamma_i^{(k)} \Delta f_i = f_k - \mathscr{F}_k \gamma^{(k)}, \tag{1.34}$$

where

$$\Delta x_i = x_{i+1} - x_i \tag{1.35}$$

$$\Delta f_i = f_{i+1} - f_i \tag{1.36}$$

$$\gamma^{(k)} = (\gamma_{k-m}^{(k)}, \ldots, \gamma_{k-1}^{(k)}), \tag{1.37}$$

and

$$\mathscr{X}_k = [\Delta x_{k-m} \ldots \Delta x_{k-1}] \tag{1.38}$$

$$\mathscr{F}_k = [\Delta f_{k-m} \ldots \Delta f_{k-1}]. \tag{1.39}$$

By rearranging, we get

$$\bar{x}_k = \sum_{j=k-m}^{k} \alpha_j x_j \tag{1.40}$$

$$\bar{f}_k = \sum_{j=k-m}^{k} \alpha_j f_j, \tag{1.41}$$

where $\sum_{j=k-m}^{k} \alpha_j = 1$, so that $\bar{x}_k$ and $\bar{f}_k$ can be viewed as weighted averages of the x_j's and f_j's.

The γ_i's are determined from the minimization problem:

$$\min_{\gamma^{(k)}} E(\gamma^{(k)}) = \min_{\gamma^{(k)}} < \bar{f}_k, \bar{f}_k > = \min_{\gamma^{(k)}} ||f_k - \mathscr{F}_k \gamma^{(k)}||_2^2, \tag{1.42}$$

whose normal equation is

$$(\mathscr{F}_k^T \mathscr{F}_k) \gamma^{(k)} = \mathscr{F}_k^T f_k. \tag{1.43}$$

From all the above, we obtain our update:

$$
\begin{aligned}
x_{k+1} &= \bar{x}_k + \beta \bar{f}_k && (1.44)\\
&= x_k + \beta f_k - (\mathscr{X} + \beta \mathscr{F}_k)\gamma^{(k)} && (1.45)\\
&= x_k + \beta f_k - (\mathscr{X} + \beta \mathscr{F}_k)(\mathscr{F}_k^T \mathscr{F}_k)^{-1} \mathscr{F}_k^T f_k. && (1.46)
\end{aligned}
$$

Here, $\beta > 0$ is a parameter of the Anderson acceleration method.

When we view the nonlinear system (1.32) as a fixed-point problem:

$$x_* = G(x_*) = f(x_*) + x_*, \tag{1.47}$$

the Anderson acceleration method will look differently.

Denote $G(x_j) = G_j$. The minimization (1.42) for $\gamma^{(k)}$ is equivalent to the following minimization problem for $\alpha = (\alpha_{k-m}, \ldots, \alpha_k)^T$:

$$\min ||\bar{f}_k|| = \min || \sum_{j=k-m}^{k} \alpha_j f_j || \qquad \text{s.t.} \sum_{j=k-m}^{k} \alpha_i = 1, \tag{1.48}$$

or

$$\min || \sum_{j=k-m}^{k} \alpha_j (G_j - x_j) || \qquad \text{s.t.} \sum_{j=k-m}^{k} \alpha_i = 1. \tag{1.49}$$

The update is:

$$
\begin{aligned}
x_{k+1} &= \bar{x}_k + \beta \bar{f}_k && (1.50)\\
&= \sum_{j=k-m}^{k} \alpha_j x_j + \beta \sum_{j=k-m}^{k} \alpha_j (G_j - x_j) && (1.51)\\
&= \sum_{j=k-m}^{k} \alpha_j G_j - (1-\beta) \sum_{j=k-m}^{k} \alpha_j (G_j - x_j) && (1.52)
\end{aligned}
$$

If $\beta = 1$, this update becomes simply:

$$x_{k+1} = \sum_{j=k-m}^{k} \alpha_j G_j \tag{1.53}$$

1.4.3 The SCF method accelerated by Anderson acceleration

Based on the SCF method, we apply the Anderson acceleration method to the fixed-point problem $\rho_* = S(\rho_*)$ to strengthen self-consistency at each iteration.

Algorithm 3 The SCF Method Accelerated by Anderson Acceleration

```
Given $\rho \in \mathbb{R}^{n\times 1}$, $tol > 0$, $condtol > 0$, $\beta > 0$ and Mixdim $> 0$. Set $K = [\,]$, $D = [\,]$.
for $iter = 1$ to $Max\_iter$ do
    Evaluate $\rho^{SCF} = S(\rho)$, $\Delta\rho = \rho^{SCF} - \rho$.
    if $||\Delta\rho|| < tol$ then
        Update $\rho \longleftarrow S(\rho)$; break.
    end if
    Update $K \longleftarrow [K, \rho^{SCF}]$, $D \longleftarrow [D, \Delta\rho]$.
    while size$(D, 2) >$ Mixdim or cond$(D) > condtol$ do
        Delete the first columns of $K$ and $D$.
    end while
    Find $\alpha \in \mathbb{R}^{k\times 1}$ from $\begin{cases} \min_\alpha ||D\alpha|| \\ \text{s.t.} \sum \alpha_i = 1 \end{cases}$.
    Evaluate $\rho_+ = K\alpha - (1-\beta)D\alpha$.
    Update $\rho \longleftarrow \rho_+$.
end for
```

Here, Mixdim stands for mixing dimension, or, the maximum number of vectors stored by Anderson acceleration, since the Anderson acceleration method is also known as Anderson mixing. Mixdim $= m$ in the previous section.

For the tolerance of the condition number, we use $condtol = 10^{16}$, since 10^{-16} is the machine epsilon for MATLAB.

Chapter 2

Anderson Acceleration of General Fixed-Point Iteration

2.1 Anderson Acceleration for General Fixed-Point Problems

2.1.1 General fixed-point problems

A general fixed-point problem has the following form:

$$x_* = G(x_*), \tag{2.1}$$

$$G:\ \mathbb{R}^{n\times 1} \to \mathbb{R}^{n\times 1} \tag{2.2}$$

The corresponding fixed-point iteration is:

Algorithm 4 Fixed-Point Iteration

```
Given $x \in \mathbb{R}^{n\times 1}$ and $tol > 0$.
for $iter = 1$ to $Max\_iter$ do
    Evaluate $x_+ = G(x)$.
    if $||x_+ - x|| < tol$ then
        Break.
    end if
    Update $x \longleftarrow x_+$.
end for
```

Now we apply the Anderson acceleration method to the general fixed-point iteration.

We use iteration indices from here for proof purposes later in this section.

Algorithm 5 Anderson Acceleration for General Fixed-Point Iteration

```
Given $x_0 \in \mathbb{R}^{n\times 1}$, $tol > 0$, $\beta > 0$ and Mixdim $> 0$. Set $K = [\,]$, $D = [\,]$.
for $k = 1$ to $Max\_iter$ do
    Evaluate $x_k^F = G(x_{k-1})$, and let $r_{k-1} = x_k^F - x_{k-1}$.
    if $||r_{k-1}|| < tol$ then
        Update $x \longleftarrow x_k^F$; break.
    end if
    Update $K \longleftarrow [K, x_k^F]$, $D \longleftarrow [D, \Delta r_{k-1}]$.
    while size$(D, 2) >$ Mixdim do
        Delete the first columns of $K$ and $D$.
    end while
    Find a column vector $\alpha$ from $\begin{cases} \min_\alpha ||D\alpha|| \\ \text{s.t.} \sum \alpha_i = 1 \end{cases}$.
    Evaluate $x_k = K\alpha - (1-\beta)D\alpha$.
end for
```

Here, the α's are different in each iteration, but we omit the superscript k for convenience.

The convergence properties of the Anderson acceleration method for general fixed-point problems constitute a very broad topic, so we start with the linear case.

2.1.2 Algorithm description

A linear fixed-point problem is:

$$x_* = G(x_*) = Ax_* + b \tag{2.3}$$

where $A \in \mathbb{R}^{n\times n}$, $b \in \mathbb{R}^{n\times 1}$.

We will study the Anderson acceleration convergence properties on this problem by comparing it with the Generalized Minimal Residual (GMRES) algorithm ([18]) applied to the equivalent linear system $(A - I)x + b = 0$.

In order to be able to compare Anderson acceleration with the GMRES algorithm, we do not include the column dropping strategy here (i.e., let Mixdim $= \infty$), and choose $\beta = 1$. The Anderson acceleration method is as follows:

Algorithm 6 Anderson acceleration for Linear Fixed-Point Iteration

```
Given $x_0 \in \mathbb{R}^{n\times 1}$, $tol > 0$ and $\beta > 0$. Set $K = [\,]$, $D = [\,]$.
for $k = 1$ to $Max\_iter$ do
    Evaluate $x_k^F = Ax_{k-1} + b$, and let $r_{k-1} = x_k^F - x_{k-1} = (A - I)x_{k-1} + b$.
    if $||r_{k-1}|| < tol$ then
        Update $x \longleftarrow x_k^F$; break.
    end if
    Update $K \longleftarrow [K, x_k^F]$, $D \longleftarrow [D, r_{k-1}]$.
    Find column vector $\alpha$ from $\begin{cases} \min_\alpha ||D\alpha|| \\ \text{s.t.} \sum \alpha_i = 1 \end{cases}$.
    Evaluate $x_k = K\alpha$.
end for
```

In the linear case, the fixed point problem $x_* = Ax_* + b$ is equivalent to the linear system $(A - I)x_* + b = 0$. Before introducing the details of the GMRES method, we define the Krylov subspace for this linear system.

Definition 2.1.1. *For a linear system $(A - I)x + b = 0$, $A \in \mathbb{R}^{n\times n}$ and $b, x \in \mathbb{R}^{n\times 1}$, given an initial guess x_0, we define the k^{th} Krylov Subspace:*

$$\mathcal{K}^k = Span\{r_0, (A - I)r_0, \ldots, (A - I)^{k-1}r_0\} \tag{2.4}$$

where $r_0 = (A - I)x_0 + b$.

The GMRES method is a Krylov Subspace method, and here is the specific algorithm for this linear system.

Algorithm 7 The GMRES Algorithm

```
Given $x_0 \in \mathbb{R}^{n\times 1}$, and $tol > 0$. Evaluate $r_0 = (A - I)x_0 + b$.
for $k = 1$ to $Max\_iter$ do
    Find $z_k \in \mathcal{K}^k$ such that $||r_k^{GMRES}|| = ||(A - I)(x_0 + z_k) + b||$ is minimal.
    if $||r_k^{GMRES}|| < tol$ then
        Let $x_k^{GMRES} = x_0 + z_k$; break.
    end if
end for
```

2.1.3 Algorithm comparison

In this subsection, we develop several theorems to compare the Anderson acceleration method and the GMRES algorithm. We assume that the same initial guess x_0 is used for both algorithms, and this assumption holds throughout. Also, superscripts are used to indicate iterates, residuals etc. generated by Anderson acceleration and GMRES whenever there is a possibility of confusion.

Theorem 2.1.2. *In each iteration of the Anderson acceleration method, let A, the r_i's $(i = 1, \dots, k)$, and α be as defined above. We have:*

$$r_k = A(\alpha_1 r_0 + \cdots + \alpha_k r_{k-1}) \in \mathcal{K}^{k+1} \tag{2.5}$$

for each k until convergence.

Proof: We plan to prove this theorem by induction, so we first analyze the recursive relationship between r_k and the r_i's $(i < k)$.

$$\begin{aligned} x_k^F &= G(x_k) = Ax_k + b & (2.6)\\ x_k &= \alpha_1 x_1^F + \cdots + \alpha_k x_k^F & (2.7)\\ &= A(\alpha_1 x_0 + \cdots + \alpha_k x_{k-1}) + b & (2.8) \end{aligned}$$

$$\begin{aligned} r_k &= G(x_k) - x_k & (2.9)\\ &= (A - I)x_k + b & (2.10)\\ &= (A - I)[A(\alpha_1 x_0 + \cdots + \alpha_k x_{k-1}) + b] + b & (2.11)\\ &= A[(A - I)(\alpha_1 x_0 + \cdots + \alpha_k x_{k-1}) + b] & (2.12)\\ &= A(\alpha_1 r_0 + \cdots + \alpha_k r_{k-1}) & (2.13) \end{aligned}$$

When $k = 0$, $r_0 \in \text{Span}\{r_0\} = \mathcal{K}^1$; when $k = 1$, $r_1 = A(\alpha_1 r_0) = \alpha_1 r_0 + (A - I)\alpha_1 r_0 \in \text{Span}\{r_0, (A - I)r_0\} = \mathcal{K}^2$. Assume that for some $i > 1$ and all $k \leq i - 1$, we have $r_k \in \mathcal{K}^{k+1}$. Then for $k = i$,

$$\begin{aligned} r_i &= A(\alpha_1 r_0 + \cdots + \alpha_i r_{i-1}) & (2.14)\\ &= (\alpha_1 r_0 + \cdots + \alpha_i r_{i-1}) & \\ &\quad + (A - I)(\alpha_1 r_0 + \cdots + \alpha_i r_{i-1}), & (2.15) \end{aligned}$$

and by the assumption, we get $r_i \in \mathcal{K}^{i+1}$.

By induction, we have for each iteration of the Anderson method until convergence:

$$r_k = A(\alpha_1 r_0 + \cdots + \alpha_k r_{k-1}) \in \mathcal{K}^{k+1}. \tag{2.16}$$

□

In GMRES, similarly,

$$r_k^{GMRES} = (A - I)(x_0 + z_k) + b = r_0 + (A - I)z_k \in \mathcal{K}^{k+1}, \tag{2.17}$$

since we have $z_k \in \mathcal{K}^k$.

Remark 2.1.3. *In Theorem 2.1.2, convergence of the Anderson acceleration iterates may not necessarily imply that the solution is reached. There could be stagnation, in which case $r_k = r_{k-1}$ for some k and the Anderson acceleration method breaks down.*

Now we know enough about the residual r_k, let's compare the iterates in both methods. In the Anderson acceleration method, denote $z_k^{AA} \equiv (x_k - x_0)$.

Theorem 2.1.4. *In each iteration of the Anderson acceleration method, $z_k^{AA} \in \mathcal{K}^k$ for each k until convergence.*

Proof: In each iteration of the Anderson acceleration method,

$$\begin{aligned}
z_k^{AA} &= x_k - x_0 && (2.18)\\
&= A(\alpha_1 x_0 + \cdots + \alpha_k x_{k-1}) + b - x_0 && (2.19)\\
&= (A - I)(\alpha_1 x_0 + \cdots + \alpha_k x_{k-1}) + b \\
&\quad + (\alpha_1 x_0 + \cdots + \alpha_k x_{k-1}) - x_0 && (2.20)\\
&= (\alpha_1 r_0 + \cdots + \alpha_k r_{k-1}) + (\alpha_1 x_0 + \cdots + \alpha_k x_{k-1}) - x_0 && (2.21)\\
&= (\alpha_1 r_0 + \cdots + \alpha_k r_{k-1}) + (\alpha_2 z_1 + \cdots + \alpha_k z_{k-1}) && (2.22)
\end{aligned}$$

When $k = 1$,

$$\begin{aligned}
z_1^{AA} &= x_1 - x_0 && (2.23)\\
&= G(x_0) - x_0 && (2.24)\\
&= (A - I)x_0 + b && (2.25)\\
&= r_0 \ \in \mathcal{K}^1 && (2.26)
\end{aligned}$$

Assume that, for some $i > 1$, $z_k^{AA} \in \mathcal{K}^k$ for $k \leq i-1$. Then it follows from equation (2.22) and Theorem 2.1.2 that $z_i^{AA} \in \mathcal{K}^i$. By induction, we have, in each iteration of the Anderson acceleration method, $z_k^{AA} \in \mathcal{K}^k$. □

It follows from the theorem that $\text{Span}\{z_1, \ldots, z_k\} \subseteq \mathcal{K}^k$. In the next theorem, we want to prove $\text{Span}\{z_1, \ldots, z_k\} = \mathcal{K}^k$. We need the following lemma.

Lemma 2.1.5. *If in the k^{th} Anderson iteration, $\alpha_k^k = 0$, (the superscript k stands for the iteration number), then the Anderson iterates have converged, i.e., $x_k = x_{k-1}$.*

Proof: If $\alpha_k^k = 0$, then we have:

$$\begin{aligned} \alpha^k &= \underset{\sum_{i=1}^{k} \alpha_i = 1}{\text{argmin}} \ ||\sum_{i=1}^{k} \alpha_i r_{i-1}|| && (2.27)\\ &= \underset{\sum_{i=1}^{k-1} \alpha_i = 1}{\text{argmin}} \ ||\sum_{i=1}^{k-1} \alpha_i r_{i-1}|| && (2.28)\\ &= \alpha^{k-1} && (2.29) \end{aligned}$$

so that

$$\begin{aligned} x_k &= \alpha_1^k x_1^F + \cdots + \alpha_k^k x_k^F && (2.30)\\ &= \alpha_1^k x_1^F + \cdots + \alpha_{k-1}^k x_{k-1}^F && (2.31)\\ &= \alpha_1^{k-1} x_1^F + \cdots + \alpha_{k-1}^{k-1} x_{k-1}^F && (2.32)\\ &= x_{k-1}. && (2.33) \end{aligned}$$

This implies the convergence of the Anderson iterates. □

Theorem 2.1.6. *In each iteration of the Anderson acceleration method, Span$\{z_1^{AA}, \ldots, z_k^{AA}\} = \mathcal{K}^k$ for each k until convergence.*

Proof: The superscript "AA" will be omitted in the proof for convenience purposes. Readers should keep in mind that $r_0 = (A-I)x_0 + b$ is the initial residual for both the Anderson and GMRES algorithms.

From Theorem 2.1.4, we have $z_k \in \mathcal{K}^k$ for each k; therefore $\text{Span}\{z_1, \ldots, z_k\} \subseteq \mathcal{K}^k$. Now we only have to show that $\mathcal{K}^k \subseteq \text{Span}\{z_1, \ldots, z_k\}$ for each k. It is enough to show that $(A-I)^{k-1} r_0 \in \text{Span}\{z_1, \ldots, z_k\}$ for each k.

Note that,

$$\begin{aligned} z_k &= A(\alpha_1 x_0 + \cdots + \alpha_k x_{k-1}) + b - x_0 && (2.34)\\ &= A(\alpha_1 x_0 + \cdots + \alpha_k x_{k-1}) - Ax_0 + (A-I)x_0 + b && (2.35)\\ &= A(\alpha_2 z_1 + \cdots + \alpha_k z_{k-1}) + r_0 && (2.36) \end{aligned}$$

$$\begin{aligned} \alpha_k A z_{k-1} &= z_k - r_0 - A(\alpha_2 z_1 + \cdots + \alpha_{k-1} z_{k-2}) && (2.37)\\ &\in \text{Span}\{z_k, r_0, Az_1, \ldots, Az_{k-2}\} && (2.38) \end{aligned}$$

From Lemma 2.1.5, since $x_k \neq x_{k-1}$, $\alpha_k \neq 0$, then

$$Az_{k-1} \in \text{Span}\{z_k, r_0, Az_1, \ldots, Az_{k-2}\}. \tag{2.39}$$

When $k = 1$, we have:

$$r_0 = z_1 \ \in \text{Span}\{z_1\} \tag{2.40}$$

Assume that, for some $i > 1$ and for $k \leq i-1$, $(A-I)^{k-1} r_0 \in \text{Span}\{z_1, \ldots, z_k\}$, i.e. $(A-I)^{k-1} r_0 = \gamma_1 z_1 + \cdots + \gamma_k z_k$. For the i^{th} iteration:

$$\begin{aligned} (A-I)^{i-1} r_0 &= (A-I)(\gamma_1 z_1 + \cdots + \gamma_{i-1} z_{i-1}) && (2.41)\\ &= A(\gamma_1 z_1 + \cdots + \gamma_{i-1} z_{i-1}) - (\gamma_1 z_1 + \cdots + \gamma_{i-1} z_{i-1}) && (2.42)\\ &\in \text{Span}\{Az_1, \ldots, Az_{i-1}, z_1, \ldots, z_{i-1}\} && (2.43) \end{aligned}$$

By applying (2.39) repeatedly for all Az_j's ($j = 1, \ldots, i-1$), and keeping in mind that $r_0 = z_1$, we have:

$$\begin{aligned} & \text{Span}\{Az_1, \ldots, Az_{i-1}, z_1, \ldots, z_{i-1}\} && (2.44)\\ = \ & \text{Span}\{Az_1, \ldots, Az_{i-2}, z_i, r_0, Az_1, \ldots, Az_{i-2}, z_1, \ldots, z_{i-1}\} && (2.45)\\ = \ & \text{Span}\{Az_1, \ldots, Az_{i-2}, z_1, \ldots, z_{i-1}, z_i\} && (2.46)\\ = \ & \text{Span}\{Az_1, \ldots, Az_{i-3}, z_1, \ldots, z_{i-1}, z_i\} && (2.47)\\ & \vdots && (2.48)\\ = \ & \text{Span}\{z_1, \ldots, z_{i-1}, z_i\} && (2.49) \end{aligned}$$

From (2.41) - (2.49), it follows that $(A-I)^{k-1} r_0 \in \text{Span}\{z_1, \ldots, z_k\}$, so by induction, we have:

$$(A-I)^{k-1}r_0 \in \text{Span}\{z_1,\dots,z_k\} \tag{2.50}$$

and thus $\text{Span}\{z_1,\dots,z_k\} = \mathcal{K}^k$. $\square$

Now let's take a close look at the x_k's generated by the two methods.

Theorem 2.1.7. *For each k until convergence of the Anderson iterates, we have:*

$$x_k^{AA} = G(x_{k-1}^{GMRES}). \tag{2.51}$$

<u>Proof</u>: Again, recall that x_0 and r_0 are the same for both Anderson acceleration and GMRES. In the k^{th} iteration of the Anderson method,

$$r_k^{AA} - r_0 = (A-I)x_k^{AA} + b - (A-I)x_0 - b = (A-I)z_k^{AA} \tag{2.52}$$

We minimize $||D\alpha||$ subject to $\sum_{i=1}^k \alpha_i = 1$. Then,

$$\begin{aligned}
D\alpha &= \alpha_1 r_0^{AA} + \cdots + \alpha_k r_{k-1}^{AA} && (2.53)\\
&= (1-\alpha_2-\cdots-\alpha_k)r_0 + \alpha_2 r_1^{AA} + \cdots + \alpha_k r_{k-1}^{AA} && (2.54)\\
&= r_0 + \alpha_2(r_1^{AA} - r_0) + \cdots + \alpha_k(r_{k-1}^{AA} - r_0) && (2.55)\\
&= (A-I)x_0 + b + (A-I)\alpha_2 z_1^{AA} + \cdots + (A-I)\alpha_{k-1} z_{k-1}^{AA} && (2.56)\\
&= (A-I)(x_0 + \alpha_2 z_1^{AA} + \cdots + \alpha_k z_{k-1}^{AA}) + b && (2.57)
\end{aligned}$$

Here,

$$\alpha = \underset{\alpha=(\alpha_2,\dots,\alpha_k)}{\text{argmin}} ||(A-I)(x_0 + \alpha_2 z_1^{AA} + \cdots + \alpha_k z_{k-1}^{AA}) + b||. \tag{2.58}$$

Since $\text{Span}\{z_1^{AA},\dots,z_{k-1}^{AA}\} = \mathcal{K}^{k-1}$,

$$\alpha_2 z_1^{AA} + \cdots + \alpha_k z_{k-1}^{AA} = \underset{z\in\mathcal{K}^{k-1}}{\text{argmin}} ||(A-I)(x_0+z)+b|| = z_{k-1}^{GMRES} \tag{2.59}$$

So we have:

$$\begin{aligned}
x_k^{AA} &= \alpha_1 x_0^F + \cdots + \alpha_k x_{k-1}^F && (2.60)\\
&= \alpha_1 G(x_0) + \cdots + \alpha_k G(x_{k-1}) && (2.61)\\
&= A(\alpha_1 x_0 + \cdots + \alpha_k x_{k-1}) + b && (2.62)\\
&= A(x_0 + \alpha_2 z_1^{AA} + \cdots + \alpha_k z_{k-1}^{AA}) + b && (2.63)\\
&= A(x_0 + z_{k-1}^{GMRES}) + b && (2.64)
\end{aligned}$$

Thus,

$$x_k^{AA} = G(x_{k-1}^{GMRES}) \qquad (2.65)$$

□

From Theorem 2.1.7, we can conclude that the Anderson iterates have the same convergence behavior as the GMRES iterates on linear systems, unless the GMRES iterates stagnate, as described in Remark 2.1.3.

2.2 The Linearly Constrained Least-Squares Problem

In the Anderson accleration method, there is a linearly constrained least-squares problem:

$$\min_{\alpha} ||D\alpha|| \qquad \text{s.t.} \sum_{i=1}^{k} \alpha_i = 1 \qquad (2.66)$$

where $D \in \mathbb{R}^{n\times k}$, $\alpha = (\alpha_1, \ldots, \alpha_k)^T \in \mathbb{R}^{k\times 1}$. Note that we will use $D = [d_1, \ldots, d_k]$ in this section, and all norms are L2 norm.

There are several approaches to solving this problem. Four are outlined below. The first two deal directly with the constrained least-squares problem and are likely to involve ill-conditioned matrices. The second two reduce the constrained problem on $\mathbb{R}^k$ to an unconstrained problem on $\mathbb{R}^{k-1}$, and allow solution approaches that involve much better conditioned matrices.

The results in this section also appeared in my ICCES'08 abstract [15].

2.2.1 Lagrange multipliers

We apply the Lagrange multipliers method to solve (2.66). This method is applied in [16] (see the general description in [3]).

Set

$$\begin{aligned}\Phi(\alpha,\lambda) &\equiv \frac{1}{2}||D\alpha||^2 - \lambda(\sum_{i=1}^{k}\alpha_i - 1)\\ &= \frac{1}{2}\alpha^T D^T D\alpha - \lambda(\sum_{i=1}^{k}\alpha_i - 1)\end{aligned}$$

In order to minimize $\Phi(\alpha,\lambda)$, we set the gradient to zero:

$$\nabla_\alpha \Phi(\alpha,\lambda) = D^T D\alpha - \lambda \begin{pmatrix}1\\ \vdots\\ 1\end{pmatrix} = 0 \tag{2.67}$$

$$\frac{\partial}{\partial\lambda}\Phi(\alpha,\lambda) = -(\sum_{i=1}^{k}\alpha_i - 1) = 0 \tag{2.68}$$

and this yields:

$$\begin{pmatrix} D^T D & -\vec{1}\\ -\vec{1}^T & 0\end{pmatrix}\begin{pmatrix}\alpha\\ \lambda\end{pmatrix} = \begin{pmatrix}\vec{0}\\ -1\end{pmatrix} \tag{2.69}$$

where $\vec{1} = (1,\ldots,1)^T$, $\vec{0} = (0,\ldots,0)^T$, $\vec{1},\vec{0} \in \mathbb{R}^{k\times 1}$.

This method involves the matrix $\begin{pmatrix} D^T D & -\vec{1}\\ -\vec{1}^T & 0\end{pmatrix}$. This may be ill-conditioned because of the term $D^T D$, which has condition number $\kappa(D^T D) = \kappa(D)^2$.

Obtaining α directly from (2.69) requires $O(nk^2)$ flops to form $D^T D$ and $O(k^3)$ flops to solve (2.69). However, this cost can be reduced in the Anderson context, as follows: One easily obtains from (2.69) that $\alpha = (D^T D)^{-1}\vec{1}/\ \vec{1}^T(D^T D)^{-1}\vec{1}$ (see also Section 2.2.2 below); thus one needs only to solve a system with coefficient matrix $D^T D$ to obtain α. Suppose we have the QR decomposition $D = QR$, where $Q \in \mathbb{R}^{n\times k}$ is orthogonal, i.e., $Q^T Q = I_k$, and $R \in \mathbb{R}^{k\times k}$ is upper-triangular. Then $D^T D = R^T Q^T Q R = R^T R$, and α can be obtained by solving triangular systems with R^T and R. Since D is obtained from its predecessor at the previous iteration by adding a new final column and, if necessary, deleting one or more initial columns, one can update the QR decomposition from the previous iteration in $O(nk)$ flops (see below). Thus at each iteration (other than the first one), one can update the QR decomposition at a cost of $O(nk)$ flops and obtain α by solving triangular systems with R^T and R at an additional cost of $O(k^2)$ flops.

We sketch the steps of the updating, referring the reader to [6] for full details. Suppose that, at some iteration, we have a predecessor matrix D and decomposition $D = QR$ from the previous iteration. Then the updating proceeds as follows:

- When adding a new final column to D, we apply the Gram–Schmidt process to orthogonalize the new final column against the columns of Q. The resulting vector and the orthogonalization coefficients then become new last columns of Q and R, respectively.

- When deleting the first column of D, we also delete the first column of R, so that we still have $D = QR$. Now R is upper-Hessenberg, and we left-multiply R by Givens rotations (see details in §5.1.8 of [6]) to restore R to triangular form. We then right-multiply Q by the transposes of the rotations in reverse order to obtain the final Q.

2.2.2 Matrix calculation

We apply matrix calculation to solve (2.66). Define a function f:

$$f = \frac{1}{2}||D\alpha||^2 = \frac{1}{2}\alpha^T D^T D\alpha \tag{2.70}$$

The gradient of f should be orthogonal to the constraint hyperplane $\alpha^T\vec{1} = 1$ at the local minimizer point:

$$\begin{aligned} & \nabla f = D^T D\alpha = \lambda\vec{1} && (2.71)\\ \Rightarrow\quad & \alpha = (D^T D)^{-1}(\lambda\vec{1}) && (2.72)\\ & && (2.73)\end{aligned}$$

Apply the α value to the constraint:

$$\begin{aligned} & 1 = \vec{1}^T\alpha = \lambda\vec{1}^T(D^T D)^{-1}\vec{1} && (2.74)\\ \Rightarrow\quad & \alpha = \frac{(D^T D)^{-1}\vec{1}}{\vec{1}^T(D^T D)^{-1}\vec{1}} && (2.75)\end{aligned}$$

However, when solving for $(D^T D)^{-1}\vec{1}$, we again may meet with high condition numbers, since $\kappa(D^T D) = \kappa(D)^2$.

2.2.3 The null-space method

We apply the null-space method to solve (2.66). The general approach of the null-space method is described in [3]. The basic idea of the method is to decompose the vector we want into the sum of a vector that satisfies the constraint and another vector in the null space of the constraint matrix. Thus the constrained problem becomes one of solving for the vector in the null space. By choosing a basis of the null space, one can then reduce the problem to an unconstrained, lower-dimensional problem of finding a minimizing linear combination of basis vectors.

For this least-squares problem, we introduce a particular implementation of the null-space method that avoids the ill-conditioning of the previous approaches and has other numerical advantages.

Denote $v = (0, \ldots, 0, 1)^T \in \mathbb{R}^{k\times 1}$ and set $\vec{1} = (1, \ldots, 1)^T \in \mathbb{R}^{k\times 1}$ as before. Write $\alpha = v + \beta$, where β is in the null space of $\vec{1}^T$, i.e., $\vec{1}^T\beta = 0$, since $v^T\vec{1} = 1$. If $V \in \mathbb{R}^{k\times(k-1)}$ is full-rank and such that $\vec{1}^T V = 0$, then the columns of V constitute a basis of the null space of $\vec{1}^T$, and we can write $\beta = V\gamma$, where $\gamma \in \mathbb{R}^{(k-1)\times 1}$. Then the minimization problem becomes:

$$\min_\alpha ||D\alpha|| = \min_\gamma ||D(v + V\gamma)|| = \min_\gamma ||d_k + DV\gamma|| \tag{2.76}$$

where $d_k = Dv$. Note that, with of our choice of v, d_k is just the last column of D and thus is available at no cost.

We choose V so that $V = (v_1, \ldots, v_{k-1})$, where

$$v_j = \begin{pmatrix} -1/\sqrt{j(j+1)} \\ \vdots \\ -1/\sqrt{j(j+1)} \\ \sqrt{j/(j+1)} \\ 0 \\ \vdots \\ 0 \end{pmatrix} \left.\vphantom{\begin{matrix} a \\ b \\ c \end{matrix}}\right\} j \text{ components}, \qquad j = 1, \ldots, k-1. \tag{2.77}$$

It is easily verified that $\vec{1}^T V = 0$ and V is full-rank; moreover, $V^T V = I_{k-1}$.

The normal equation of this unconstrained minimization problem is:

$$(DV)^T(DV)\gamma = -(DV)^T Dv \tag{2.78}$$

This has condition number $\kappa((DV)^T(DV)) = \kappa(DV)^2$. One can obtain a better-conditioned system with a QR decomposition $DV = QR$, where as above $Q \in \mathbb{R}^{n\times(k-1)}$ is orthogonal and $R \in \mathbb{R}^{(k-1)\times(k-1)}$ is upper-triangular. Then (2.78) becomes

$$(QR)^T(QR)\gamma = -(QR)^T d_k \tag{2.79}$$

$$\iff \quad R\gamma = -Q^T d_k. \tag{2.80}$$

Thus, we can obtain γ and subsequently $\alpha = v + V\gamma$ by solving a linear system with condition number $\kappa(R) = \kappa(QR) = \kappa(DV) = \sqrt{\kappa(DV)^2}$, which is typically much smaller than that of (2.78). Since $V^T V = 1$, we also have the bound $\kappa(DV) \le \kappa(D)\kappa(V) = \kappa(D)$.

As in the Lagrange multipliers method, one can at each Anderson acceleration iteration obtain the QR decomposition of DV in $O(nk)$ flops by updating a QR decomposition from the previous iteration. In this case, we store Q_0 and R_0 such that $D = Q_0R_0$ in the previous iteration. When D is modified at the current iteration by adding or dropping columns, we update Q_0 and R_0 in $O(nk)$ flops as in the Lagrange multiplier method to obtain $D = Q_0R_0$ for the modified D. Noting that R_0V is upper-Hessenberg since V is upper-Hessenberg, we then apply Givens rotations to R_0V and to Q_0 as in the previous method to obtain $DV = QR$ in $O(nk)$ flops.

2.2.4 The method of elimination

We apply the method of elimination to solve (2.66). The general approach of the method of direct elimination is outlined in [3]. The basic idea of the method is to use the constraint to express some of the variables in terms of others, and then to substitute the expression into the function we want to minimize. Thus the constrained least-squares problem becomes an unconstrained problem in fewer variables. The specific method considered here comes from [12].

Writing $D = (d_1, \ldots, d_k)$, $d_i \in \mathbb{R}^{n\times 1}$, $i = 1, \ldots, k$, we introduce new variables $\bar{\alpha} =$

$(\bar{\alpha}_1, \ldots, \bar{\alpha}_{k-1})^T$ such that:

$$\begin{aligned}
D\alpha &= \sum_{i=1}^{k} \alpha_i d_i \\
&= \alpha_1 d_1 + \alpha_2 d_2 + \cdots + \alpha_k d_k \\
&= \bar{\alpha}_1(d_2 - d_1) + \bar{\alpha}_2(d_3 - d_2) + \cdots + \bar{\alpha}_{k-1}(d_k - d_{k-1}) + d_k \\
&= \bar{D}\bar{\alpha} + d_k
\end{aligned}$$

where $\bar{D} = DW$ and

$$W = \begin{pmatrix} -1 & & & \\ 1 & -1 & & \\ & \ddots & \ddots & \\ & & 1 & -1 \\ & & & 1 \end{pmatrix} \tag{2.81}$$

and

$$\begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_{k-1} \\ \alpha_k - 1 \end{pmatrix} = \begin{pmatrix} -1 & & & \\ 1 & -1 & & \\ & \ddots & \ddots & \\ & & 1 & -1 \\ & & & 1 \end{pmatrix} \begin{pmatrix} \bar{\alpha}_1 \\ \bar{\alpha}_2 \\ \vdots \\ \bar{\alpha}_{k-1} \end{pmatrix} = W\bar{\alpha} \tag{2.82}$$

Then the minimization problem becomes an unconstrained one:

$$\min_{\bar{\alpha} \in \mathbb{R}^{k-1}} ||\bar{D}\bar{\alpha} + d_k|| \tag{2.83}$$

One possibility is to solve the normal equation for $\bar{\alpha}$:

$$\bar{D}^T \bar{D}\bar{\alpha} = -\bar{D}^T d_k, \tag{2.84}$$

and then use equation (2.82) to calculate α from $\bar{\alpha}$. This approach is suggested in §4.2 of [12].

However, this normal equation involves $\kappa(\bar{D}^T\bar{D}) = \kappa(\bar{D})^2$. As in the null-space method, we can improve the condition number with QR decomposition, this time of $\bar{D}$, i.e., $\bar{D} = QR$, where $Q \in \mathbb{R}^{n\times(k-1)}$ is orthogonal and $R \in \mathbb{R}^{(k-1)\times(k-1)}$ is upper-triangular. Then

$$(QR)^T(QR)\bar{\alpha} = -(QR)^T d_k \tag{2.85}$$

$$\iff \quad R\bar{\alpha} = -Q^T d_k. \tag{2.86}$$

Thus one can obtain $\bar{\alpha}$ by solving a linear system with R, which has condition number $\kappa(R) = \kappa(QR) = \kappa(\bar{D}) = \sqrt{\kappa(\bar{D})^2}$. We also have the bound $\kappa(\bar{D}) = \kappa(DW) \leq \kappa(D)\kappa(W)$, and one can show numerically that $\kappa(W) \approx 2k/\pi$ for all but the smallest values of k.

Again, we can avoid doing a direct QR decomposition of $\bar{D}$ at every iteration by making use of the upper-Hessenberg property of W. Specifically, we store Q_0 and R_0 such that $D = Q_0R_0$ and update them at each iteration as in the null-space method, with the upper-Hessenberg W replacing the upper-Hessenberg V.

2.2.5 Method comparison

Here is a table comparing the condition number of the linear systems each method meets with, and the number of arithmetic operations per iteration (for convenience, denote M = Mixdim). In Table 2.1, QR updates are applied to all the four methods.

Methods	Cond. No.	No. of Operations
Lagrange Multipliers	$\approx \kappa(D)^2$	$O(nM) + O(M^3)$
Matrix Calculation	$\kappa(D)^2$	$O(nM) + O(M^2)$
Null-Space Method	$\kappa(DV) \leq \kappa(D)$	$O(nM) + O(M^2)$
Method of Elimination	$\kappa(\bar{D}) \lesssim \frac{2M}{\pi}\kappa(D)$	$O(nM) + O(M^2)$

Table 2.1: Theoretical Method Comparison

The analysis of condition number and number of operations for the four methods indicates that the null-space method and the method of elimination will have better performance, but the difference between these two seems unlikely to be significant.

We also performed some preliminary experiments to support our analytical results with a modification of a code provided by C. Yang at Lawrence Berkeley National Laboratory. Our main interest was in observing the maximum condition numbers encountered by the methods with varying bounds on the maximum allowable value of vector storage Mixdim. Table 2.2 shows typical results, which were obtained in the case of a water molecule. In the table, the first row indicates Mixdim, and the second through fourth rows indicate the maximum condition numbers observed during the iterations.

Mixdim	5	10	15	20
Lagrange Multipliers	1.44e+011	4.70e+014	2.61e+016	2.61e+016
Matrix Calculation	9.72e+007	1.56e+016	6.21e+017	6.21e+017
Null-Space Method	1.19e+003	2.04e+007	1.55e+008	1.55e+008
Method of Elimination	1.01e+003	1.51e+007	1.38e+008	1.38e+008

Table 2.2: Maximum Observed Condition Numbers

From Table 2.2, the maximum condition number comparison results are consistent with our previous analytical results. We also tracked the time expense of the four methods, which did not show significant differences. The main reason is that the Mixdim's are much smaller than n's in our experiments. See Section 3.2 for more experimental results.

There are two widely used approaches to producing a QR decomposition, one based on the Gram-Schmidt process and the other based on Householder transformations. See [6] for descriptions.

For a general QR decomposition $A = QR$, there will be error in the Q matrix, i.e. $Q^TQ = I + E$. Denote the error matrix generated by Gram-Schmidt E^{GS}, and the error matrix generated by Householder E^H. We have:

$$||E^{GS}|| \leq ||A||\epsilon \quad (2.87)$$

$$||E^{H}|| \leq ||A||\kappa(A)\epsilon \quad (2.88)$$

where ϵ denotes machine epsilon. So Householder QR decomposition has less computational error (see [6] for details). In [5], Householder QR decomposition is implemented to mitigate the condition number. However, the condition numbers in our problem are acceptable, it is not necessary to apply Householder QR decomposition.

Chapter 3

Numerical Experiments

3.1 KSSOLV Introduction

Our numerical experiments are based on a MATLAB package "KSSOLV", which is designed to solve the nonlinear eigenvalue problem (1.29). There are 8 materials (test problems) in this package, and the function H's for them are different, depending on different materials. The Self-Consistent Field (SCF) method is applied, and can be accelerated by a number of optional acceleration methods. See detailed descriptions in [24].

In the KSSOLV code, the SCF iteration is slightly different from what is described in Chapter 1. In KSSOLV, each SCF iteration starts with an initial Hamiltonian H_k, then calculates X_k from H_k, evaluates ρ_k, computes the potential v_k^{out}, uses v_k^{out} to obtain the new Hamiltonian H_{k+1}, and goes to next iteration. That is,

$$H_k \to X_k \to \rho_k \to v_k^{out} \to H_{k+1}, \tag{3.1}$$

which can equivalently be viewed as

$$v_k^{out} \to H_{k+1} \to X_{k+1} \to \rho_{k+1} \to v_{k+1}^{out}. \tag{3.2}$$

In (3.2), the essential process is $v_k^{out} \longrightarrow v_{k+1}^{out}$, so in KSSOLV, the SCF method solves the fixed-point problem $v_*^{out} = S(v_*^{out})$, and the Anderson acceleration and other mixing algorithms are applied to the same fixed-point iteration. This does not change the nature of the problem, so we did not explain it at the beginning.

In order to perform our customized experiments, we modified the KSSOLV code, and included the QR decomposition and QR updates in the least-squares subroutine as described in previous chapters.

There are 8 materials in the test problems provided with KSSOLV:

```
sih4_setup.m        --- Silane molecule
sibulk_setup.m      --- Silicon bulk (2 Silicon atoms per cell)
qdot_setup.m        --- A four electron quantum dot with an external potential
ptnio_setup.m       --- PtNiO molecule
co2_setup.m         --- Carbon dioxide molecule
h2o_setup.m         --- Water molecule
hnco_setup.m        --- Isocyanic acid molecule
c2h6_setup.m        --- Ethane molecule
```

Table 3.1 gives the sizes of the test problems (we look at the size of X in equation (1.29)):

Test Problems	No. Columns of X	No. Rows of X
sih4_setup.m	2103	4
sibulk_setup.m	1639	4
qdot_setup.m	2103	8
ptnio_setup.m	4609	43
co2_setup.m	2103	8
h2o_setup.m	2103	4
hnco_setup.m	2103	8
c2h6_setup.m	2103	7

Table 3.1: Size of the Test Problems

We work on the SCF method accelerated by 5 different mixing types implemented in KSSOLV: “Anderson”, “Broyden”, “Pulay”, “Kerker” and “Pulay+Kerker”.

Here, “Anderson” stands for the Anderson acceleration method, which has been introduced in previous chapters; “Broyden” stands for the generalized Broyden's method, which is explained in [5]; “Pulay” stands for the Pulay mixing, which is explained in [16] and [17]; “Kerker” stands for the Kerker mixing, which is explained in [10]; “Pulay+Kerker” is a combination of Pulay mixing and Kerker mixing. Following is a brief description of these methods.

3.2 Method Description

We have already described the Anderson acceleration method in previous chapters; now we will introduce the other methods in this section.

3.2.1 Generalized Broyden's method

Recall that in KSSOLV we have:

$$H_k \to X_k \to \rho_k \to v_k^{out} \to H_{k+1}, \tag{3.3}$$

or briefly:

$$H_k \to v_k^{out} \to H_{k+1}, \tag{3.4}$$

The Broyden's method will insert one mixing step v_k^{in} into it,

$$H_k \to v_k^{out} \to v_k^{in} \to H_{k+1}, \tag{3.5}$$

where v_k^{in} comes from previous storage of v_j^{in}'s and v_j^{out}'s.

We will start with an initial guess v_0^{in} and a given $\beta > 0$. Assume that in the $(k+1)^{th}$ iteration, we have $(v_0^{in}, \dots, v_k^{in})$ and $(v_1^{out}, \dots, v_{k+1}^{out})$ before mixing. Denote

$$\begin{aligned} \Delta F &= (\Delta f_1, \Delta f_2, \dots, \Delta f_k) && (3.6) \\ \Delta V &= (\Delta v_1, \Delta v_2, \dots, \Delta v_k) && (3.7) \end{aligned}$$

where

$$\begin{aligned} \Delta f_k &= v_{k+1}^{out} - v_k^{out} && (3.8) \\ \Delta v_k &= v_k^{in} - v_{k-1}^{in} && (3.9) \end{aligned}$$

Solve the least squares problem:

$$\min_{\gamma=(\gamma_1,\dots,\gamma_k)^T} ||\Delta F\gamma - \beta v_{k+1}^{out}|| \tag{3.10}$$

and evaluate

$$b = -\Delta V\gamma \tag{3.11}$$

$$f = \beta v_{k+1}^{out} - \Delta F\gamma \tag{3.12}$$

Update

$$v_{k+1}^{in} = v_k^{in} - (\beta f - b). \tag{3.13}$$

See more details in [5].

3.2.2 Pulay mixing

Given $x_0, x_1, \ldots, x_k$, denote $\Delta x_i = x_i - x_{i-1}$, $i = 0, \ldots, k$, solve the least-squares problem:

$$\min_{\alpha=(\alpha_1,\ldots,\alpha_k)^T} ||\alpha_i \Delta x_i||, \qquad \text{s.t. } \sum_{i=1}^{k} \alpha_i = 1 \tag{3.14}$$

and update

$$x_{k+1} = \sum_{i=1}^{k} \alpha_i x_i. \tag{3.15}$$

See more details in [16] [17] [24].

3.2.3 Kerker mixing

As mentioned previously, in KSSOLV, the mixing strategy is mixing the vector v^{out} to get v^{in} and go to next iteration:

$$H_k \rightarrow v_k^{out} \rightarrow v_k^{in} \rightarrow H_{k+1}. \tag{3.16}$$

The idea of Kerker mixing is to use $v_k^{res} = v_k^{out} - v_{k-1}^{in}$ to adjust v_{k-1}^{in} to get v_k^{in}:

$$v_k^{in} = v_{k-1}^{in} + v_k^{cor} + 0.8 * v_k^{res}. \tag{3.17}$$

Here, v_k^{cor} is computed from v_k^{res}, by modifying its projection in the FFT space. See more details in [10] [24].

3.3 Test of Convergence

For each of the 8 different materials, we tested with 6 different random initial guesses of ρ, and implemented 5 mixing strategies (with $Mixdim = 9$) as well as SCF alone ("No Mix") under each initial guess. The number of iterations and time spent for each experiment are collected in the following table. In these experiments, we chose the maximum iteration number to be 300 (and this is a reasonable number from our experience), so the iteration number 301 implies failure of covergence.

In the following data, for each material, each column gives results using the same initial guess for all mixing strategies.

```
-------------------------------------------
sih4_setup.m   --- Silane molecule
mixtype             iteration numbers            convergence time
No Mix         252 252 252 252 252 252 |   907   906   906   905   910   896
anderson        11  11  11  11  11  11 |    64    62    63    62    63    63
broyden         14  14  14  14  14  14 |    72    72    74    73    74    73
pulay          301 301 301 301 301 301 |   395   398   393   389   553   572
kerker          68  68  68  68  68  68 |   239   236   239   238   236   237
pulay+kerker    20  20  19  20  20  20 |    97    97    94    97    96    95
-------------------------------------------
sibulk_setup.m --- Silicon bulk (2 Silicon atoms per cell)
mixtype             iteration numbers            convergence time
No Mix         301 301 301 301 301 301 |  1133  1168  1165  1142  1144  1189
anderson        13  12  12  12  12  12 |    61    61    59    59    59    59
broyden         16  16  15  17  17  16 |    74    75    72    79    79    75
pulay          301 301 301 301 301 301 |   418   766   434   449   433   562
kerker          28  28  28  28  28  28 |   104   108   105   103   107   106
pulay+kerker   301 301 301 301 301 301 |  1273  1293  1306  1295  1301  1277
-------------------------------------------
qdot_setup.m   --- A four electron quantum dot with an external potential
mixtype             iteration numbers            convergence time
No Mix         301 301 301 222 221 301 |  1843  1858  1841  1370  1345  1846
anderson       301 301 301 301 301 301 |  1897  1913  1912  1924  1921  1906
```

```
broyden       301 301 277 301 301 257 |  1926  1951  1798  1951  1946  1653
pulay         301 301 301 301 301 301 |  1892  1895  1892  1913  1902  1907
kerker        294 301 257 276 226 301 |  1826  1846  1573  1689  1396  1804
pulay+kerker  210 218 257 198 194 260 |  1332  1391  1623  1254  1242  1653
-------------------------------------------
ptnio_setup.m  --- PtNiO molecule
mixtype             iteration numbers            convergence time
No Mix        301 301 301 301 301 301 | 17308 18293 17639 17135 18304 18279
anderson       69  58  71  74  68  76 |  4076  3592  4443  4568  4064  4610
broyden       301 301 301 301 301 301 | 17590 18269 18108 17572 18509 18318
pulay         301 301 301 301 301 301 |  8517 11375  8315  9975  9629 16368
kerker        301 301 301 301 301 301 | 17268 17443 17350 18317 18094 18251
pulay+kerker  301 301 301 301 301 301 | 17615 18109 18051 18516 18122 18264
-------------------------------------------
co2_setup.m    --- Carbon dioxide molecule
mixtype             iteration numbers            convergence time
No Mix        301 301 301 301 301 301 |  1860  1753  1885  1798  1893  1830
anderson       13  14  13  14  13  13 |    99   105    98   101    99   101
broyden        15  15  15  16  15  16 |   114   111   109   117   113   111
pulay         301 301 301 301 301 301 |   759   953   636   548   771   935
kerker         71  71  71  71  71  71 |   402   396   402   405   404   402
pulay+kerker  301 301 301 301 301 301 |  2002  1858  2010  1960  1971  1972
-------------------------------------------
h2o_setup.m    --- Water molecule
mixtype             iteration numbers            convergence time
No Mix        301 301 301 301 301 301 |  1030  1068  1049  1067  1050  1067
anderson       13  13  13  13  13  13 |    63    62    62    63    61    61
broyden        15  14  15  15  15  15 |    67    65    68    69    68    67
pulay         301 301 301 301 301 301 |   367   360   493   503  1179   436
kerker         71  71  71  71  71  71 |   226   233   228   225   228   227
pulay+kerker  119 119 119 119 119 119 |   459   454   476   470   463   471
-------------------------------------------
hnco_setup.m   --- Isocyanic acid molecule
mixtype             iteration numbers            convergence time
```

```
No Mix        301 301 301 301 301 301 |  1859  1828  1825  1893  1887  1827
anderson       16  16  16  16  16  16 |   138   139   135   138   139   137
broyden        17  17  17  16  17  17 |   144   143   142   136   144   139
pulay         301 301 301 301 301 301 |   983   966  1986   948   950   979
kerker         72  72  72  72  72  72 |   446   447   419   439   445   415
pulay+kerker  301 301 301 301 301 301 |  1988  1982  2013  1960  2046  1996
-------------------------------------------
c2h6_setup.m   --- Ethane molecule
mixtype             iteration numbers            convergence time
No Mix        301 301 301 301 301 301 |  1724  1759  1785  1766  1770  1758
anderson       14  14  13  14  14  14 |   101    98    97   100   101   101
broyden        15  15  15  15  15  15 |   108   108   108   108   109   108
pulay         301 301 301 301 301 301 |   807   828   813   799   818   770
kerker         72  72  72  72  72  72 |   359   375   358   374   375   374
pulay+kerker   35  35  35  35  35  35 |   223   222   213   223   226   225
-------------------------------------------
```

Each material gives a different H matrix from (1.29), and thus produces a different fixed-point problem. In the test problem "qdot_setup.m", the convergence rate is slow, maybe because the corresponding fixed-point problem is very nonlinear.

In the experiments, all mixing methods will give essentially the same charge density upon convergence. We show all the isosurfaces of the ground-state charge density of the 8 materials in Figure 3.1 – Figure 3.4:

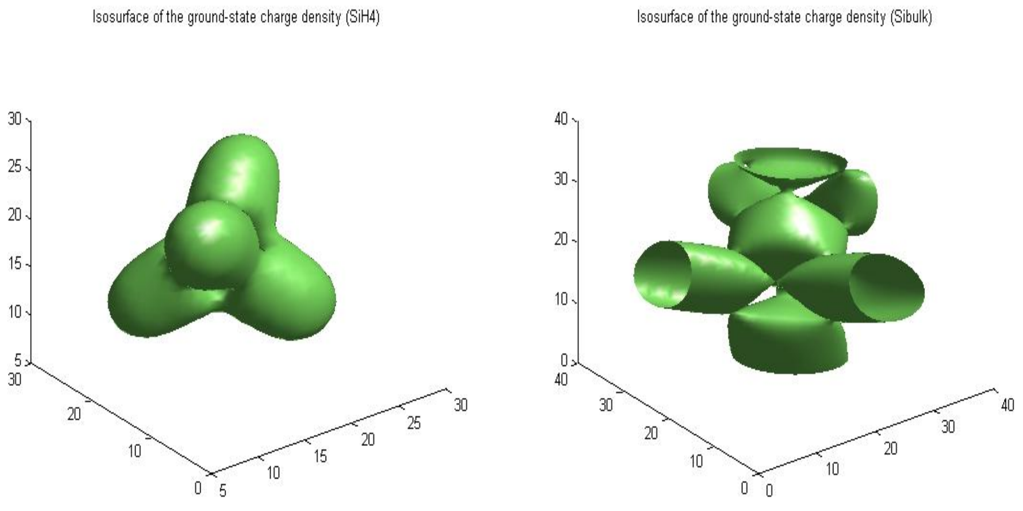
Isosurface of the ground-state charge density (SiH4)
Isosurface of the ground-state charge density (Sibulk)

Figure 3.1: Charge Density for SiH_4 & Silicon bulk

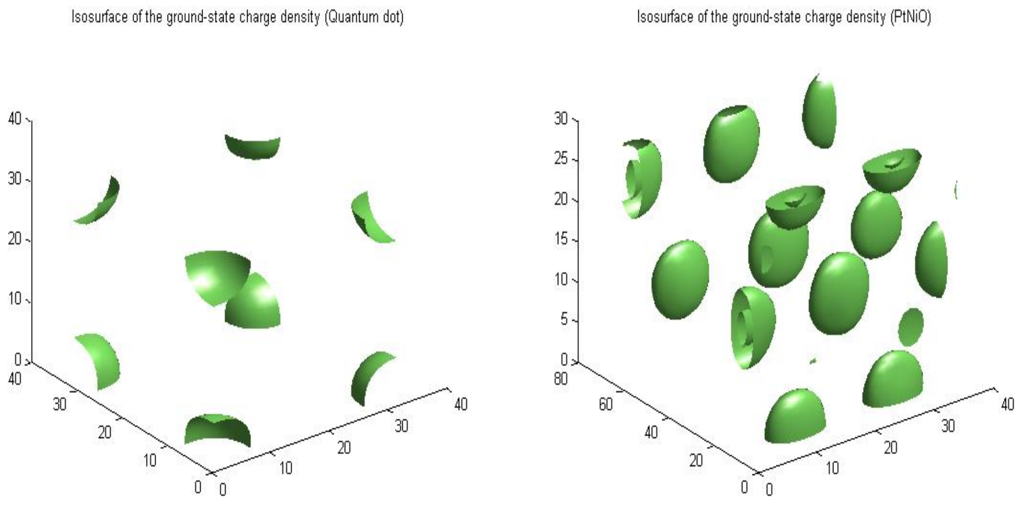
Isosurface of the ground-state charge density (Quantum dot)
Isosurface of the ground-state charge density (PtNiO)

Figure 3.2: Charge Density for Quantum Dot & PtNiO

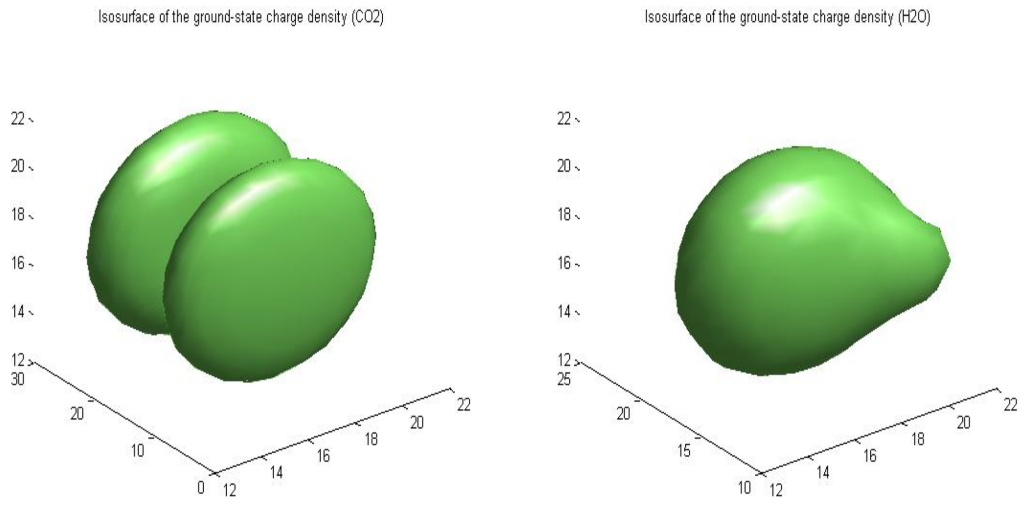
Isosurface of the ground-state charge density (CO2)
Isosurface of the ground-state charge density (H2O)

Figure 3.3: Charge Density for CO_2 & H_2O

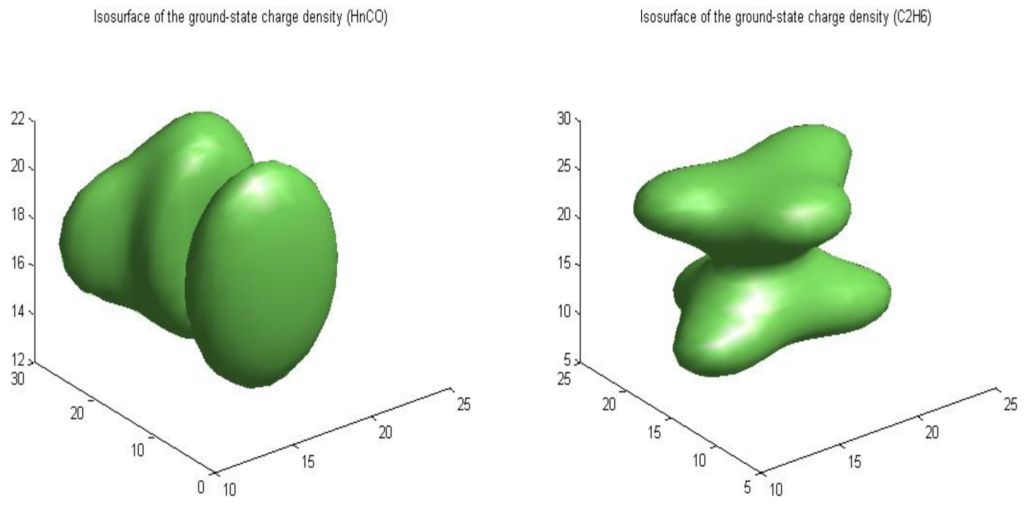
Isosurface of the ground-state charge density (HnCO)
Isosurface of the ground-state charge density (C2H6)

Figure 3.4: Charge Density for HnCO & C_2H_6

We also tested the local-global convergence properties, and our limited experimental results imply that the convergence for each problem is independent of the initial guess, though the initial guess will affect the number of iterations before convergence for particular mixing types.

3.4 Test of Varying Mixdim

For each of the 8 different materials, we vary Mixdim from 0 to 15 with 5 mixing strategies under the same initial guess, where Mixdim = 0 stands for the case of SCF alone. The number of iterations and time spent for each experiment are collected in the following table. In these experiments, we again choose the maximum iteration number to be 300, so the iteration number 301 implies failure of covergence.

We also varied Mixdim up to 30 for some materials under certain mixing strategies, and the results did not carry much more information. Therefore, we used only Mixdim from 0 to 15 to avoid further computational cost.

```
-------------------------------------------
sih4_setup.m   --- Silane molecule
Mixdim           0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
-------------
anderson       252  14  14  12  12  11  11  11  11  11  11  11  11  11  11  11
broyden        252  16  14  14  14  14  14  14  14  14  14  14  14  14  14  14
pulay          252 301 301 301 301 301 301 301 301 301 301 301 301 301 301 301
kerker         252  68  68  68  68  68  68  68  68  68  68  68  68  68  68  68
pulay+kerker   252  63  26  23  23  24  22  21  20  20  20  20  20  20  20  20
-------------------------------------------
sibulk_setup.m --- Silicon bulk (2 Silicon atoms per cell)
Mixdim           0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
-------------
anderson       301  39  18  18  15  14  14  13  13  12  12  12  12  12  12  12
broyden        301  20  20  18  19  17  17  16  16  16  16  16  16  16  16  16
pulay          301 301 301 301 301 301 301 301 301 301 301 301 301 301 301 301
kerker         301  28  28  28  28  28  28  28  28  28  28  28  28  28  28  28
```

```
pulay+kerker  301  26 301 301 301 301 301 301 301 301 301 301 301 301 301 301
-------------------------------------------
qdot_setup.m   --- A four electron quantum dot with an external potential
Mixdim          0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
-------------
anderson      274 206 179 251 301 301 301 301 301 301 301 301 301 301 301 301
broyden       274 261 230 171 211 202 301 301 214 301 301 301 259 237 301 301
pulay         274 301 301 301 301 301 301 301 301 301 301 301 301 301 301 301
kerker        274 256 256 256 256 256 256 256 256 256 256 256 256 256 256 256
pulay+kerker  274 226 209 220 168 229 214 239 234 249 243 228 218 216 267 201
-------------------------------------------
ptnio_setup.m  --- PtNiO molecule
Mixdim          0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
-------------
anderson      301 301 301 301 301 301 130  88  82  66  62  51  47  46  44  44
broyden       301 301 301 301 301 301 301 301 301 301 301 301 301 301 301 301
pulay         301 301 301 301 301 301 301 301 301 301 301 301 301 301 301 301
kerker        301 301 301 301 301 301 301 301 301 301 301 301 301 301 301 301
pulay+kerker  301 301 301 301 301 301 301 301 301 301 301 301 301 301 301 301
-------------------------------------------
co2_setup.m    --- Carbon dioxide molecule
Mixdim          0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
-------------
anderson      301  25  18  16  15  15  14  14  13  13  13  13  13  13  13  13
broyden       301  20  18  16  16  17  15  15  15  16  16  16  16  16  16  16
pulay         301 301 301 301 301 301 301 301 301 301 301 301 301 301 301 301
kerker        301  71  71  71  71  71  71  71  71  71  71  71  71  71  71  71
pulay+kerker  301  57  32 301 301 301 301 301 301 301 301 301 301 301 301 301
-------------------------------------------
h2o_setup.m    --- Water molecule
Mixdim          0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
-------------
anderson      301  17  14  13  13  13  13  13  13  13  13  13  13  13  13  13
broyden       301  18  16  15  15  15  14  14  14  14  14  14  14  14  14  14
```

```
pulay         301 301 301 301 301 301 301 301 301 301 301 301 301 301 301 301
kerker        301  71  71  71  71  71  71  71  71  71  71  71  71  71  71  71
pulay+kerker  301  63 116 121 119 119 114 119 119 115 119 119 115 119 119 119
-------------------------------------------
hnco_setup.m   --- Isocyanic acid molecule
Mixdim          0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
-------------
anderson      301  35  20  18  17  17  16  16  16  16  16  16  16  16  16  16
broyden       301  22  19  18  18  18  18  17  17  17  17  17  17  17  17  17
pulay         301 301 301 301 301 301 301 301 301 301 301 301 301 301 301 301
kerker        301  72  72  72  72  72  72  72  72  72  72  72  72  72  72  72
pulay+kerker  301  62 301 301 301 301 301 301 301 301 301 301 301 301 301 301
-------------------------------------------
c2h6_setup.m   --- Ethane molecule
Mixdim          0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
-------------
anderson      301  23  17  15  14  14  14  14  14  14  14  14  14  14  14  14
broyden       301  19  17  16  16  15  15  15  15  15  15  15  15  15  15  15
pulay         301 301 301 301 301 301 301 301 301 301 301 301 301 301 301 301
kerker        301  72  72  72  72  72  72  72  72  72  72  72  72  72  72  72
pulay+kerker  301  65  37  35  36  35  35  35  35  35  35  35  35  35  35  35
-------------------------------------------
```

Here are plots of log residual norm, that is log $||\delta\rho||$based on the same experimental data as above.

sih4_setup.m — Silane molecule (Figure 3.5 – Figure 3.9)

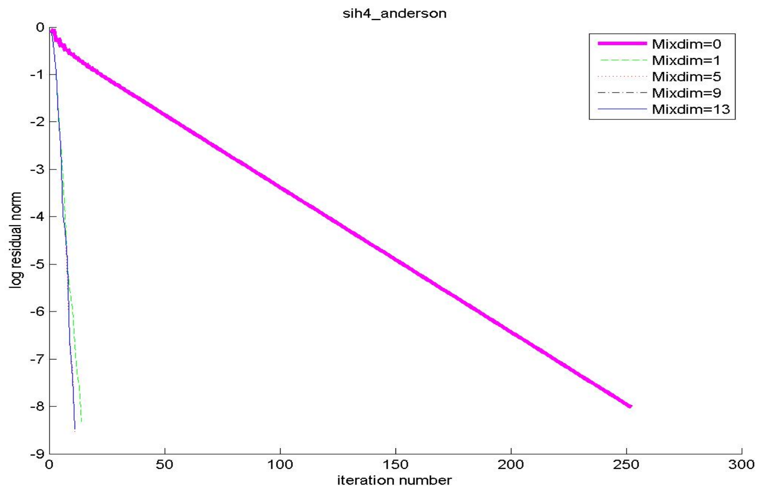

Figure 3.5: Anderson Mixing for SiH_4

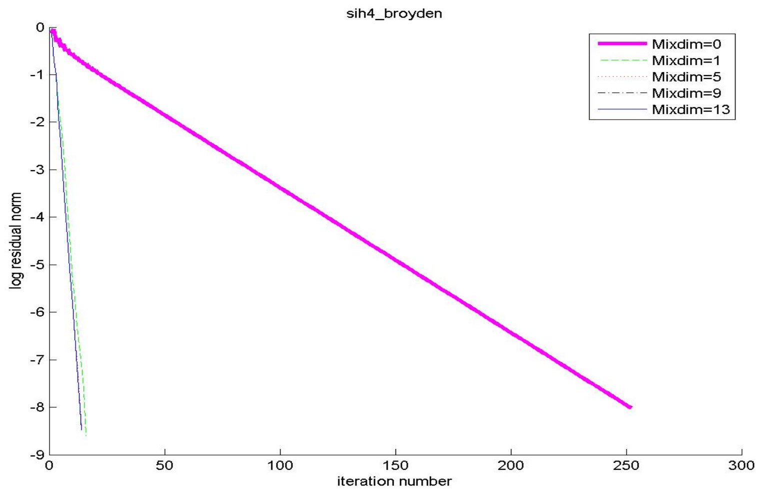

Figure 3.6: Broyden Mixing for SiH_4

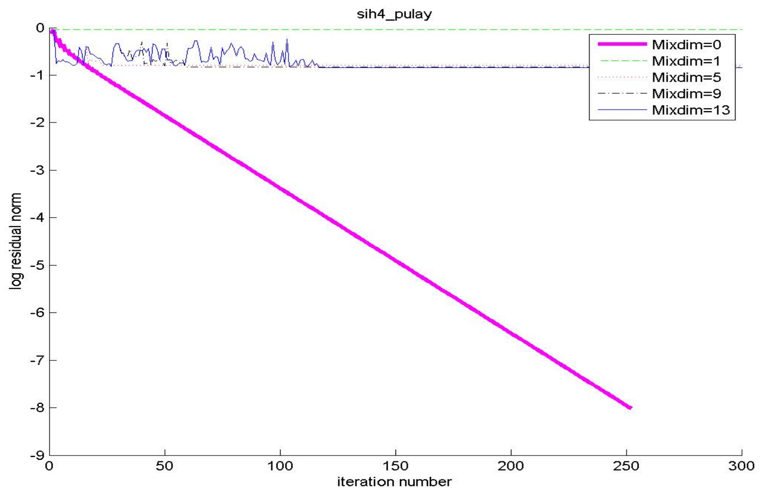

Figure 3.7: Pulay Mixing for SiH_4

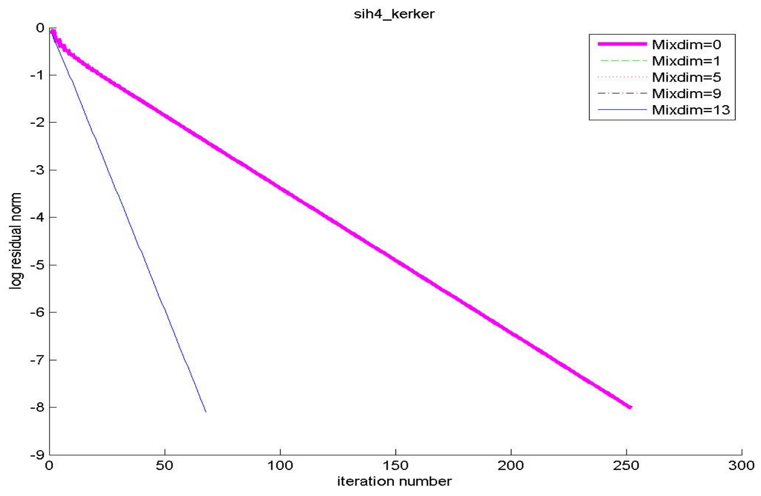

Figure 3.8: Kerker Mixing for SiH_4

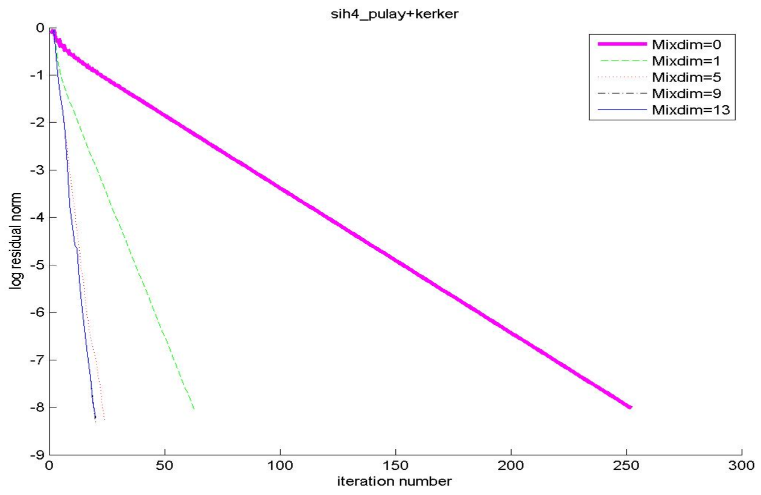

Figure 3.9: Pulay + Kerker Mixing for SiH_4

From Figure 3.5 – Figure 3.9, we see that for SiH_4, the SCF iterates converge linearly, and Anderson and Broyden mixing lead to convergence and have better acceleration effects than other mixing types.

sibulk_setup.m — Silicon bulk (2 Silicon atoms per cell) (Figure 3.10 – Figure 3.14)

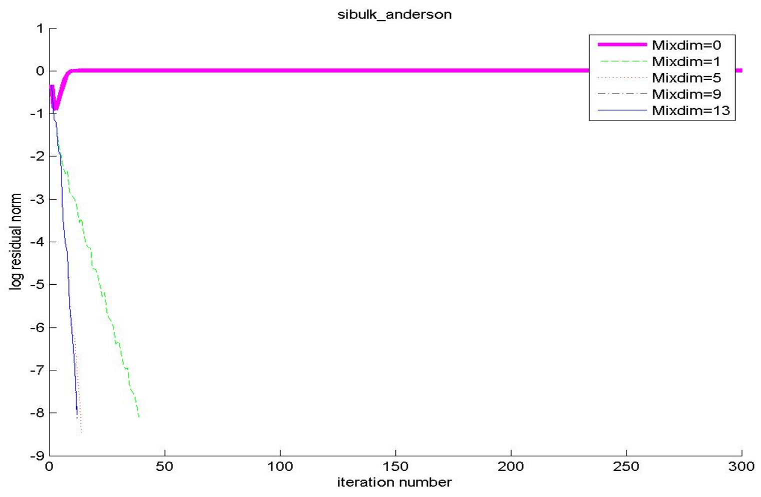

Figure 3.10: Anderson Mixing for Silicon Bulk

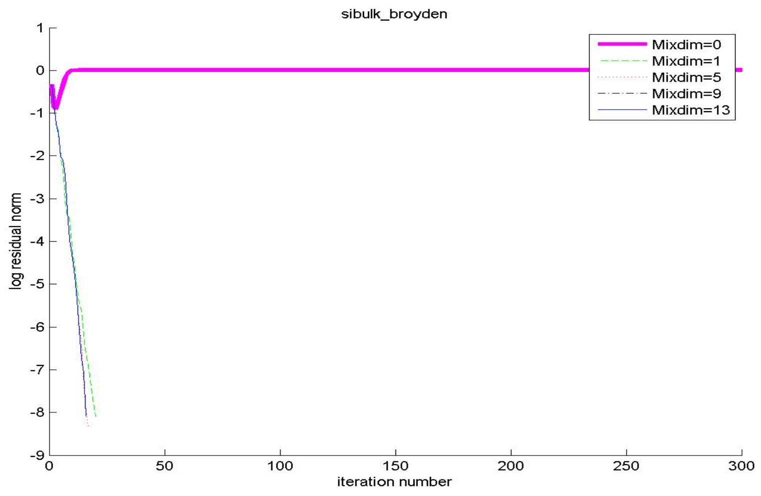

Figure 3.11: Broyden Mixing for Silicon Bulk

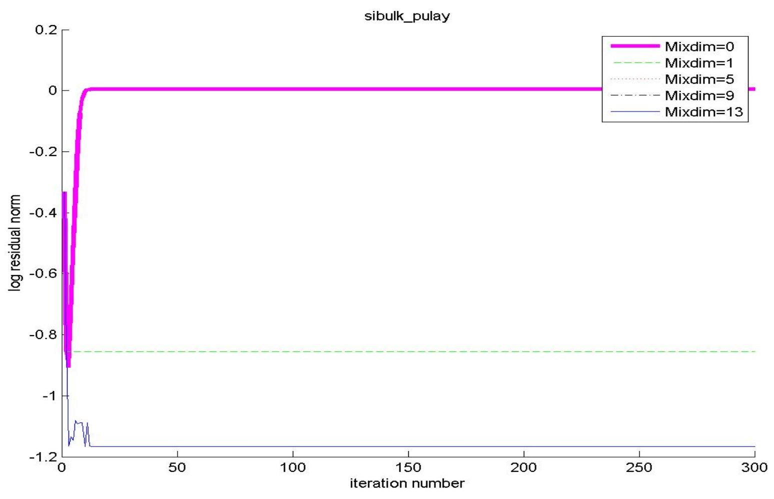

Figure 3.12: Pulay Mixing for Silicon Bulk

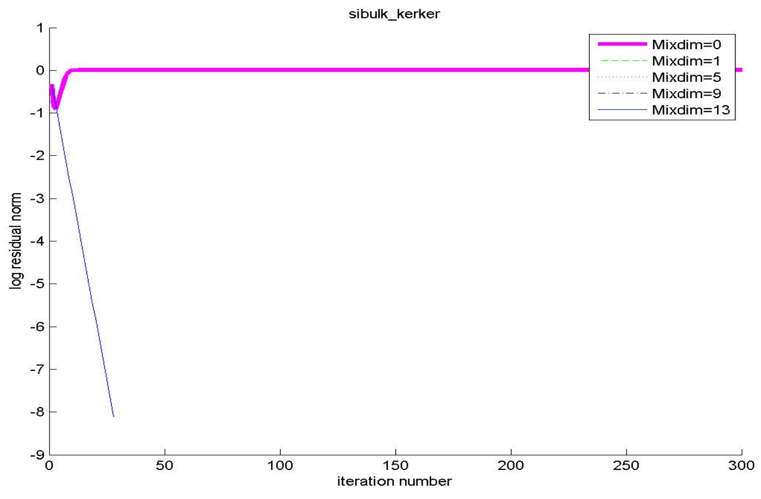

Figure 3.13: Kerker Mixing for Silicon Bulk

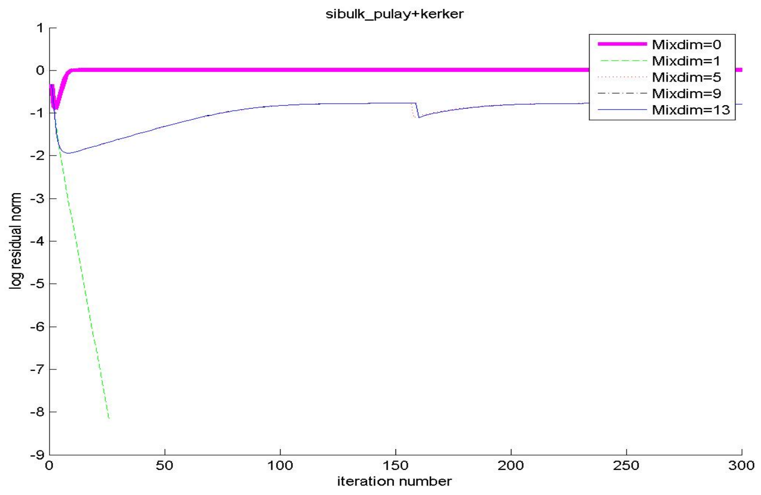

Figure 3.14: Pulay + Kerker Mixing for Silicon Bulk

From Figure 3.10 – Figure 3.14, we see that for silicon bulk, the SCF iterates diverge, and Anderson, Broyden and Kerker mixing give good convergence.

qdot_setup.m — A four electron quantum dot with an external potential (Figure 3.15 – Figure 3.19)

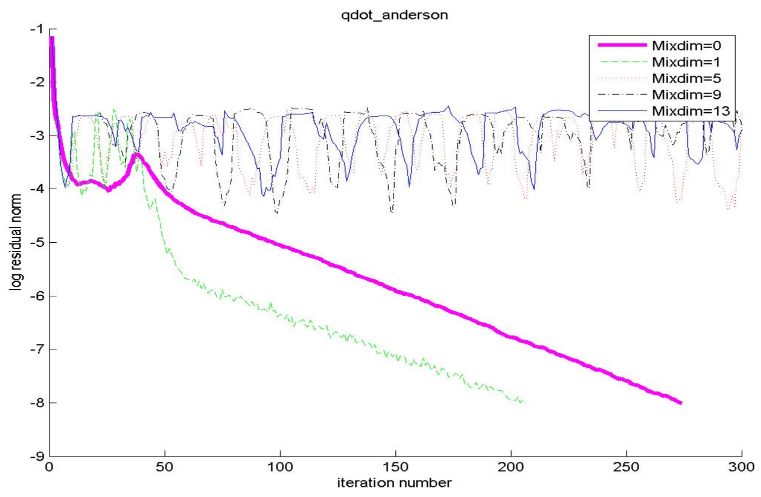

Figure 3.15: Anderson Mixing for Quantum Dot

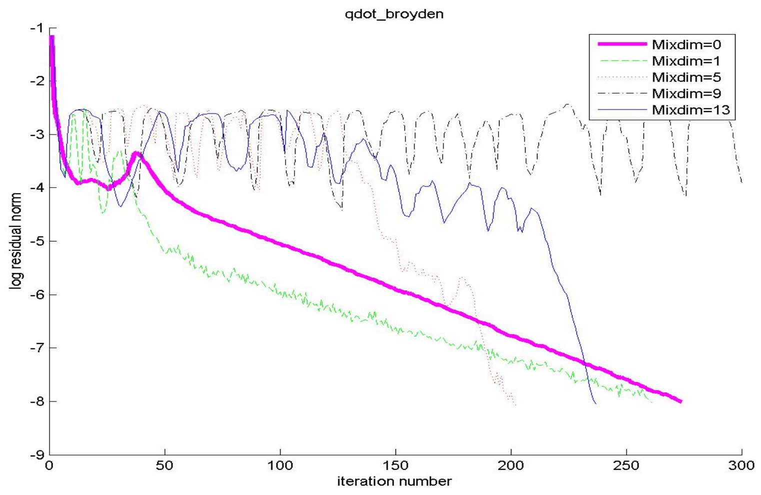

Figure 3.16: Broyden Mixing for Quantum Dot

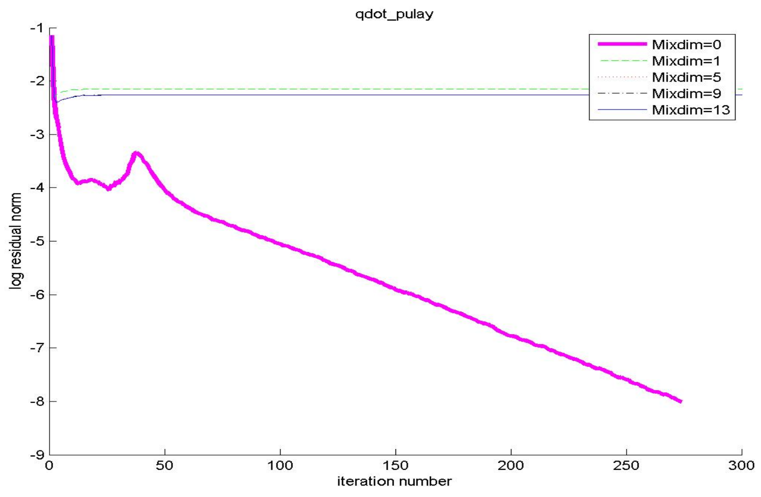

Figure 3.17: Pulay Mixing for Quantum Dot

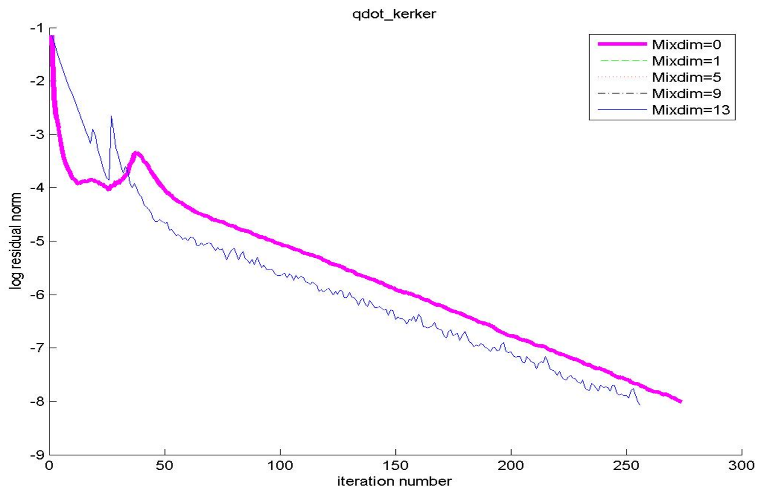

Figure 3.18: Kerker Mixing for Quantum Dot

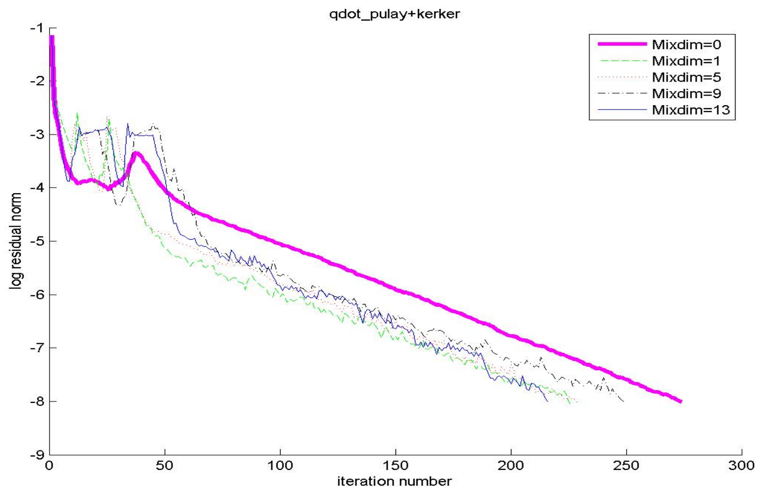

Figure 3.19: Pulay + Kerker Mixing for Quantum Dot

From Figure 3.15 – Figure 3.19, we see that for quantum dot, the SCF method gives linear convergence, and all mixing methods except Pulay mixing lead to convergence, though the improvement is not significant.

ptnio_setup.m — PtNiO molecule (Figure 3.20 – Figure 3.24)

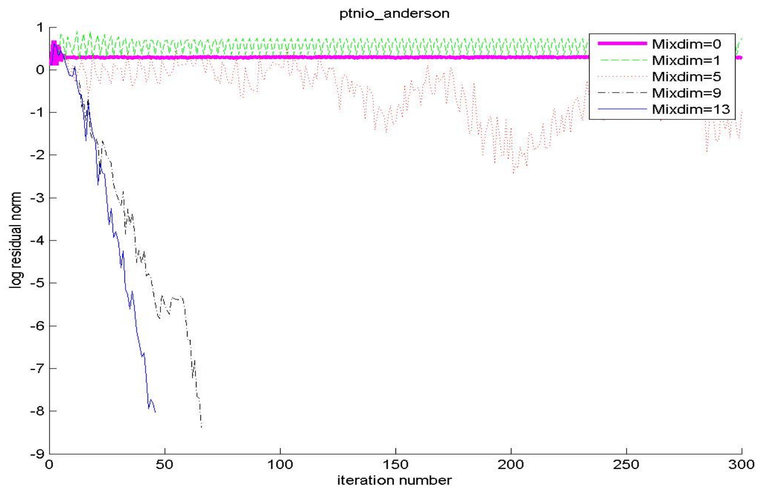

Figure 3.20: Anderson Mixing for PtNiO

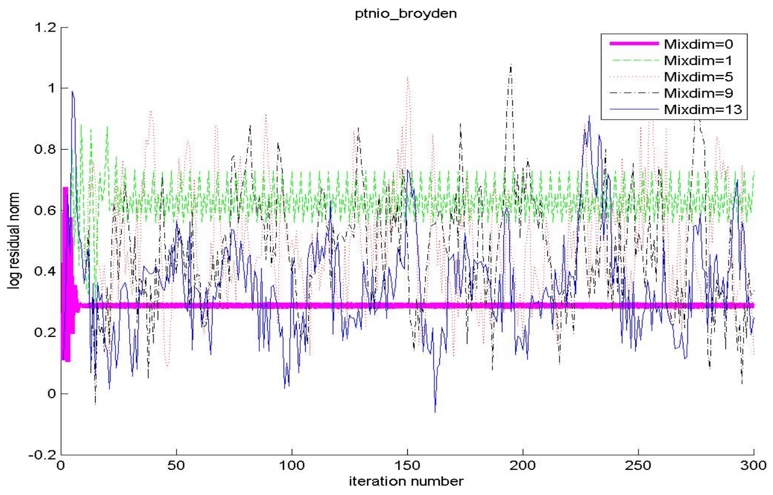

Figure 3.21: Broyden Mixing for PtNiO

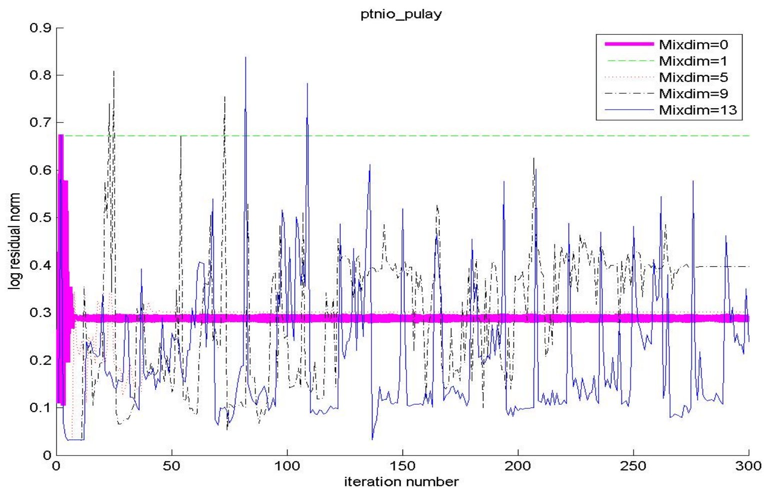

Figure 3.22: Pulay Mixing for PtNiO

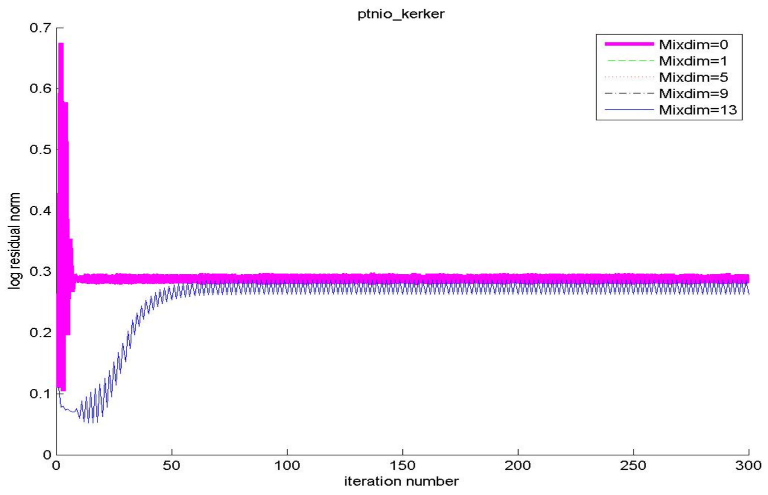

Figure 3.23: Kerker Mixing for PtNiO

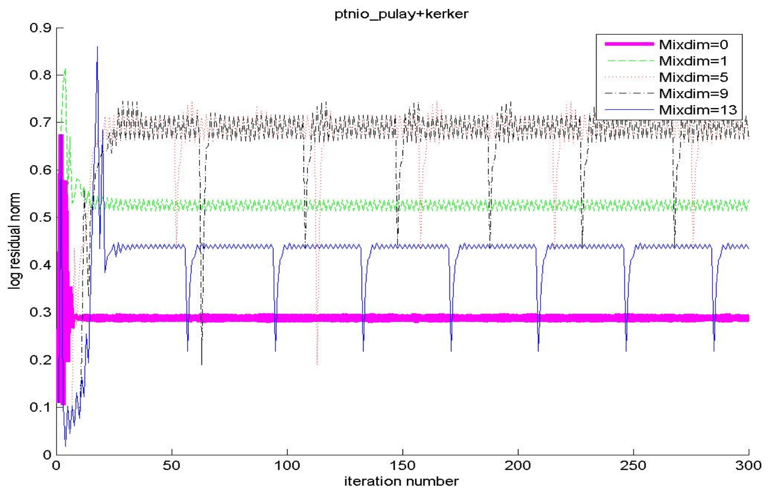

Figure 3.24: Pulay + Kerker Mixing for PtNiO

From Figure 3.20 – Figure 3.24, we can see that for PtNiO, the SCF iterates diverge, while Anderson mixing leads to fast convergence and has the best acceleration effect.

co2_setup.m — Carbon dioxide molecule (Figure 3.25 – Figure 3.29)

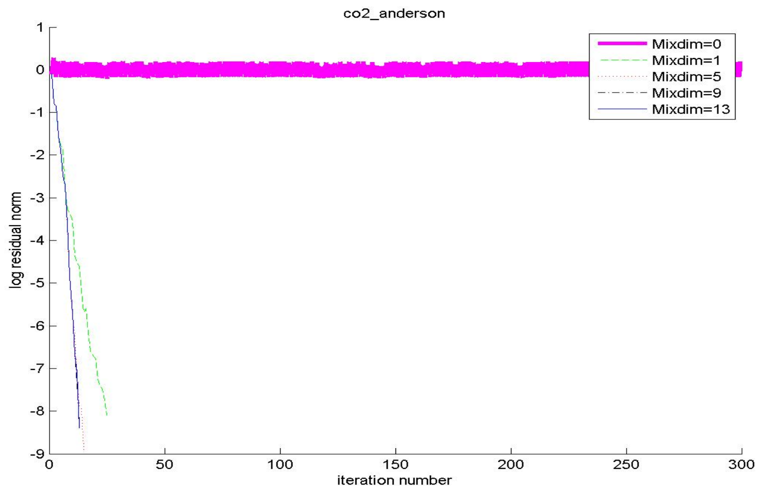

Figure 3.25: Anderson Mixing for CO_2

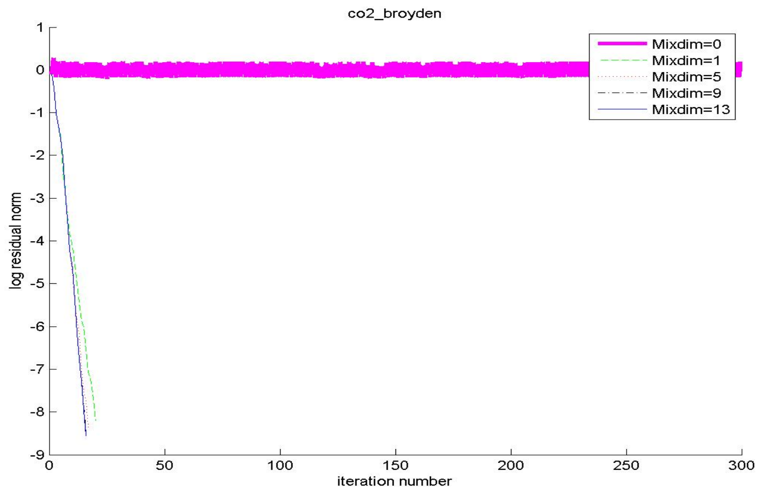

Figure 3.26: Broyden Mixing for CO_2

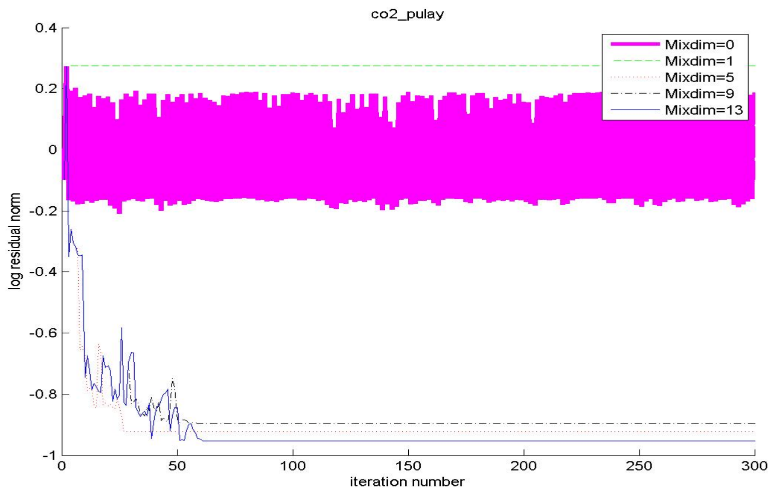

Figure 3.27: Pulay Mixing for CO_2

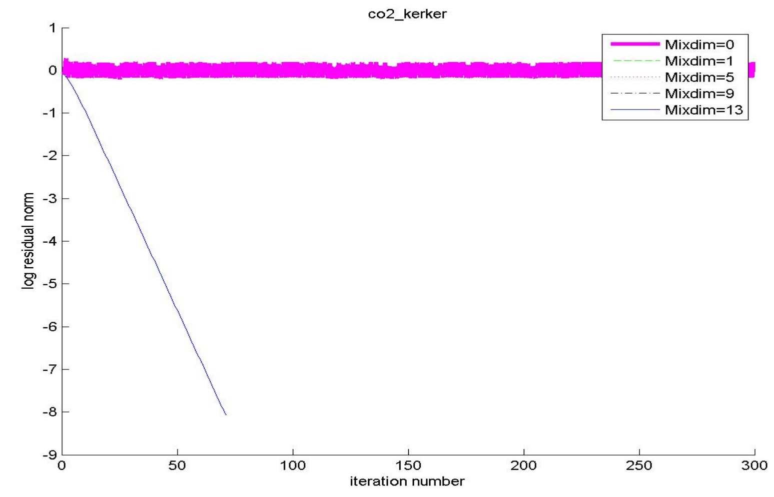

Figure 3.28: Kerker Mixing for CO_2

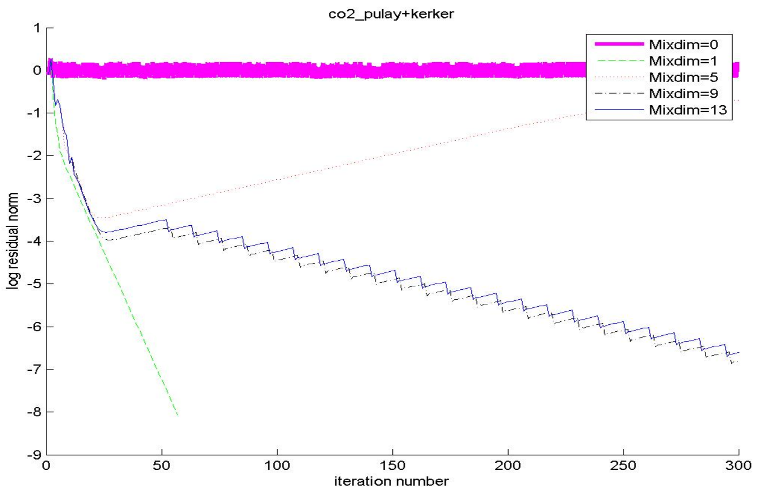

Figure 3.29: Pulay + Kerker Mixing for CO_2

From Figure 3.25 – Figure 3.29, we can see that for CO_2, the SCF iterates diverge, while Anderson and Broyden mixing lead to faster convergence than other methods.

h2o_setup.m — Water molecule (Figure 3.30 – Figure 3.34)

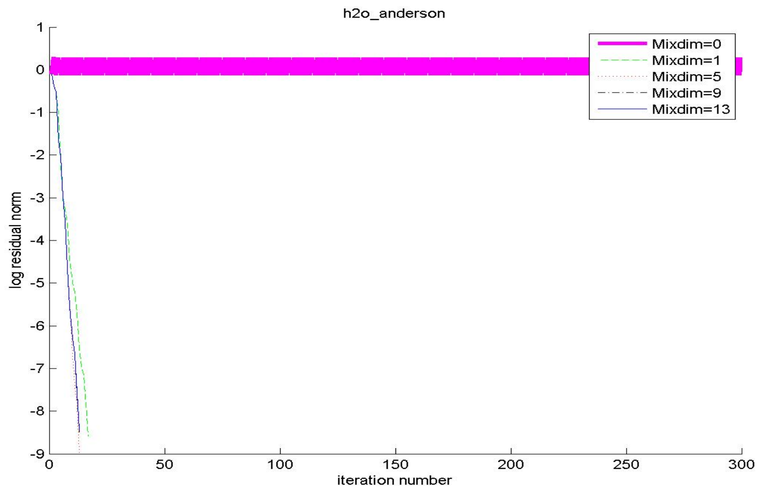

Figure 3.30: Anderson Mixing for H_2O

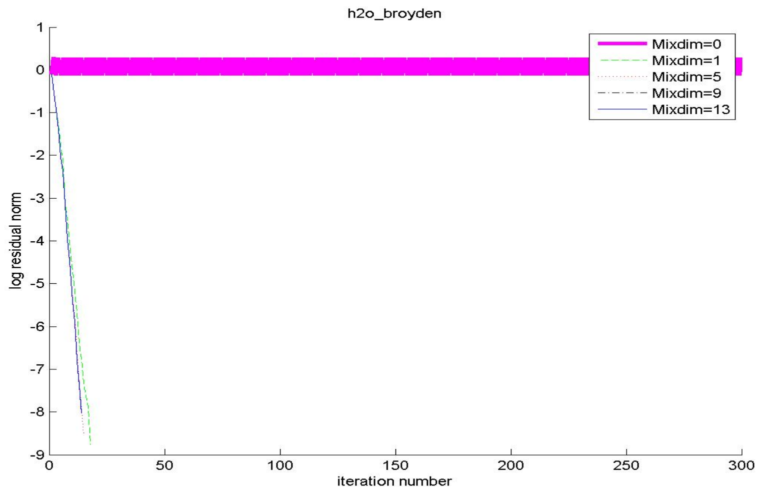

Figure 3.31: Broyden Mixing for H_2O

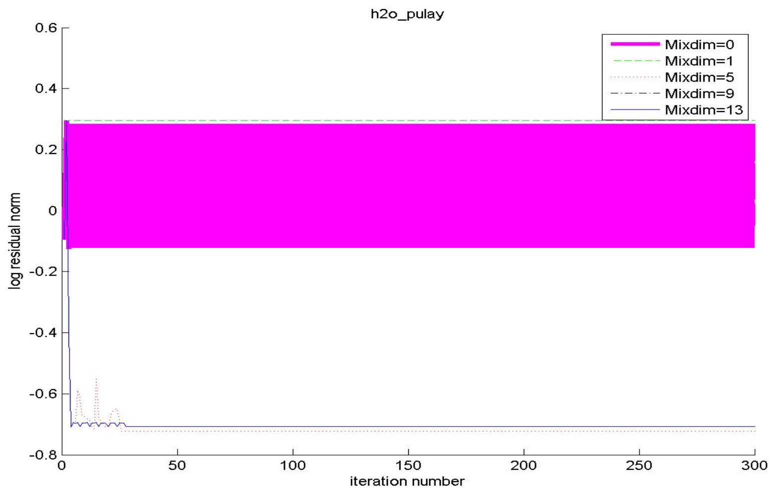

Figure 3.32: Pulay Mixing for H_2O

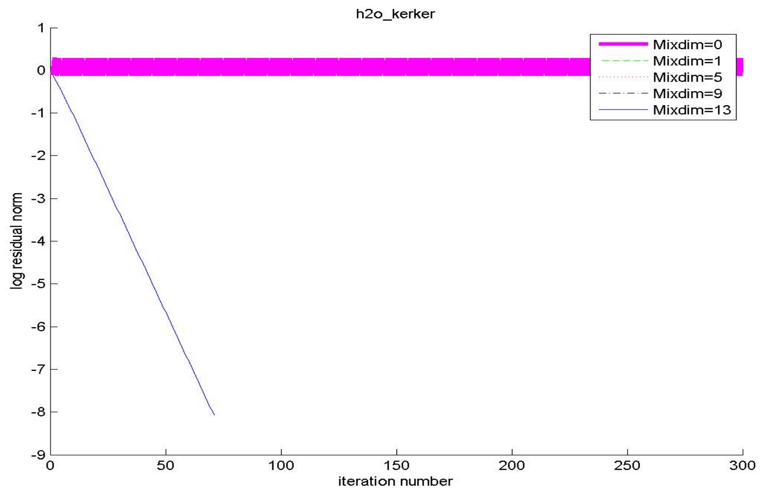

Figure 3.33: Kerker Mixing for H_2O

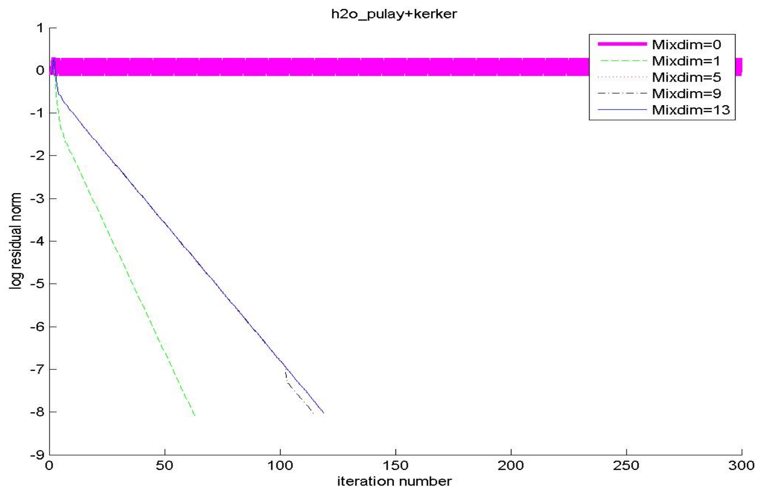

Figure 3.34: Pulay + Kerker Mixing for H_2O

From Figure 3.30 – Figure 3.34, we can see that for H_2O, the SCF iterates diverge, and Anderdon and Broyden mixing lead to faster convergence than other approaches.

hnco_setup.m — Isocyanic acid molecule (Figure 3.35 – Figure 3.39)

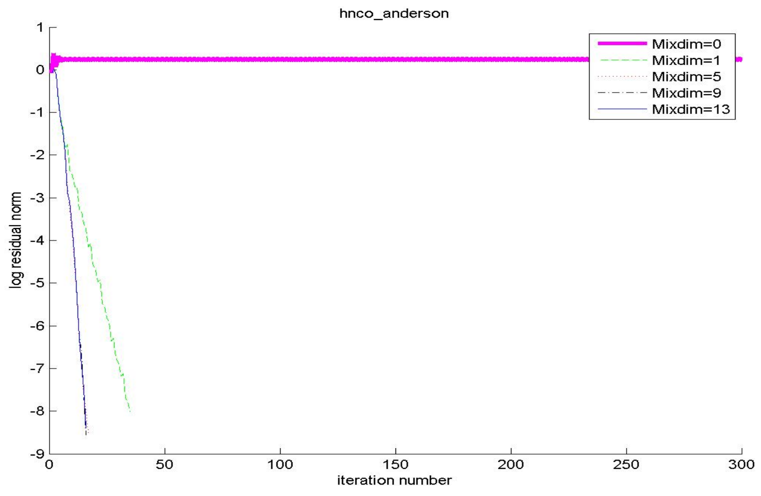

Figure 3.35: Anderson Mixing for HNCO

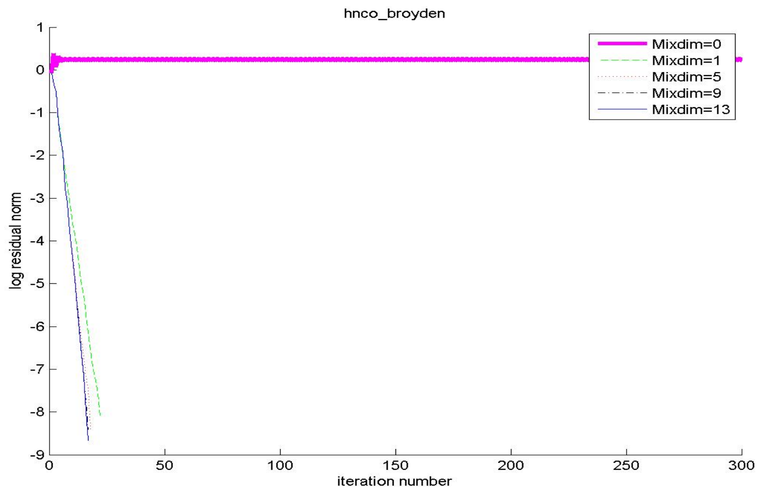

Figure 3.36: Broyden Mixing for HNCO

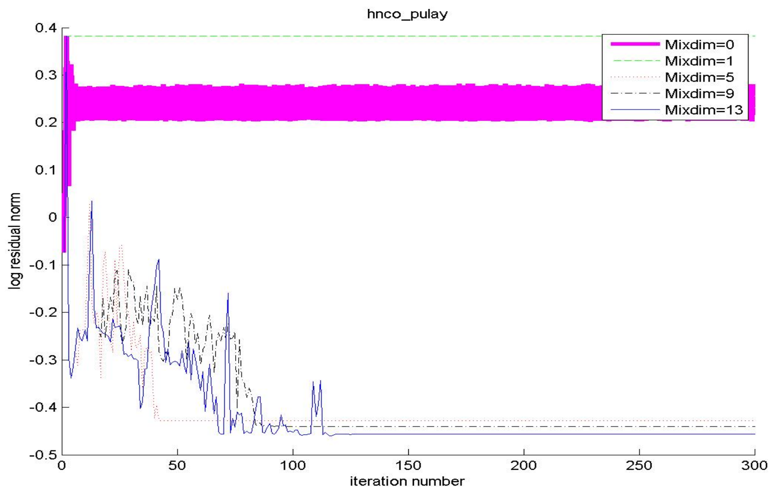

Figure 3.37: Pulay Mixing for HNCO

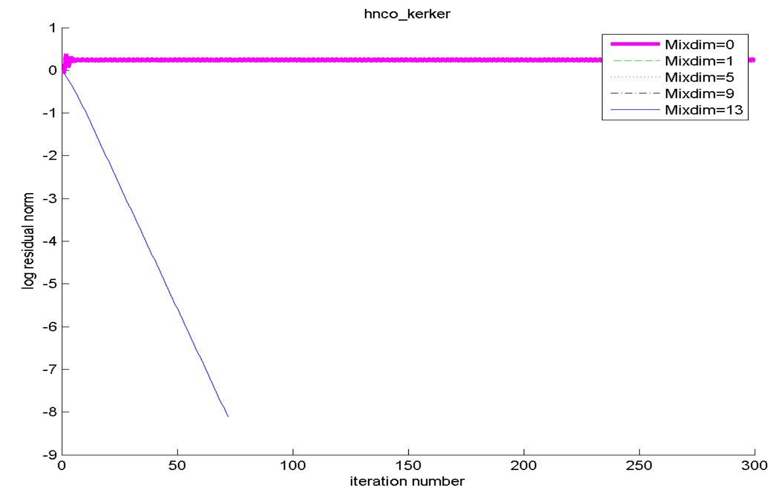

Figure 3.38: Kerker Mixing for HNCO

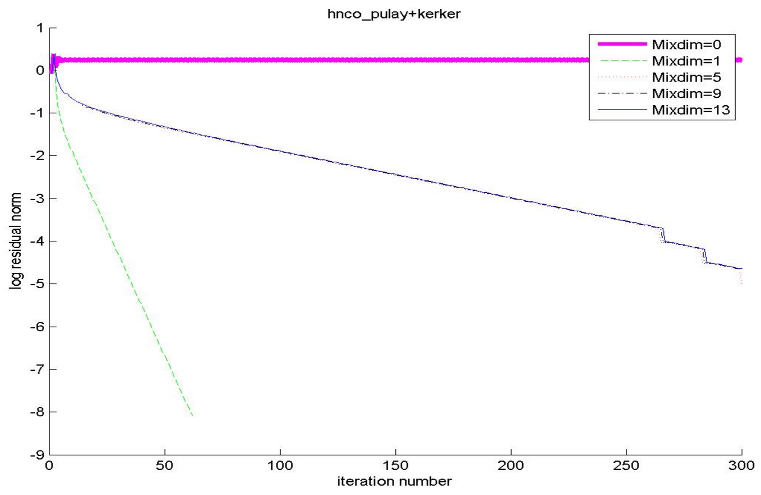

Figure 3.39: Pulay + Kerker Mixing for HNCO

From Figure 3.35 – Figure 3.39, we can see that for HNCO, the SCF iterates diverge, and Anderson and Broyden mixing lead to fast convergence.

c2h6_setup.m — Ethane molecule (Figure 3.40 – Figure 3.44)

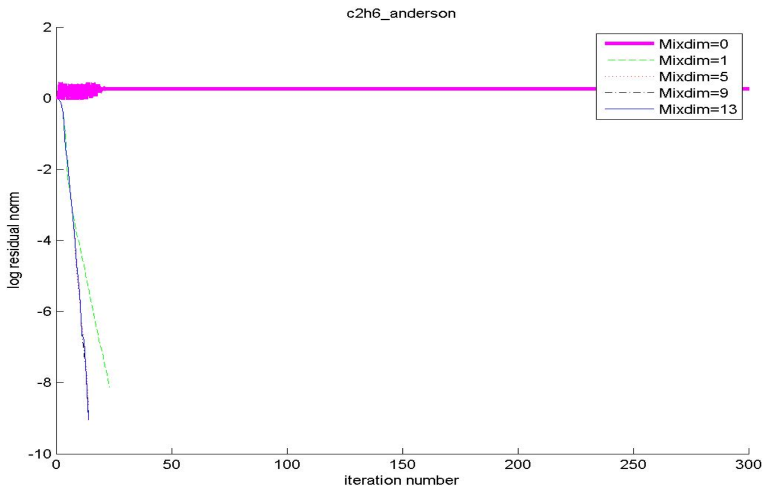

Figure 3.40: Anderson Mixing for C_2H_6

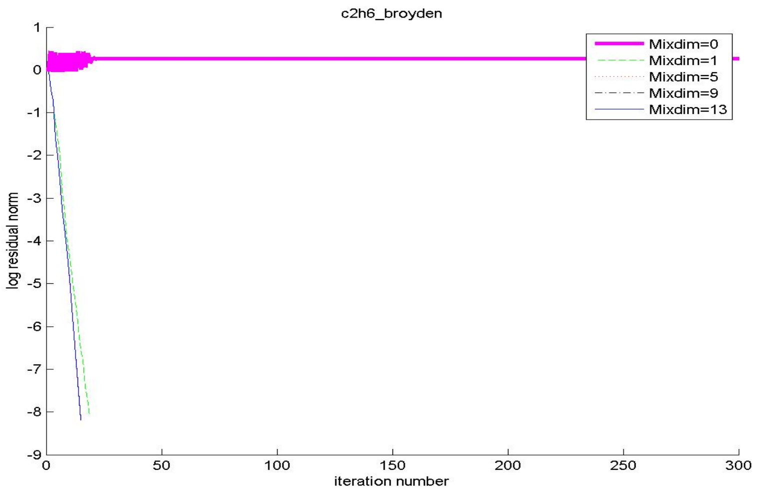

Figure 3.41: Broyden Mixing for C_2H_6

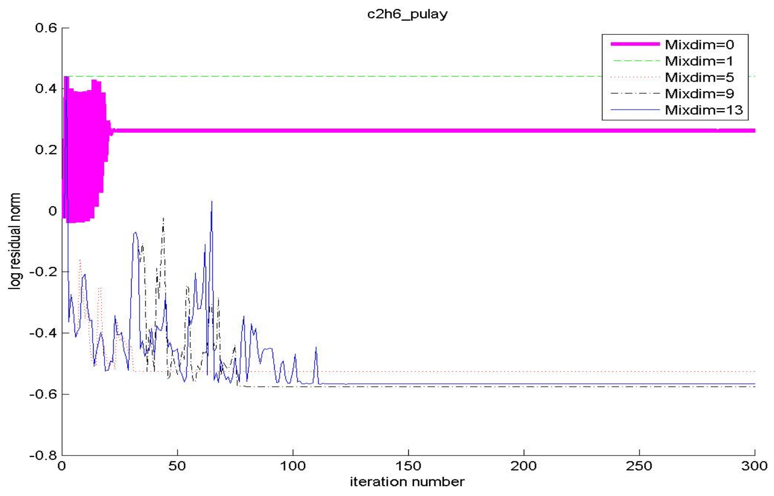

Figure 3.42: Pulay Mixing for C_2H_6

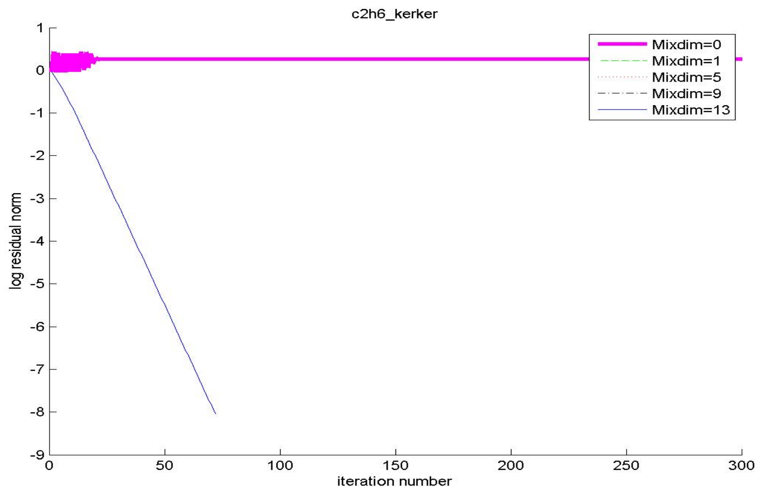

Figure 3.43: Kerker Mixing for C_2H_6

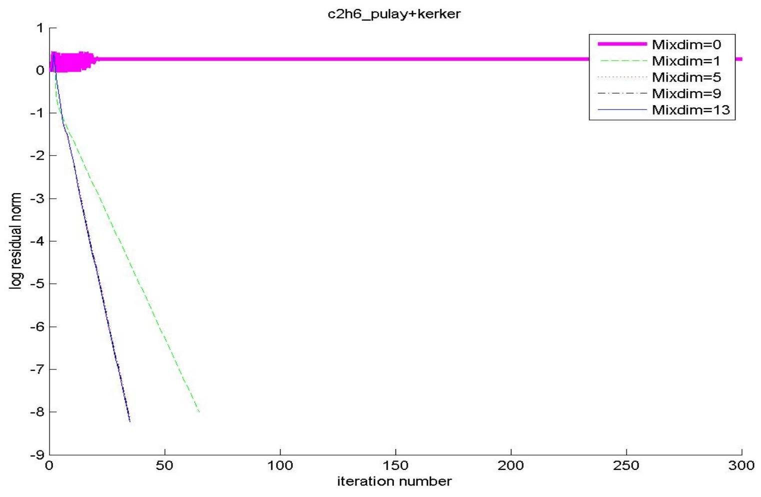

Figure 3.44: Pulay + Kerker Mixing for C_2H_6

From Figure 3.40 – Figure 3.44, we can see that for C_2H_6, the SCF iterates diverge, while Anderson and Broyden mixing give good acceleration effects and lead to fast convergence.

3.5 Least-squares problem

We modify the least-square part of KSSOLV by including the four methods with QR updates mentioned in Section 2.2, and use this code to experiment on the maximum condition numbers and the time expense under different Mixdim's. We did not include the column-dropping strategy in this code. Following are the experimental data.

```
                      Compare Maxcond
------------------------------------------------------------------
sih4_setup.m   --- Silane molecule
Mixdim                      5         10         15         20
---------------------
Lagrange Multipliers :  4.35e+011  1.48e+014  1.48e+014  1.48e+014
Matrix Calculation   :  8.84e+007  5.72e+016  5.72e+016  5.72e+016
Null-Space Method    :  9.88e+002  1.17e+007  1.17e+007  1.17e+007
Method of Elimination:  9.21e+002  9.39e+006  9.39e+006  9.39e+006
------------------------------------------------------------------
sibulk_setup.m --- Silicon bulk (2 Silicon atoms per cell)
Mixdim                      5         10         15         20
---------------------
Lagrange Multipliers :  1.41e+011  1.56e+012  5.64e+012  5.64e+012
Matrix Calculation   :  6.39e+006  1.15e+013  1.98e+015  1.98e+015
Null-Space Method    :  4.86e+002  1.17e+006  2.24e+006  2.24e+006
Method of Elimination:  4.94e+002  8.15e+005  1.90e+006  1.90e+006
------------------------------------------------------------------
qdot_setup.m   --- A four electron quantum dot with an external potential
Mixdim                      5         10         15         20
---------------------
Lagrange Multipliers :  3.06e+005  4.18e+005  1.62e+006  1.87e+007
Matrix Calculation   :  1.19e+005  4.30e+005  1.23e+007  3.94e+008
Null-Space Method    :  1.17e+002  6.03e+002  1.22e+003  4.32e+003
Method of Elimination:  6.76e+001  4.01e+002  1.13e+003  5.67e+003
------------------------------------------------------------------
ptnio_setup.m  --- PtNiO molecule
Mixdim                      5         10         15         20
```

```
---------------------
Lagrange Multipliers :  1.57e+012  6.18e+012  4.10e+013  4.81e+013
Matrix Calculation   :  3.90e+004  1.08e+006  2.36e+010  3.11e+014
Null-Space Method    :  1.90e+001  6.18e+002  7.30e+004  6.65e+006
Method of Elimination:  3.13e+001  8.10e+002  1.08e+005  7.49e+006
------------------------------------------------------------------
co2_setup.m    --- Carbon dioxide molecule
Mixdim                      5         10         15         20
---------------------
Lagrange Multipliers :  7.18e+010  1.11e+013  7.75e+016  7.75e+016
Matrix Calculation   :  2.44e+007  2.44e+014  1.67e+018  1.67e+018
Null-Space Method    :  1.83e+003  3.30e+006  2.66e+008  2.66e+008
Method of Elimination:  1.84e+003  3.64e+006  2.18e+008  2.18e+008
------------------------------------------------------------------
h2o_setup.m    --- Water molecule
Mixdim                      5         10         15         20
---------------------
Lagrange Multipliers :  1.44e+011  4.70e+014  2.61e+016  2.61e+016
Matrix Calculation   :  9.72e+007  1.56e+016  6.21e+017  6.21e+017
Null-Space Method    :  1.19e+003  2.04e+007  1.55e+008  1.55e+008
Method of Elimination:  1.01e+003  1.51e+007  1.38e+008  1.38e+008
------------------------------------------------------------------
hnco_setup.m   --- Isocyanic acid molecule
Mixdim                      5         10         15         20
---------------------
Lagrange Multipliers :  6.51e+011  5.65e+011  4.18e+015  4.18e+015
Matrix Calculation   :  3.88e+005  1.69e+013  1.03e+017  1.03e+017
Null-Space Method    :  1.80e+002  7.30e+005  6.24e+007  6.24e+007
Method of Elimination:  1.71e+002  7.18e+005  7.09e+007  7.09e+007
------------------------------------------------------------------
c2h6_setup.m   --- Ethane molecule
Mixdim                      5         10         15         20
---------------------
Lagrange Multipliers :  9.62e+011  3.35e+013  1.29e+017  1.29e+017
```

```
Matrix Calculation   :  8.22e+006  1.37e+015  3.01e+018  3.01e+018
Null-Space Method    :  8.22e+002  5.45e+006  3.47e+008  3.47e+008
Method of Elimination:  6.84e+002  4.18e+006  3.17e+008  3.17e+008
-------------------------------------------------------------------

                      Compare Timetot
-------------------------------------------------------------------
sih4_setup.m   --- Silane molecule
Mixdim                      5         10         15         20
---------------------
Lagrange Multipliers :  1.77e+000  1.97e+000  2.69e+000  2.48e+000
Matrix Calculation   :  1.49e+000  1.92e+000  2.62e+000  1.88e+000
Null-Space Method    :  1.47e+000  2.53e+000  2.73e+000  2.15e+000
Method of Elimination:  1.80e+000  2.55e+000  2.50e+000  2.06e+000
-------------------------------------------------------------------
sibulk_setup.m --- Silicon bulk (2 Silicon atoms per cell)
Mixdim                      5         10         15         20
---------------------
Lagrange Multipliers :  2.22e+000  3.15e+000  3.21e+000  3.47e+000
Matrix Calculation   :  2.19e+000  3.09e+000  3.43e+000  3.50e+000
Null-Space Method    :  2.32e+000  3.08e+000  3.35e+000  3.75e+000
Method of Elimination:  2.40e+000  3.46e+000  3.95e+000  3.81e+000
-------------------------------------------------------------------
qdot_setup.m   --- A four electron quantum dot with an external potential
Mixdim                      5         10         15         20
---------------------
Lagrange Multipliers :  3.40e-001  4.20e-001  5.50e-001  7.00e-001
Matrix Calculation   :  4.10e-001  4.40e-001  5.30e-001  6.30e-001
Null-Space Method    :  4.00e-001  4.70e-001  5.50e-001  7.10e-001
Method of Elimination:  3.60e-001  4.70e-001  6.30e-001  7.20e-001
-------------------------------------------------------------------
ptnio_setup.m  --- PtNiO molecule
Mixdim                      5         10         15         20
---------------------
Lagrange Multipliers :  1.75e+000  4.42e+001  4.96e+001  5.36e+001
```

```
Matrix Calculation   :  1.88e+000  4.44e+001  4.84e+001  5.36e+001
Null-Space Method    :  1.90e+000  4.42e+001  4.90e+001  4.93e+001
Method of Elimination:  1.98e+000  4.45e+001  4.91e+001  4.91e+001
------------------------------------------------------------------
co2_setup.m    --- Carbon dioxide molecule
Mixdim                      5         10         15         20
---------------------
Lagrange Multipliers :  4.16e+000  3.33e+000  2.89e+000  3.30e+000
Matrix Calculation   :  4.17e+000  3.04e+000  3.29e+000  3.08e+000
Null-Space Method    :  4.37e+000  3.27e+000  3.27e+000  3.31e+000
Method of Elimination:  4.57e+000  3.02e+000  3.14e+000  3.36e+000
------------------------------------------------------------------
h2o_setup.m    --- Water molecule
Mixdim                      5         10         15         20
---------------------
Lagrange Multipliers :  2.40e+000  2.33e+000  2.25e+000  2.16e+000
Matrix Calculation   :  2.46e+000  2.26e+000  2.06e+000  2.26e+000
Null-Space Method    :  2.24e+000  2.29e+000  2.17e+000  2.37e+000
Method of Elimination:  2.27e+000  2.33e+000  2.24e+000  2.28e+000
------------------------------------------------------------------
hnco_setup.m   --- Isocyanic acid molecule
Mixdim                      5         10         15         20
---------------------
Lagrange Multipliers :  3.39e+000  4.36e+000  3.64e+000  4.03e+000
Matrix Calculation   :  3.64e+000  4.27e+000  3.90e+000  3.87e+000
Null-Space Method    :  3.45e+000  3.94e+000  3.96e+000  4.18e+000
Method of Elimination:  3.42e+000  4.22e+000  4.13e+000  4.25e+000
------------------------------------------------------------------
c2h6_setup.m   --- Ethane molecule
Mixdim                      5         10         15         20
---------------------
Lagrange Multipliers :  3.45e+000  3.26e+000  3.43e+000  3.32e+000
Matrix Calculation   :  3.82e+000  3.34e+000  3.60e+000  3.33e+000
Null-Space Method    :  3.64e+000  3.04e+000  3.54e+000  3.68e+000
```

```
Method of Elimination:  3.50e+000  3.10e+000  3.33e+000  3.99e+000
------------------------------------------------------------------
```

Chapter 4

Conclusions

4.1 Summary

In this dissertation,

- we have considered the SCF method accelerated by the Anderson acceleration method as applied to the nonlinear eigenvalue problem in electronic structure computations;
- for the general Anderson acceleration method, we gave a new result and a complete proof that, when applied to a linear system, the iterates are closely related to those of the GMRES method until convergence;
- in the least-squares problem in Anderson acceleration, the null-space method was developed in this context by us (the specific choice of basis was orginal), and we used QR decomposition as well as QR updates to reduce the condition number and number of operations;
- we also did numerical experiments to study the convergence of the Anderson acceleration method and other mixing methods, and the effects of changing the parameter Mixdim within the Anderson routine.

Through the comparison of the experimental data for the nonlinear eigenvalue problem, the Anderson acceleration has the best performance among mixing methods tested. The Anderson acceleration method has overall better performance when the Mixdim value is around 9 in our experiments.

For the least-squares problem within the Anderson acceleration routine, by our analysis, the null-space method will solve linear systems with the smallest condition number; however, in our experiments, the condition numbers in the null-space method and the method of elimination do not have significant differences. The condition numbers for both of the methods are smaller than the ones in the other two methods. The use of QR decomposition reduces the condition numbers relative to other approaches. Updating the QR factors reduces the number of operations, but in our experiments the run time was not reduced significantly, probably because solving the least-squares problem was a relatively small part of the overall computations.

4.2 Future Research

Acceleration algorithms for fixed-point iterations is a broad topic, and we have only done a very small part of it.

As a continuation of this dissertation work, we may apply the Anderson acceleration method to various types of fixed-point problems and study its performance and convergence properties. Also, we may do more theoretical research on convergence properties of the algorithm applied to some particular nonlinear problems. In addition, we may study vector extrapolation methods, compare them with Anderson acceleration, and draw some conclusions on the convergence properties from the proved properties of the vector extrapolation methods.

Under this subject, there still remains a lot of interesting topics for us to develop in the future.

Bibliography

[1] D. G. Anderson. Iterative procedures for nonlinear integral equations. *J. Assoc. Comput. Mach.*, 12:547–560, 1965.

[2] N. Argaman and G. Makov. Density functional theory: An introduction. *American Journal of Physics*, 68(1):69–79, 2000.

[3] Å. Björck. *Numerical Methods for Least Squares Problems.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1996.

[4] C. Le. Bris. Computational chemistry from the perspective of numerical analysis. *Acta Numerica*, 14:363–444, 2005.

[5] H. Fang and Y. Saad. Two Classes of Multisecant Methods for Nonlinear Acceleration. *Numer. Linear Algebra with Appl. (2008)*, 2008.

[6] G. H. Golub and C. F. Van Loan. *Matrix Computations.* Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, third edition, 1996.

[7] P. Hohenberg and W. Kohn. Inhomogeneous electron gas. *Phys. Rev.*, 136(3B):B864–B871, Nov 1964.

[8] K. Jbilou and H. Sadok. Some results about vector extrapolation methods and related fixed-point iterations. *J. Comput. Appl. Math.*, 36(3):385–398, 1991.

[9] K. Jbilou and H. Sadok. Vector extrapolation methods. Applications and numerical comparison. *J. Comput. Appl. Math.*, 122(1-2):149–165, 2000.

[10] G. P. Kerker. Efficient iteration scheme for self-consistent pseudopotential calculations. *Phys. Rev. B*, 23(6):3082–3084, Mar 1981.

[11] W. Kohn and L. J. Sham. Self-consistent equations including exchange and correlation effects. *Phys. Rev.*, 140(4A):A1133–A1138, Nov 1965.

[12] G. Kresse and J. Furthmüller. Efficiency of ab-initio total energy calculations for metals and semiconductors using a plane-wave basis set. *Computational Materials Science*, 6:15–50, 1996.

[13] H. L. Neal. Density functional theory of one-dimensional two-particle systems. *American Journal of Physics*, 66(6):512–516, 1998.

[14] H. L. Neal. A density functional perspective for single-particle systems. *American Journal of Physics*, 72:605–607, May 2004.

[15] P. Ni and H. Walker. A linearly constrained least-squares problem in electronic structure computations. *Advances in Computational & Experimental Engineering and Sciences*, 7(1):43–50, 2008. Publisher: Tech Science Press. Editors: Murakawa, H., Okada, H., Dowling, J.P., Lee, S.W., Tang, D. and Tewary, V.

[16] P. Pulay. Convergence acceleration of iterative sequences. The case of SCF iteration. *Chem. Phys. Letters*, 73(2):393–398, 1980.

[17] P. Pulay. Improved SCF convergence acceleration. *J. Comput. Chem.*, 3(4):556–560, 1982.

[18] Y. Saad and M. H. Schultz. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.*, 7(3):856–869, 1986.

[19] A. Schindlmayr. Universality of the hohenberg–kohn functional. *American Journal of Physics*, 67(10):933–934, 1999.

[20] A. Sidi. Convergence and stability properties of minimal polynomial and reduced rank extrapolation algorithms. *SIAM J. Numer. Anal.*, 23(1):197–209, 1986.

[21] A. Sidi. Efficient implementation of minimal polynomial and reduced rank extrapolation methods. *J. Comput. Appl. Math.*, 36(3):305–337, 1991.

[22] A. Sidi, W. F. Ford, and D. A. Smith. Acceleration of convergence of vector sequences. *SIAM J. Numer. Anal.*, 23(1):178–196, 1986.

[23] M. Sipics. Unknown Nanostructures Give Up Secrets to Interdisciplinary Group at LBNL. http://www.siam.org/news/news.php?id=1148, 2007.

[24] C. Yang, J. C. Meza, B. Lee, and L. W. Wang. KSSOLV - A MATLAB toolbox for solving the Kohn-Sham equations. *ACM Transactions on Matematical Software*, V(4):1–31, 2008.

[25] C. Yang, J. C. Meza, and L.W. Wang. A trust region direct constrained minimization algorithm for the Kohn-Sham equation. *SIAM J. Sci. Comput.*, 29(5):1854–1875 (electronic), 2007.