Electronic Transactions on Numerical Analysis.
Volume 38, pp. 347-362, 2011.

ISSN 1068-9613.

ETNA
Kent State University
http://etna.math.kent.edu

REDUCED RANK EXTRAPOLATION APPLIED TO ELECTRONIC STRUCTURE COMPUTATIONS*

SÉBASTIEN DUMINIL† AND HASSANE SADOK†

Abstract. This paper presents a new approach for accelerating the convergence of a method for solving a nonlinear eigenvalue problem that arises in electronic structure computations. Specifically, we seek to solve the Schrödinger equation using the Kohn-Sham formulation. This requires the solution of a nonlinear eigenvalue problem. The currently prevailing method for determining an approximate solution is the Self-Consistent Field (SCF) method accelerated by Anderson's iterative procedure or a Broyden-type method. We propose to formulate the nonlinear eigenvalue problem as a nonlinear fixed point problem and to accelerate the convergence of fixed-point iteration by vector extrapolation. We revisit the reduced rank extrapolation method, a polynomial-type vector extrapolation method, and apply it in the RSDFT (real-space density functional theory) software.

Key words. nonlinear eigenvalue problem, vector extrapolation, Kohn-Sham equation, Anderson's method, Broyden's method, reduced rank extrapolation.

AMS subject classifications. 65H10, 65F10

1. Introduction.

The task of determining limits of sequences arises frequently in numerical analysis, applied mathematics, and engineering. The sequences may be produced by iterative methods or perturbation techniques. Many of the encountered sequences converge so slowly that it is difficult to determine their limit with desired accuracy in a straightforward manner. The computation of the limit requires the use of convergence acceleration methods.

We are concerned with electronic structure computations by solving the Kohn-Sham formulation of the Schrödinger equation. The solution can be determined by computing a nonlinear eigenvalue problem. The currently prevailing solution method for the latter is the Self-Consistent Field (SCF) method accelerated by Anderson's method or a Broyden-type method; see, e.g., [6, 16, 25, 26]. The acceleration methods are known as "mixing" methods in the literature on electronic structure computations. We propose to apply extrapolation to solve the nonlinear eigenvalue problem.

This paper is organized as follows. Section 2 introduces the SCF method, the nonlinear eigenvalue problem to be solved, as well as Anderson- and Broyden-type acceleration methods. In Section 3, we introduce polynomial extrapolation methods. Numerical examples are reported in Section 4.

2. Electronic structure computations.

This section uses the same notation as [21]. The electronic structure is described by a wave function ψ which can be obtained by solving the Schrödinger equation

$$H\Psi = E\Psi,$$

where H is the Hamiltonian operator for the system and E is the total energy.

In its original form, the operator H is very complex, involving sums over all electrons and nuclei and the Laplacian related to each nucleus. To render the problem more tractable, we use two fundamental approximations: the Born-Oppenheimer approximation and the one-electron approximation; see [13, 19] for more details on the simplification of the Schrödinger

* Received May 10, 2011. Accepted for publication November 14, 2011. Published online December 7, 2011. Recommended by L. Reichel.

†Laboratoire de Mathématiques Pures et Appliquées, Université du Littoral, Centre Universitaire de la Mi-Voix, Batiment H. Poincarré, 50 Rue F. Buisson, BP 699, 62228 Calais cedex, France ({duminil, sadok}@lmpa.univ-littoral.fr).

equation. With these approximations, we obtain a simplified form of the Schrödinger equation known as the Kohn-Sham equation

$$H(\rho(r))\psi(r) = \left[\frac{-\hbar\nabla^2}{2m} + V_{tot}\left[\rho(r), r\right]\right]\psi(r) = E\psi(r), \tag{2.1}$$

where the Laplacian ∇^2 represents the kinetic operator, $\hbar$ is Planck's constant, m is the electron mass, and V_{tot} is the total potential at some point r in space. The potential depends on the charge density ρ defined below. We note that V_{tot} is the sum of three components: the ionic potential which reflects the energy from the core electrons, the Hartree potential which reflects electron-electron Coulombic energies, and the Exchange-Correlation potential which arises from the one-electron approximation:

$$V_{tot} = V_{ion} + V_H + V_{XC}. \tag{2.2}$$

Both terms V_H and V_{XC} are functions of the charge density $\rho(r)$, which depends on the wave functions of the above equation,

$$\rho(r) = \sum_{\text{occupied states}} |\psi_i(r)|^2 . \tag{2.3}$$

The exchange-correlation potential V_{XC} is easily approximated by a potential from the local density expression. Once the charge density $\rho(r)$ is known, the Hartree potential is obtained by solving the Poisson equation

$$\nabla^2 V_H = -4\pi\rho(r).$$

Both potentials V_H and V_{XC} have a local character and are represented by diagonal matrices in the discrete form of the problem. The ionic potential is more complex; see [20] for details. It follows from (2.3) that ρ can be computed from the wave functions ψ_i for all occupied states. The wave functions ψ_i are solutions of the eigenvalue problem (2.1), whose coefficients depend on the potential. The central computational task is the repeated solution of a large, symmetric eigenvalue problem. This problem can be regarded as a nonlinear eigenvalue problem, where the nonlinearity is handled by the SCF iteration. An algorithm for solving the nonlinear eigenvalue problem using simplemix iterations is described in Table 2.1. The quantity ω used in the algorithm is a given constant. Our aim is to explore the possibility of accelerating the simplemix iterations by using the Reduced Rank Extrapolation (RRE) method.

In matrix form, the Hamiltonian is the sum of a Laplacian matrix, three diagonal matrices, and a matrix representing the nonlocal contributions. Two of the diagonal matrices arise from discretizing V_{XC} and V_H. The third one is due to the local part of the ionic potential. The nonlocal matrix is the sum of simple rank-one updates over all atoms and quantum numbers.

The SCF iteration can be expressed as fixed-point iteration for the equation

$$s = \Phi(s), \tag{2.4}$$

where $\Phi(s)$ represents the new potential obtained by solving (2.1) and using (2.2) with the resulting wave functions and values, and $s = V_{tot}$. We remark that the SCF iterations may be less effective than desired: the convergence may be slow, and sometimes the iterates diverge. In the following, we focus on accelerating the SCF iterations. We will replace the constant ω in Table 2.1 by a so-called "mixing term", which refers to an acceleration by a quasi-Newton

ETNA
Kent State University
http://etna.math.kent.edu

TABLE 2.1
SCF iteration algorithm.

Initial approximate charge density $\rho(r)$.
Evaluate $V_{tot}(\rho(r), r)$.
For iter=$1, 2, \ldots,$ Maxiter do
  Evaluate $H(\rho(r))$ by $H(\rho(r)) = \left[\frac{-\hbar\nabla^2}{2m} + V_{tot}\left(\rho(r), r\right)\right]$.
  Solve $H(\rho(r))\psi_i(r) = E\psi_i(r)$ for $i = 1, 2, \ldots$
  Compute new charge density $\rho(r)_{new}$ by (2.3).
  Solve new Hartree potential V_H from $\nabla^2 V_H(r) = -4\pi\rho(r)$.
  Update V_{XC} and V_{ion}.
  Evaluate $\tilde{V}_{tot}(\rho(r), r) = V_{ion}(r) + V_H(\rho(r), r) + V_{XC}(\rho(r), r)$.
  Compute $\tilde{V}_{tot} = V_{tot} + \omega(\tilde{V}_{tot} - V_{tot})$.
  If $\frac{\|\tilde{V}_{tot} - V_{tot}\|}{\|\tilde{V}_{tot}\|} <$ tol stop.
  Update $V_{tot} = \tilde{V}_{tot}$.
end for

method. Broyden-type methods and Anderson's method are used in [6, 16] to improve the convergence. Fang and Saad [6] report that Broyden's method sometimes gives the fastest convergence; in other experiments Anderson's method is superior. Throughout this paper $\|\cdot\|$ denotes the Euclidean vector norm.

2.1. Generalized Broyden's methods. Fang and Saad developed Generalized Broyden's methods in [6]. Determining a fixed point of (2.4) is equivalent to solving the nonlinear system of equations

$$f(s) = 0, \tag{2.5}$$

where $f(s) = s - \Phi(s)$.

2.1.1. Broyden's methods. Introduce the linear approximation of f at s,

$$f(s + \Delta s) \approx f(s) + J(s)\Delta s,$$

where $J(s)$ denotes the Jacobian matrix of f at s. Newton's method uses this approximation to determine a correction Δs_k of an available approximate solution s_k of (2.5). Specifically, Δs_k is determined by

$$\begin{aligned} J(s_k)\Delta s_k &= -f(s_k), \\ s_{k+1} &= s_k + \Delta s_k, \end{aligned} \tag{2.6}$$

for $k = 1, 2, 3, \ldots$ The iterations are carried out until a sufficiently accurate approximate solution has been determined, assuming that Newton's method converges.

Newton's method requires the Jacobian matrix to be computed at each iteration. This can be expensive. Quasi-Newton methods approximate $J(s_k)$ by a matrix J_k that is cheaper to compute. Broyden [3, 14] proposed to let J_{k+1} be a rank-one modification of J_k. Specifically, Broyden suggested that equation (2.6) be replaced by

$$J_{k+1}\Delta s_k = \Delta f_k,$$

where $\Delta f_k = f(s_{k+1}) - f(s_k)$ and

$$J_{k+1} = J_k + (\Delta f_k - J_k\Delta s_k)\,\frac{\Delta s_k^T}{\Delta s_k^T \Delta s_k}.$$

ETNA
Kent State University
http://etna.math.kent.edu

Then J_{k+1} satisfies

$$J_{k+1}q = J_k q \quad \forall q \text{ such that } q^T \Delta s_k = 0. \tag{2.7}$$

2.1.2. A generalized Broyden's method. Suppose that the most recent $m+1$ iterates $\{s_{k-m}, s_{k-m+1}, \ldots, s_k\}$ are available and let

$$\Delta s_i = s_{i+1} - s_i, \qquad \Delta f_i = f(s_{i+1}) - f(s_i), \qquad i = k-m, k-m+1, \ldots, k-1.$$

We now describe a generalized Broyden's method with a flexible rank of the update of the approximate Jacobian J_k, which is required to satisfy m secant conditions,

$$J_k \Delta s_i = \Delta f_i, \qquad i = k-m, k-m+1, \ldots, k-1,$$

where the vectors $\Delta f_{k-m}, \Delta f_{k-m+1}, \ldots, \Delta f_{k-1}$ are assumed to be linearly independent and $m \leq n$. These equations can be expressed in matrix form as

$$J_k \Delta S_k = \Delta F_k,$$

where

$$\Delta S_k = [\Delta s_{k-m}, \Delta s_{k-m+1}, \ldots, \Delta s_{k-1}], \quad \Delta F_k = [\Delta f_{k-m}, \Delta f_{k-m+1}, \ldots, \Delta f_{k-1}].$$

The analogue of condition (2.7) is

$$(J_k - J_{k-m})q = 0$$

for all q orthogonal to the range of ΔS_k.

We approximate the Jacobian at the $(k+1)^{\text{st}}$ step by

$$J_{i+1} = J_i + (\Delta F_i - J_i \Delta S_i)(\Delta S_i^T \Delta S_i)^{-1} \Delta S_i^T, \quad i = 1, \ldots, k.$$

In the context of mixing, the base case is

$$J_1 = \omega I,$$

where ω is the mixing parameter. The next iterate is given by

$$s_{k+1} = s_k + \Delta s_k.$$

2.2. Anderson's method. Let $f\colon \mathbb{R}^N \to \mathbb{R}^N$ be a nonlinear function. Anderson's method is an iterative procedure for the solution of the given nonlinear system of equations $f(s) = 0$. Denote the most recent iterates by $s_{k-m}, \ldots, s_k$ and let $f_j = f(s_j)$ for all $j = k-m, \ldots, k$. Anderson's method [1, 6, 16, 25, 26] determines the next iterate, s_{k+1}, in the following way: introduce

$$\begin{aligned} \bar{s}_k &= s_k - \sum_{i=k-m}^{k-1} \gamma_i^{(k)} \Delta s_i = s_k - \Delta S_k \gamma_k, \\ \bar{f}_k &= f_k - \sum_{i=k-m}^{k-1} \gamma_i^{(k)} \Delta f_i = f_k - \Delta F_k \gamma_k, \end{aligned} \tag{2.8}$$

where

$$\Delta s_i = s_{i+1} - s_i, \quad \Delta f_i = f_{i+1} - f_i, \quad \gamma_k = [\gamma_{k-m}^{(k)}, \ldots, \gamma_{k-1}^{(k)}]$$

and

$$\Delta S_k = [\Delta s_{k-m}, \ldots, \Delta s_{k-1}], \quad \Delta F_k = [\Delta f_{k-m}, \ldots, \Delta f_{k-1}].$$

By rearranging, we get

$$\bar{s}_k = \sum_{i=k-m}^{k} \omega_j s_i, \quad \bar{f}_k = \sum_{i=k-m}^{k} \omega_j f_i$$

with $\sum_{i=k-m}^{k} \omega_j = 1$. The quantities $\bar{s}_k$ and $\bar{f}_k$ are weighted averages of $s_{k-m}, \ldots, s_k$ and $f_{k-m}, \ldots, f_k$, respectively.

The γ_i's are determined by minimizing

$$E(\gamma^{(k)}) = \langle \bar{f}_k, \bar{f}_k \rangle = \left\| f_k - \Delta F_k \gamma^{(k)} \right\|_2^2.$$

The solution satisfies the normal equations

$$(\Delta F_k^T \Delta F_k)\gamma^{(k)} = \Delta F_k^T f_k.$$

Substitution into (2.8) gives

$$\begin{aligned} s_{k+1} &= \bar{s}_k + \beta \bar{f}_k \\ &= s_k + \beta f_k - (\Delta S_k + \beta \Delta F_k)\gamma^{(k)} \\ &= s_k + \beta f_k - (\Delta S_k + \beta \Delta F_k)(\Delta F_k^T \Delta F_k)^{-1} \Delta F_k^T f_k. \end{aligned}$$

3. Vector extrapolation methods. The solution of systems of nonlinear equations by an iterative method yields a sequence of vectors (approximate solutions). When this sequence converges slowly, extrapolation often can be applied to enhance convergence. Typically, vector extrapolation methods perform better than scalar extrapolation applied to each vector component independently. The most popular vector extrapolation methods are minimal polynomial extrapolation (MPE) by Cabay and Jackson [4], reduced rank extrapolation (RRE) by Eddy [5] and Mesina [15], and modified minimal polynomial extrapolation (MMPE) by Sidi, Ford, and Smith [24], Brezinski [2], and Pugachev [17]. Convergence analyses of these methods are presented in [10, 24]. Several different recursive algorithms for implementing these methods are described in [2, 5, 7, 11]. We note that, when applied to linearly generated vector sequences, the MPE, RRE, and MMPE methods are related to Krylov subspace methods. It is shown in [18, 22] that the MPE and RRE approaches are mathematically equivalent to Arnoldi's method and to the generalized minimal residual (GMRES) method, respectively. Vector extrapolation methods are very effective solution methods for systems of nonlinear equations [9, 12].

The first subsection discusses methods for accelerating scalar sequences. We summarize classical results for scalar extrapolation. These results will be extended to vector extrapolation, which is the topic of the second subsection. Specifically, this subsection is concerned with the RRE method.

3.1. Scalar extrapolation. Given a real sequence $\{s_k\}_{k \in \mathbf{N}}$ with $\lim_{k\to\infty} s_k = s$, an acceleration transformation constructs a second sequence $\{t_k\}_{k\in\mathbf{N}}$ that converges faster than the original sequence in the sense that

$$\lim_{k\to\infty} \frac{t_k - s}{s_k - s} = 0.$$

If the original sequence is divergent, the sequence transformation acts as an extrapolation method to the anti-limit s.

ETNA
Kent State University
http://etna.math.kent.edu

3.1.1. Aitken's Δ^2 process. Given a sequence $\{s_k\}_{k\in\mathbf{N}}$ with limit s, we can write the limit as

$$s = s_k - \frac{s_{k+1} - s_k}{\frac{s_{k+1}-s}{s_k-s} - 1}.$$

If the s_k are of the form

$$s_k = s + a_1(\lambda_1)^k + a_2(\lambda_2)^k, \tag{3.1}$$

where $0 < |\lambda_2| < |\lambda_1| < 1$ and $a_1 a_2 \neq 0$, then

$$\frac{s_{k+1} - s}{s_k - s} = \frac{a_1(\lambda_1)^{k+1} + a_2(\lambda_2)^{k+1}}{a_1(\lambda_1)^k + a_2(\lambda_2)^k}.$$

On the one hand, we have

$$\frac{s_{k+1} - s}{s_k - s} = \lambda_1 + O\left(\left(\frac{\lambda_2}{\lambda_1}\right)^k\right).$$

For instance, the Lusternik process is a linear sequence transformation that gives improved convergence when applied to the sequence (3.1). It yields the sequence

$$t_k^{(1)} = s_k - \frac{s_{k+1} - s_k}{\lambda_1 - 1}, \quad k = 1, 2, 3, \ldots$$

We have $t_k^{(1)} - s = a_2 \frac{\lambda_2 - \lambda_1}{1-\lambda_1}(\lambda_2)^k$ and, therefore,

$$\lim_{k\to\infty} \frac{t_k^{(1)} - s}{s_k - s} = 0.$$

On the other hand, we also have

$$\frac{s_{k+2} - s_{k+1}}{s_{k+1} - s_k} = \lambda_1 + O\left(\left(\frac{\lambda_2}{\lambda_1}\right)^k\right).$$

This relation suggests the Aitken Δ^2 process

$$t_k^{(2)} = s_k - \frac{s_{k+1} - s_k}{\frac{s_{k+2}-s_{k+1}}{s_{k+1}-s_k} - 1}, \quad k = 1, 2, 3, \ldots$$

The $t_k^{(2)}$ satisfy

$$\lim_{k\to\infty} \frac{t_k^{(2)} - s}{s_k - s} = 0.$$

Aitken's Δ^2 process also can be expressed as

$$t_k^{(2)} = s_k - \frac{(\Delta s_k)^2}{\Delta^2 s_k} = s_k - \Delta s_k \left(\Delta^2 s_k\right)^{-1} \Delta s_k,$$

where

$$\Delta s_k = s_{k+1} - s_k, \quad \Delta^2 s_k = \Delta s_{k+1} - \Delta s_k = s_k - 2s_{k+1} + s_{k+2}.$$

This shows that the $t_k^{(2)}$ can be written as ratio of two determinants

$$t_k^{(2)} = \begin{vmatrix} s_k & s_{k+1} \\ \Delta s_k & \Delta s_{k+1} \end{vmatrix} \Bigg/ \begin{vmatrix} 1 & 1 \\ \Delta s_k & \Delta s_{k+1} \end{vmatrix}.$$

3.1.2. Interpretation of Aitken's Δ^2 process. We also have

$$t_k^{(2)} = \gamma_k^{(0)} s_k + \gamma_k^{(1)} s_{k+1},$$

where $\gamma_k^{(0)} = \frac{\Delta s_{k+1}}{\Delta^2 s_k}$ and $\gamma_k^{(1)} = -\frac{\Delta s_k}{\Delta^2 s_k}$. Consequently, $\gamma_k^{(0)}$ and $\gamma_k^{(1)}$ satisfy the linear system of equations

$$\begin{array}{lcll} \gamma_k^{(0)} & + & \gamma_k^{(1)} & = 1, \\ \gamma_k^{(0)} \Delta s_k & + & \gamma_k^{(1)} \Delta s_{k+1} & = 0. \end{array}$$

3.1.3. Fixed point methods. Let $s_{k+1} = \phi(s_k)$, $k = 1, 2, 3, \ldots$, for some nonlinear function ϕ, and assume that the s_k converge to a limit s. Then

$$\frac{s_{k+1} - s}{s_k - s} = \phi'(s) + O(s_k - s).$$

We can construct a linear convergence acceleration method

$$t_k^{(3)} = s_k - \frac{s_{k+1} - s_k}{\phi'(s) - 1}.$$

Let $f(x) = x - \phi(x)$. Then $f(s) = 0$. We define the nonlinear transformation

$$t_k^{(4)} = s_k - \frac{s_{k+1} - s_k}{\phi'(s_k) - 1} = s_k - \frac{f(s_k)}{f'(s_k)}.$$

If the s_k, for $k = 1, 2, 3, \ldots$, converge to s, then

$$t_k^{(3)} - s = O((s_k - s)^2), \quad t_k^{(4)} - s = O((s_k - s)^2).$$

Let us introduce the cycling process. We choose a starting point x_0 and at every k^{th} iteration, we set $s_0 = x_k$ and compute

$$x_{k+1} = t_0^{(4)}.$$

This yields the well-known Newton's method

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}.$$

When using Aitken's Δ^2 process in cycling mode, we set

$$x_{k+1} = t_0^{(2)}$$

and obtain Steffensen's method

$$x_{k+1} = x_k + \frac{f(x_k)^2}{f(x_k - f(x_k)) - f(x_k)}.$$

Both Newton's and Steffensen's methods yield quadratic convergence,

$$|x_{k+1} - s| \leq K \, |x_k - s|^2.$$

We will generalize the cycling process below.

ETNA
Kent State University
http://etna.math.kent.edu

3.2. Vector extrapolation. Let $\{s_k\}_{k\in\mathbf{N}}$ be a sequence of vectors in $\mathbb{R}^N$, and define the first and second forward differences

$$\Delta s_k = s_{k+1} - s_k, \qquad k = 0, 1, 2, \ldots,$$
$$\Delta^2 s_k = \Delta s_{k+1} - \Delta s_k, \qquad k = 0, 1, 2, \ldots$$

It is known that the RRE method, when applied to the sequence $\{s_k\}_{k\in\mathbf{N}}$, produces approximations $t_{k,q}^{RRE}$ of the limit or the anti-limit of the sequence $\{s_k\}_{k\in\mathbf{N}}$. These approximations are defined by

$$t_{k,q}^{RRE} = \sum_{j=0}^{q} \gamma_j^{(q)} s_{k+j}, \tag{3.2}$$

where

$$\sum_{j=0}^{q} \gamma_j^{(q)} = 1, \qquad \sum_{j=0}^{q} \eta_{ij}\gamma_j^{(q)} = 0, \quad i = 0, 1, \ldots, q-1, \tag{3.3}$$

with the scalars η_{ij} defined by the inner products

$$\eta_{ij} = (\Delta^2 s_{k+i}, \Delta s_{k+j})$$

in $\mathbb{R}^N$. It follows from (3.2) and (3.3) that $t_{k,q}^{RRE}$ can be expressed as a ratio of two determinants:

$$t_{k,q}^{RRE} = \frac{\begin{vmatrix} s_k & s_{k+1} & \cdots & s_{k+q} \\ \eta_{0,0} & \eta_{0,1} & \cdots & \eta_{0,q} \\ \vdots & \vdots & \vdots & \vdots \\ \eta_{q-1,0} & \eta_{q-1,1} & \cdots & \eta_{q-1,q} \end{vmatrix}}{\begin{vmatrix} 1 & 1 & \cdots & 1 \\ \eta_{0,0} & \eta_{0,1} & \cdots & \eta_{0,q} \\ \vdots & \vdots & \vdots & \vdots \\ \eta_{q-1,0} & \eta_{q-1,1} & \cdots & \eta_{q-1,q} \end{vmatrix}}.$$

Introduce the matrices

$$\Delta^i S_{k,q} = [\Delta^i s_k, \ldots, \Delta^i s_{k+q-1}], \quad i = 1, 2.$$

Using Schur complements, $t_{k,q}^{RRE}$ can be written as

$$t_{k,q}^{RRE} = s_k - \Delta S_{k,q} \Delta^2 S_{k,q}^{+} \Delta s_k,$$

where $\Delta^2 S_{k,q}^{+}$ denotes the Moore-Penrose generalized inverse of $\Delta^2 S_{k,q}$; it is defined by

$$\Delta^2 S_{k,q}^{+} = (\Delta^2 S_{k,q}^T \Delta^2 S_{k,q})^{-1} \Delta^2 S_{k,q}^T,$$

provided that $\det(\Delta^2 S_{k,q}^T \Delta^2 S_{k,q}) \neq 0$. We assume this to be the case. Then $t_{k,q}^{RRE}$ is well defined and unique. The vectors $t_{k,q}^{RRE}$ can be computed efficiently for several values of k and q using algorithms described in [7].

ETNA
Kent State University
http://etna.math.kent.edu

We will give an estimate for the residual norm for nonlinear problems. Introduce the new approximation

$$\tilde{t}_{k,q}^{RRE} = \sum_{j=0}^{q} \gamma_j^{(q)} s_{k+j+1}.$$

In [12], we defined the generalized residual of $t_{k,q}^{RRE}$ as

$$\tilde{r}(t_{k,q}^{RRE}) = \tilde{t}_{k,q}^{RRE} - t_{k,q}^{RRE}, \tag{3.4}$$

which can be expressed as

$$\tilde{r}(t_{k,q}^{RRE}) = \Delta s_k - \Delta^2 S_{k,q} \Delta^2 S_{k,q}^{+} \Delta s_k.$$

Notice that $\tilde{r}(t_{k,q}^{RRE})$ is obtained by projecting orthogonally Δs_k onto the subspace generated by the vectors $\Delta^2 s_k, \ldots, \Delta^2 s_{k+q-1}$. We remark that $\tilde{r}(t_{k,q}^{RRE})$ may be considered as an approximation of the true residual. In particular, when the sequence is generated linearly, $\tilde{r}(t_{k,q}^{RRE})$ is the true residual. Therefore a stopping criterion can be based on $\|\tilde{r}(t_{k,q}^{RRE})\|$.

3.2.1. Implementation of the RRE method. From an implementation perspective, we will be interested only in the case when k is kept fixed. Accordingly, we set $k = 0$ and denote the vector $t_{0,q}^{RRE}$ by t_q^{RRE}.

The linear system (3.3) is written as

$$\begin{array}{lllll} \gamma_0^{(q)} & + \gamma_1^{(q)} & + \ldots & + \gamma_q^{(q)} & = 1 \\ \gamma_0^{(q)}(\Delta^2 s_0, \Delta s_0) & + \gamma_1^{(q)}(\Delta^2 s_0, \Delta s_1) & + \ldots & + \gamma_q^{(q)}(\Delta^2 s_0, \Delta s_q) & = 0 \\ \gamma_0^{(q)}(\Delta^2 s_1, \Delta s_0) & + \gamma_1^{(q)}(\Delta^2 s_1, \Delta s_1) & + \ldots & + \gamma_q^{(q)}(\Delta^2 s_1, \Delta s_q) & = 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \gamma_0^{(q)}(\Delta^2 s_{q-1}, \Delta s_0) & + \gamma_1^{(q)}(\Delta^2 s_{q-1}, \Delta s_1) & + \ldots & + \gamma_q^{(q)}(\Delta^2 s_{q-1}, \Delta s_q) & = 0. \end{array} \tag{3.5}$$

Introduce the scalars $\beta_i^{(q)} = \dfrac{\gamma_i^{(q)}}{\gamma_q^{(q)}}$ for $i = 0, \ldots, q$. Then

$$\gamma_i^{(q)} = \frac{\beta_i^{(q)}}{\sum_{i=0}^{q} \beta_i^{(q)}} \quad \text{for} \quad i = 0, \ldots, q-1 \quad \text{and} \quad \beta_q^{(q)} = 1.$$

With this notation, the linear system (3.5) becomes

$$\begin{array}{lllllll} \beta_0^{(q)}(\Delta^2 s_0, \Delta s_0) & + & \ldots & + & \beta_{q-1}^{(q)}(\Delta^2 s_0, \Delta s_{q-1}) & = & -(\Delta^2 s_0, \Delta s_q) \\ \beta_0^{(q)}(\Delta^2 s_{q-1}, \Delta s_0) & + & \ldots & + & \beta_{q-1}^{(q)}(\Delta^2 s_{q-1}, \Delta s_{q-1}) & = & -(\Delta^2 s_{q-1}, \Delta s_q). \end{array}$$

This system can be written in the form

$$(\Delta^2 S_q^T \, \Delta S_q) \, \beta^{(q)} = -\Delta S_q^T \, \Delta s_q, \tag{3.6}$$

where $\beta^{(q)} = [\beta_0^{(q)}, \ldots, \beta_{q-1}^{(q)}]^T$ and $\Delta S_q = [\Delta s_0, \ldots, \Delta s_{q-1}]$.

Assume now that the coefficients $\gamma_0^{(q)}, \ldots, \gamma_q^{(q)}$ have been calculated, and introduce the new variables

$$\alpha_0^{(q)} = 1 - \gamma_0^{(q)}, \quad \alpha_j^{(q)} = \alpha_{j-1}^{(q)} - \gamma_j^{(q)}, \quad 1 \le j \le q-1, \text{ and } \alpha_{q-1}^{(q)} = \gamma_q^{(q)}.$$

ETNA
Kent State University
http://etna.math.kent.edu

Then the vector t_q^{RRE} can be expressed as

$$t_q^{RRE} = s_0 + \sum_{j=0}^{q-1} \alpha_j^{(q)} \Delta s_j = s_0 + \Delta S_q \, \alpha^{(q)}, \tag{3.7}$$

where $\alpha^{(q)} = [\alpha_0^{(q)}, \ldots, \alpha_{q-1}^{(q)}]^T$.

We remark that in order to determine the $\gamma_i^{(q)}$, we must first compute the $\beta_i^{(q)}$ by solving the linear system of equations (3.6). Using (3.4) and (3.7), the generalized residual $\tilde{r}(t_q^{RRE})$ can be expressed as

$$\tilde{r}(t_q^{RRE}) = \sum_{i=0}^{q} \gamma_i^{(q)} \, \Delta s_i = \Delta S_{q+1} \, \gamma^{(q)},$$

where $\gamma^{(q)} = [\gamma_0^{(q)}, \ldots, \gamma_q^{(q)}]^T$.

3.2.2. An algorithm for the RRE method. Fast, stable, and storagewise economical algorithms are described in [8, 24]. These algorithms solve least-squares problems by QR factorization. We provide an overview.

Let $\Delta S_{k,(q+1)}$ have full rank, namely $\text{rank}(\Delta S_{k,(q+1)}) = q + 1$. Then we can determine a QR factorization $\Delta S_{k,(q+1)} = Q_q R_q$, where $Q_q = [q_0, q_1, \ldots, q_q] \in \mathbb{R}^{N\times(q+1)}$ has orthonormal columns q_j, and $R_q \in \mathbb{R}^{(q+1)\times(q+1)}$ is upper triangular with positive diagonal entries. The matrix Q_q is obtained from $Q_{q-1} \in \mathbb{R}^{N\times q}$ by appending the column q_q. Similarly, R_q is obtained from $R_{q-1} \in \mathbb{R}^{q\times q}$ by appending a row and a column to R_{q-1}. The details of the resulting algorithm are described in Table 3.1. Note that, we need to store only the vector s_k and the matrix Q_q. The remaining quantities can be overwritten. We remark that the vector q_q does not have to be computed since it is not needed for determining $t_{k,q}^{RRE}$. The QR factorization can be computed inexpensively by applying the modified Gram-Schmidt process to the vectors $s_k, s_{k+1}, \ldots, s_{k+q+1}$.

TABLE 3.1
Basic RRE algorithm.

Step 0.	Input: the vectors $s_k, s_{k+1}, \ldots, s_{k+q+1}$.
Step 1.	Compute $u_i = \Delta s_i = s_{i+1} - s_i, \quad i = k, k+1, \ldots, k+q$.
	Set $U_j = \left[\, u_k \,\vert\, u_{k+1} \,\vert \ldots \vert\, u_{k+j} \,\right], \;\; j = 0, 1, \ldots$
	Compute the QR factorization of U_q, namely $U_q = Q_q R_q$.
	($U_{q-1} = Q_{q-1} R_{q-1}$ is contained in U_q).
Step 2.	Solve the linear system
	$R_q^T R_q d = e; \;\; d = [d_0, d_1, \ldots, d_q]^T, \; e = [1, 1, \ldots, 1]^T$.
	(This amounts to solving two triangular systems.)
	Set $\lambda = \sum_{i=0}^{q} d_i$.
	Set $\gamma_i = (1/\lambda)\, d_i$ for $i = 0, \ldots, q$.
Step 3.	Compute $\alpha = [\alpha_0, \alpha_1, \ldots, \alpha_{q-1}]^T$ via
	$\alpha_0 = 1 - \gamma_0; \;\; \alpha_j = \alpha_{j-1} - \gamma_j, \; j = 1, \ldots, q-1$.
	Compute $t_{k,q}^{RRE}$ via
	$t_{k,q}^{RRE} = s_k + Q_{q-1}(R_{q-1}\alpha)$.

The basic RRE algorithm of Table 3.1 becomes increasingly expensive as q increases, because the work requirement grows quadratically with the number of iteration steps q and

ETNA
Kent State University
http://etna.math.kent.edu

the storage requirement grows linearly. A good way to keep the storage requirement and the average computational cost low is to restart the RRE algorithm periodically. Table 3.2 describes the restarted method.

TABLE 3.2
Restarted RRE algorithm, with restarts every q steps.

Step 0.	Input: set $k = 0$, choose an integer q and the vectors s_0.
Step 1.	Generate $s_{j+1} = \Phi(s_j), \quad j = 0, \ldots, q+1$ (see (3.9)).
Step 2.	Compute the approximation t_q^{RRE} using the RRE Algorithm in Table 3.1.
Step 3.	If t_q^{RRE} satisfies accuracy test, stop.
	Otherwise, set $s_0 = t_q^{RRE}$, $k = k + 1$ and go to Step 1.

We will apply the RRE method to the solution of the nonlinear system of equations

$$f(x) = 0 \tag{3.8}$$

defined by the nonlinear function $f : \mathbb{R}^N \to \mathbb{R}^N$. Denote the solution of interest by s and let s_0 be an initial approximation of s. We generate the sequence $s_1, s_2, s_3, \ldots$ by a fixed-point iteration, say

$$s_{k+1} = \Phi(s_k), \quad k = 0, 1, 2, \ldots \quad \Phi : \mathbb{R}^N \to \mathbb{R}^N, \tag{3.9}$$

where $x - \Phi(x) = 0$ is a possibly preconditioned form of (3.8) with the same solution s, and is such that $\lim_{k\to\infty} s_k = s$ in case of convergence of the s_k.

We consider the following algorithm:

- choose a starting point x^0,
- at the iteration k, set $s_0 = x^k$ and $s_{i+1} = \Phi(s_i)$ for $i = 0, \ldots, q_k$,
- compute x^{k+1} such that

$$x^{k+1} = t_{0,q_k}^{RRE} = s_0 - \Delta S_{0,q_k} \Delta^2 S_{0,q_k}^{+} \Delta s_0,$$

where q_k is the degree of the minimal polynomial of $\Phi'(s)$ for the vector $x^k - s$, i.e., the degree of the polynomial p of lowest degree such that $p(\Phi'(s))(x^k - s) = 0$.

We will show quadratic convergence of the RRE method. The following theorem uses the notation

$$\alpha_k(x) = \sqrt{\det(H_k^*(x) H_k(x))},$$

where

$$H_k(x) = \left(\frac{\Phi(x) - x}{\|\Phi(x) - x\|}, \ldots, \frac{\Phi^{q_k}(x) - \Phi^{q_k - 1}(x)}{\|\Phi^{q_k}(x) - \Phi^{q_k - 1}(x)\|} \right).$$

THEOREM 1. *Let $J = \Phi'(s)$ and assume that the matrix $J - I$ is regular. We set $M = \left\|(J - I)^{-1}\right\|$. Let the Frechet derivative Φ' satisfy the Lipschitz condition*

$$\|\Phi'(x) - \Phi'(y)\| \leq L \|x - y\|, \quad \forall x, y \in D,$$

where D is an open and convex subset of $\mathbb{C}^p$. If

$$\exists \alpha > 0, \exists K, \forall k \geq K : \quad \alpha_k(x^k) > \alpha,$$

then there is a neighborhood U of s such that for all $s_0 \in U$

$$\|x^{k+1} - s\| \leq K \left(\|x^k - s\|^2 \right).$$

This result was shown in [9]. Since there exists a neighborhood of s such that the vectors $\Delta s_0, \ldots, \Delta s_{q_k-1}$ are linearly independent and the vectors $\Delta s_0, \ldots, \Delta s_{q_k}$ are linearly dependent, and since q_k is not known in practice, we can replace q_k by l_k, where l_k is such that

$$\|\tilde{r}(t_{l_k}^{RRE})\| = 0.$$

Hence, we obtain a variant of the method which has the same convergence property and which is more interesting for application. In practice, q_k is unknown. Therefore we let q_k be fixed and small.

It is useful to apply the extrapolation method to an equivalent preconditioned system. The new system will be chosen such that the new basic iteration is convergent. In this case, the application of the extrapolation scheme after a number of basic iterations is recommended.

4. Numerical experiments. All computational experiments presented were carried out using MATLAB 7.1 with the RSDFT (real-space density functional theory) software. RSDFT calculates properties of the electronic structure of molecules. It is based on previous work done by Chelikowsky's Material Science Group at the University of Texas (Austin) in collaboration with Saad and his colleagues at the University of Minnesota. These scientists created a program entitled Pseudopotential Algorithms for Real Space Eigenvalue Calculations (PARSEC) on which RSDFT is based. The algorithms used in RSDFT are simpler than those of PARSEC, with the hope that RSDFT can be used for teaching and prototyping of new algorithms.

RSDFT calculates the Laplacian and then computes the ionic potential of the molecule. The Laplacian is calculated using finite difference methods by Fornberg. The program represents the Laplacian as a linear term in nearby wave function values. The ionic potential is approximated by a pseudopotential proposed by Shaw in [23]. Then the program goes through a self-consistent loop, which calculates the Ceperly-Alder exchange correlation potential, the Hartree potential, and determines the eigenvalues and eigenvectors. If the computed results do not satisfy a specified tolerance, the loop is repeated.

For all tests, iteration is terminated as soon as a vector x with

$$\frac{\|f(x)\|}{\|x\|} \leq \text{tol}$$

has been determined. We refer to $\|f(x)\|$ as the residual error. We present results for three different problems.

4.1. Accelerating process. This section explores the potential for accelerating the simplemix iteration using the RRE method discussed in the last section. The simplemix iterations are constructed by applying the SCF algorithm given in Table 2.1.

Figures 4.1, 4.2, and 4.3 show the behavior of the residual norm, using a logarithmic scale, for the simplemix and RRE methods for several values of ω. The three figures display results for three different sizes of the nonlinear systems.

In the first example, we consider the carbon atom (C). The order of the problem is $n = 54872$. Figure 4.1 displays the computed results. Both the simplemix method with the mixing parameter $\omega = 0.4$ and the RRE method perform well, with the latter method giving a somewhat faster reduction of the norm of the residual error. We remark that the mixing parameter $\omega = 0.3$ is suitable for this problem. For the following two problems, $\omega = 0.3$ is a better choice.

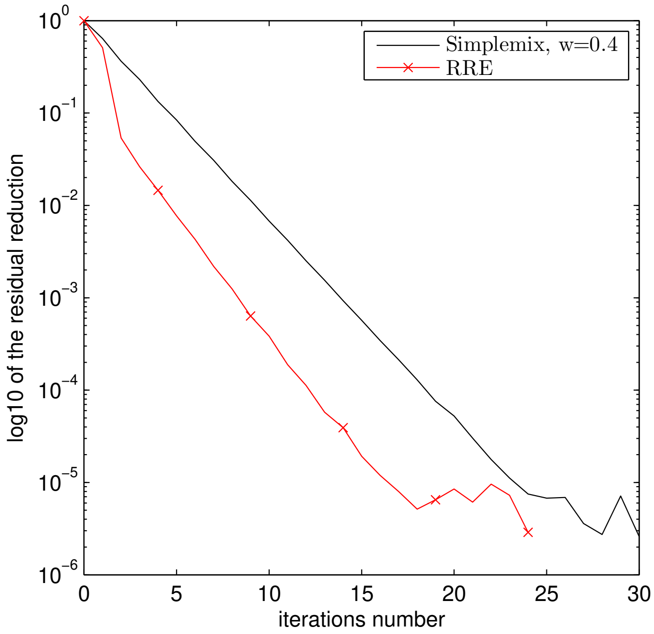

FIGURE 4.1. *Results of methods, RSDFT, C atom, $\omega = 0.4$.*

The second example applies the simplemix and RRE methods to the Sodium atom (Na) and the Silicon atom (Si). The size of the system is $n = 74088$. Figure 4.2 shows the residual reduction. The simplemix method is applied with the mixing parameter $\omega = 0.3$. The RRE method is seen to yield the fastest convergence.

Finally, in the last example, we have chosen a Carbon monoxide molecule (CO). The size of the system is $n = 157464$. Figure 4.3 displays the computed residual errors obtained with the simplemix method with $\omega = 0.3$ and with RRE. The latter method yields the fastest convergence of the residual error.

4.2. Cycling process. We compare the restarted RRE method with the generalized Broyden method and the Anderson method. These methods are applied to the same examples as above. Here RRE(k) denotes the RRE process restarted every k steps.

Figures 4.4–4.5 display the computed results. Figure 4.4 shows both the RRE(6) and RRE(10) method exhibit fast convergence, whereas the generalized Broyden method (this method performs better than the Broyden method [6]) and the Anderson method converge slower; in fact the latter methods are outperformed by the simplemix method. In Figure 4.5 all methods perform about the same.

5. Conclusion. We applied the RRE method and the restarted RRE method to the Kohn-Sham formulation of the Schrödinger equation. We obtained fairly fast convergence for all examples. Thus, vector extrapolation methods may have a useful role to play in electronic structure computations. We plan to test it in PARSEC software.

ETNA
Kent State University
http://etna.math.kent.edu

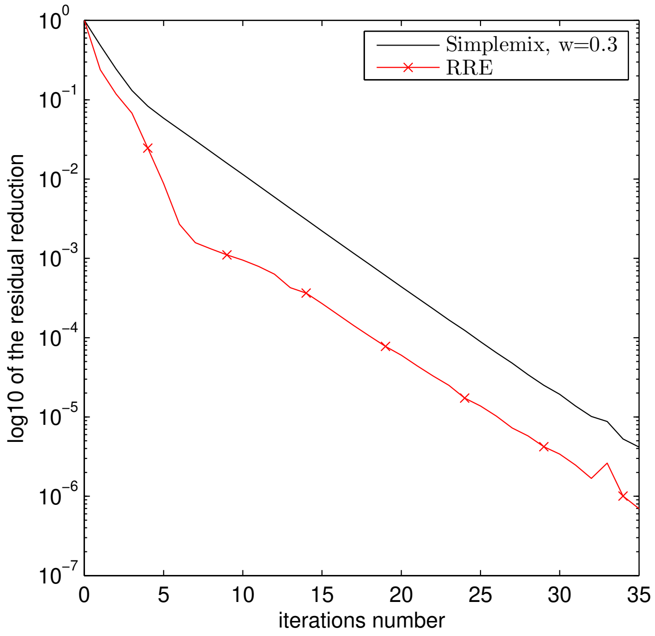

FIGURE 4.2. *Results of methods, RSDFT, Si and Na atoms,* $\omega = 0.3$.

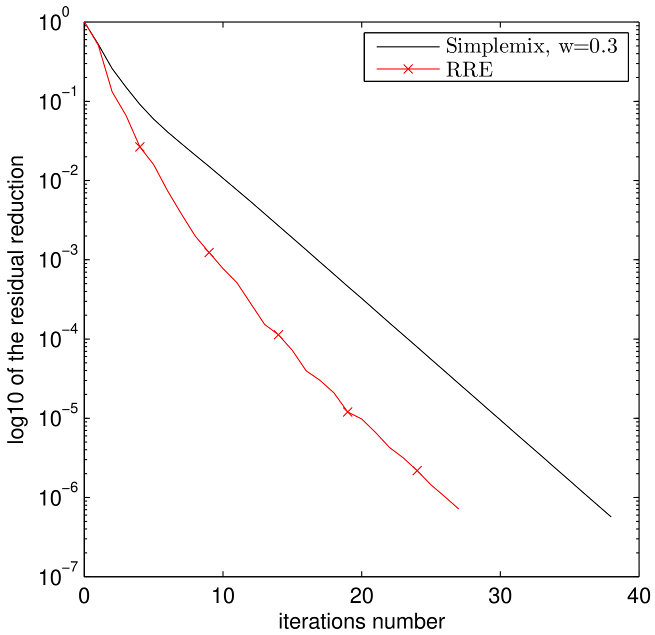

FIGURE 4.3. *Results of methods, RSDFT, CO atom,* $\omega = 0.3$.

ETNA
Kent State University
http://etna.math.kent.edu

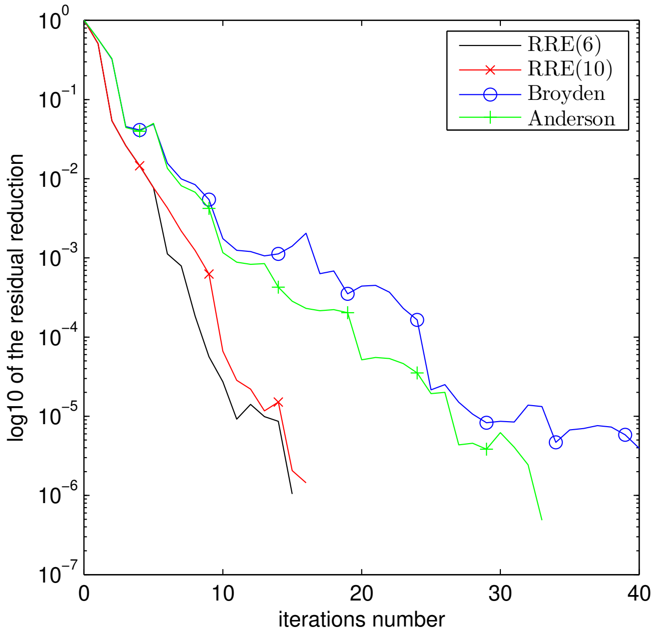

FIGURE 4.4. *Results of methods, RSDFT, C atom.*

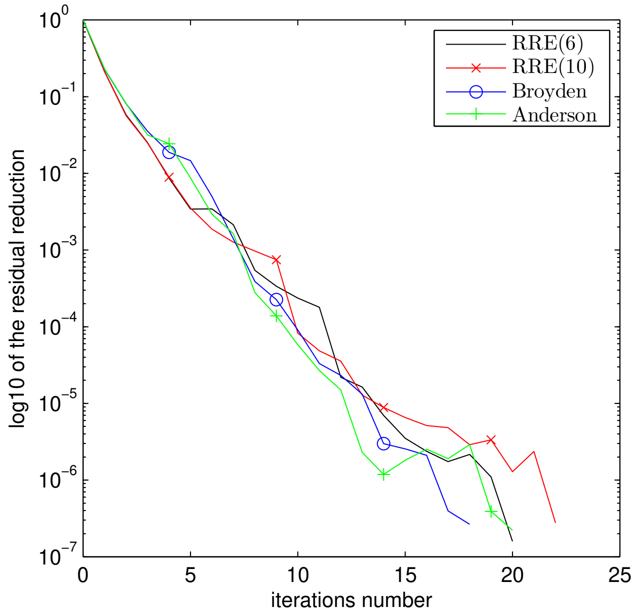

FIGURE 4.5. *Results of methods, RSDFT, Si and Na atoms.*

REFERENCES

[1] D. G. ANDERSON, *Iterative procedures for nonlinear integral equations*, J. Assoc. Comput. Mach., 12 (1965), pp. 547–560.
[2] C. BREZINSKI, *Généralisation de la transformation de Shanks, de la table de Padé et de l'ϵ-algorithm,* Calcolo, 12 (1975), pp. 317–360.
[3] C. G. BROYDEN, *A class of methods for solving nonlinear simulteneous equations*, Math. Comp., 19 (1965), pp. 577–593.
[4] S. CABAY AND L. W. JACKSON, *A polynomial extrapolation method for finding limits and antilimits for vector sequences,* SIAM J. Numer. Anal., 13 (1976), pp. 734–752.
[5] R. P. EDDY, *Extrapolation to the limit of a vector sequence*, in P. C. C. Wang, ed., Information Linkage Between Applied Mathematics and Industry, Academic Press, New York, 1979, pp. 387–396.
[6] H. FANG AND Y. SAAD, *Two classes of multisecant methods for nonlinear acceleration*, Numer. Linear Algebra Appl., 16 (2009), pp. 197–221.
[7] W. D. FORD AND A. SIDI, *Recursive algorithms for vector extrapolation methods,* Appl. Numer. Math., 4 (1988), pp. 477–489.
[8] K. JBILOU, *A general projection algorithm for solving linear systems of equations*, Numer. Algorithms, 4 (1993), pp. 361–377.
[9] K. JBILOU AND H. SADOK, *Some results about vector extrapolation methods and related fixed point iterations,* J. Comput. Appl. Math., 36 (1991), pp. 385–398.
[10] ———, *Analysis of some vector extrapolation methods for linear systems*, Numer. Math., 70 (1995), pp. 73–89.
[11] ———, *LU-implementation of the modified minimal polynomial extrapolation method*, IMA J. Numer. Anal., 19 (1999), pp. 549–561.
[12] ———, *Vector extrapolation methods. Applications and numerical comparison*, J. Comput. Appl. Math., 122 (2000), pp. 149–165.
[13] W. KOHN AND L. J. SHAM, *Self-consistent equations including exchange and correlation effects*, Phys. Rev., 140 (1965), pp. 1133–1138.
[14] C. T. KELLEY, *Iterative Methods for Linear and Nonlinear Equations*, SIAM, Philadelphia, 1995.
[15] M. MEŠINA, *Convergence acceleration for the iterative solution of $x = Ax + f$*, Comput. Methods Appl. Mech. Engrg., 10 (1977), pp. 165–173.
[16] P. NI, *Anderson acceleration of fixed-point iteration with applications to electronic structure computations*, Ph.D. Thesis, Department of Mathematical Sciences, Worcester Polytechnic Institute, 2009.
[17] B. P. PUGACHEV, *Acceleration of the convergence of iterative processes and a method for solving systems of nonlinear equations*, U.S.S.R. Comput. Math. and Math. Phys., 17 (1978), pp. 199–207.
[18] Y. SAAD, *Krylov subspace methods for solving large unsymmetric linear systems*, Math. Comput., 37 (1981), pp. 105–126.
[19] Y. SAAD, J. CHELIKOWSKY, AND S. SHONTZ, *Numerical methods for electronic structure calculations of materials*, SIAM Rev., 52 (2010), pp. 3–54.
[20] Y. SAAD, A. STATHOPOULOS, J. CHELIKOWSKY, K. WU, AND S. ÖĞÜT, *Solution of large eigenvalue problems in electronic structure calculations*, BIT, 36 (1996), pp. 563–578.
[21] Y. SAAD, Y. ZHOU, C. BEKAS, M. L. TIAGO, AND J. CHELIKOWSKY, *Diagonalization methods in PARSEC*, Phys. Status Solidi (B), 243 (2006), pp. 2188–2197.
[22] Y. SAAD AND M. H. SCHULTZ, *GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.
[23] R. SHAW, *Optimum form of a modified Heine-Abarenkov model potential for the theory of simple metals*, Phys. Rev., 174 (1968), pp. 769–781.
[24] A. SIDI, W. F. FORD, AND D. A. SMITH, *Acceleration of convergence of vector sequences,* SIAM J. Numer. Anal., 23 (1986), pp. 178–196.
[25] C. YANG, J. C. MEZA, B. LEE, AND L. W. WANG, *KSSOLV – a MATLAB toolbox for solving the Kohn-Sham equations*, ACM Trans. Math. Software, 36 (2009), pp. 10:1–10:35.
[26] 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 (2007), pp. 1854–1875.