# Parallel calculation of electron multiple scattering using Lanczos algorithms

A. L. Ankudinov,[1] C. E. Bouldin,[2] J. J. Rehr,[1] J. Sims,[2] and H. Hung[2]
[1]*Department of Physics, University of Washington, Seattle, Washington 98195*
[2]*National Institute of Standards and Technology, Gaithersburg, Maryland 20899*
(Received 12 September 2001; published 27 February 2002)

Real space multiple scattering calculations of the electronic density of states and x-ray spectra in solids typically scale as the cube of the system and basis set size, and hence are highly demanding computationally. For example, such x-ray absorption near edge structure (XANES) calculations typically require clusters of order $N_R$ atoms and *s*, *p*, and *d* states for convergence, with $N_R$ between about $10^2$–$10^3$; for this case about $10^2$ inversions of $9N_R \times 9N_R$ matrices are needed, one for each energy point. We discuss here two ways to speed up these calculations: (1) message passing interface (MPI) parallel processing and (2) fast, Lanczos multiple scattering algorithms. Together these algorithms can reduce computation times typically by two orders of magnitude. These are both implemented in a generalization of the *ab initio* self-consistent FEFF8 code, which thus makes practical XANES calculations in complex systems with of order $10^3$ atoms. The Lanczos algorithm also yields a natural crossover between full and finite-order multiple scattering with increasing energy, thus differentiating the extended and near-edge regimes.

DOI: 10.1103/PhysRevB.65.104107 PACS number(s): 71.10.−w, 78.70.Dm

## I. INTRODUCTION

Multiple scattering (MS) theory is widely used to calculate physical properties of solids, ranging from electronic structure to optical and x-ray response.[1] Within single particle theory, which is usually an adequate approximation, these physical properties can all be calculated in terms of the total one electron propagator or Green's function $G=1/(E-H+i\Gamma)$, where $H$ is the Hamiltonian and $\Gamma$ accounts for inelastic losses and lifetime effects. In this paper we focus on the real space version of MS theory, i.e., the real space Green's function (RSGF) approach,[1] which is the real space analog of the KKR (Korringa-Kohn-Rostoker) band structure method. The RSGF approach has several advantages over traditional electronic structure methods, especially for complex systems. First, a real space approach is not restricted to periodic materials, and second, the approach can be extended to energies far above the Fermi level (e.g., up to about 2000 eV). Moreover, a real space approach is essential for processes such as x-ray absorption in which a symmetry breaking effects such as the core-hole and inelastic losses (e.g., the photoelectron mean free path damping), must be taken into account, even in perfect crystals.

The central quantity in RSGF calculations is the matrix form of the propagator $G_{L'R',LR}(E)$ in a site $R$ and angular momentum $L=(l,m)$ representation $|LR\rangle = i^l j_l(kr_R)Y_{lm}(\hat{r}_R)$, where $\vec{r}_R=\vec{r}-\vec{R}$ and $L=(l,m)$. The matrix elements represent the amplitude for an electron to propagate between the states $|LR\rangle$ and $|L'R'\rangle$. This matrix satisfies the multiple-scattering equations[2] for a cluster with $N_R$ sites

$$G = G^c + G^{sc}$$

$$G^{sc} = e^{i\delta}[\mathbf{1} - G^0 T]^{-1} G^0 e^{i\delta'}, \tag{1}$$

where for simplicity here and elsewhere (unless otherwise specified), matrix indices are suppressed. In these equations $G^c$ represents the central atom contribution for which $(-1/\pi)\mathrm{Im}\, G^c = \delta_{LL'}\delta_{R,R'}$ and $G^{sc}$ is the scattering part from the surroundings. Thus the main ingredients in the calculations are the damped free propagators $G^0_{L'R',LR}(E)$, and the dimensionless scattering $t$ matrix $T=t_{lR}\delta_{R,R'}\delta_{L,L'}$ which incorporates the spherical scattering potentials in terms of $t_{lR} = e^{i\delta_{lR}}\sin\delta_{lR}$, where $\delta_{lR}$ are partial wave phase shifts for site $R$. In this formulation, multiple scattering is implicitly taken to all orders via matrix inversion. The multiple scattering series for $G^{sc}$ is obtained by expansion of the matrix inverse in a geometric series

$$G^{sc} = e^{i\delta'}[G^0 T G^0 + G^0 T G^0 T G^0 + \cdots]e^{i\delta}. \tag{2}$$

This expression give a separation of total Green's function into contributions from individual scattering paths, and is hence is termed the multiple scattering (MS) or path expansion. The complete expression in Eq. (1) is termed full multiple scattering (FMS), since it is formally equivalent to a sum over all MS paths within a chosen finite cluster through which an electron can propagate, starting from and returning to the central atom.[3]

Once the propagator $G$ is obtained using either Eq. (1) or (2), many physical quantities can be calculated. For example, the contribution to the x-ray absorption spectra (XAS) from a given site and final state angular momentum $L$ (with a relaxed core hole), as given by the golden rule, can be written as

$$\mu(E) \propto -\frac{1}{\pi}\mathrm{Im}\sum_{L,L'} M^*_{L'}(E) G_{L'0,L0}(E) M_L(E), \tag{3}$$

where $M_L(E)=\langle L,0|\hat{\epsilon}\cdot\vec{r}|c\rangle$ is a transition dipole matrix element between the atomic core state and a local final state $|L,0\rangle$ and $\hat{\epsilon}$ is the x-ray polarization. As the transition matrix elements are relatively smooth functions of energy, the fine structure in $\mu(E)$ arises predominantly from that in the scattering contribution to the propagator $G^{sc}$.

 

Remarkably the multiple scattering expansion [Eq. (2)] generally converges well for photoelectron energies above about 30 eV, where typically less than about 100 MS paths account for all structure in the the x-ray absorption spectra. This simplification justifies the traditional path-by-path EXAFS (extended x-ray absorption fine structure) analysis. On the other hand, below about 30 eV, scattering is often much stronger. There may even be eigenvalues of the $G^0T$ matrix that are larger than unity, in which case the MS expansion Eq. (2) fails to converge. For these reasons, XAS calculations are often split into two regions, based on the strength of photoelectron scattering: the EXAFS region above about 30 eV based on Eq. (2), and the XANES (x-ray absorption near edge structure) region below about 30 eV based on Eq. (1). XANES calculations probe low energy excited states and are thus important to determine electronic and chemical information from x-ray spectra (e.g, charge counts, spin and orbital momenta). However, the cross-over between these regimes is ill defined, and heretofore, has lacked a quantitative distinction. In this work, we discuss iterative MS methods based primarily on Lanczos algorithms,[4] which naturally interpolate between the full and finite MS limits. Our results below clarify how the MS expansion converges with respect to energy and when a path expansion is valid, thus providing a clear way to differentiate the extended and near edge regimes.

Many other spectroscopies can be obtained with a similar RSGF formalism. For example, calculations of the x-ray photoelectron spectroscopy (XPS) cross section $\sigma(\vec{k})$ and of photoelectron diffraction (XPD) can be obtained from $G$ using the expression[3]

$$\sigma(\vec{k}) \propto \sum_{LR} \left| \left[ Y_L(\hat{k})\,\delta_{R,R'} + \sum_{L'R'} Y_{L'}(\hat{k}) e^{-i\vec{k}\cdot(\vec{R}'-\vec{R})} G^{sc}_{L'R',LR}(E_k) \right] e^{i\delta_{l,R}} M_{LR} \right|^2, \tag{4}$$

where $\vec{k}$ is the outgoing photoelectron momentum. Calculations of Eq. (4) thus require knowledge of the site off-diagonal matrix elements of $G_{L'R',LR}$. Note that once these matrix elements are determined, one can determine $\sigma(\vec{k})$ for any direction, i.e., the ARPES (angular resolved photoemission spectra). LEED (low energy electron diffraction) spectra can also be obtained from $\sigma(\vec{k})$ and the surface structure factor.[5]

There is also a direct connection between $G$ and electronic structure. For example, the angular momentum projected density of electron states or LDOS $\rho_{l,R}$ at a given site requires similar calculations, i.e.,

$$\rho_{l,R}(E) = -\frac{1}{\pi} \mathrm{Im}\, \mathrm{Tr}_m G_{LR,LR}(E). \tag{5}$$

Thus, local electron densities and charge counts can also be obtained from $G$.

Due to the core-hole lifetime and final state broadening, the effective one-particle Hamiltonian $H$ is not Hermitian, and the free propagators decay with distance, i.e., $G^0_{L'R';LR}(E) \propto \exp[(ik - 1/\lambda_k)|R-R'|]/|R-R'|$, where $k = (2E)^{1/2}$ is the electron wave number, and $\lambda_k$ is an effective energy dependent electron mean-free path that includes lifetime broadening. The effective mean free path can be quite large near the Fermi level, but is still finite at threshold due to the core-hole lifetime. Thus the size of a cluster needed for an accurate solution to the MS equations for $G$ is roughly comparable to the mean free path $\lambda_k$, which varies between about 5 and 20 Å for all materials.

The high energy or extended regime is important in several x-ray and electron spectroscopies such as EXAFS, XPS, XPD, and anomalous x-ray scattering (AXS). These spectroscopic methods typically make use of the energy or angular dependent modulations in photoelectron scattering to extract local electronic and structural information. Due to developments in curved-wave scattering theory, e.g., the separable representation for the free propagators,[3] EXAFS and XPD calculations are now both accurate and efficient. Moreover, during the last years, computing power has increased dramatically, following Moore's law,[6] so that now such calculations using the MS expansion in Eq. (2) are readily executed on inexpensive desktop computers. As a result such calculations have become routine in EXAFS data analysis to determine accurate geometrical information about the local structure of materials.

On the other hand, XANES calculations have remained time consuming for many materials. The FMS calculations via Eq. (1) require repeated inversions of large, complex and only semisparse matrices. For example, XANES calculations at the $K$ edge of Si (which has a long mean free path) require atomic clusters of about $N_R = 10^3$ atomic sites with $s$, $p$, and $d$ electrons ($l_{\max} = 2$), i.e., a basis of dimension $N_R(l_{\max}+1)^2 \approx 9 \times 10^3$. As a result, such a calculation up to about 30 eV above threshold using Eq. (1) requires inversions of $9N_R \times 9N_R$ such matrices at about 100 energy points. These calculations typically scale as the cube of the matrix dimension, so the above example has heretofore taken several days on modern computers. Higher $Z$ materials (e.g., transition metals) may also require $f$ electrons (or higher), but tend to have shorter core-hole lifetimes, and hence comparable matrix dimensions. XANES calculations also demand a more sophisticated treatment of electronic structure than EXAFS. For example, self-consistent potentials, an accurate treatment of lifetime effects and inelastic losses are needed to obtain quantitative results. Further improvements in computer codes for XANES, such as relaxation of the muffin-tin approximation (i.e., the use of non-spherical potentials) and better treatments of many-body effects[7] will likely improve accuracy, but further increase computational time. These computational demands have led us to investigate ways to speed up the calculations.

In this paper, we present two strategies for achieving much faster near edge RSGF calculations for electronic structure and x-ray spectra. The first is to use iterative MS algorithms that replace conventional matrix-inversion methods. In particular we make use of recent developments in

Lanczos-type algorithms for solving systems of linear equations.[8] These methods are generally more efficient than the conventional continued-fraction Lanczos approach, which usually gives only a single inverse matrix element. The second strategy is to use parallel processing, i.e., by distributing different parts of the calculation across multiple processors based on the MPI protocol.[9] These techniques may be applied simultaneously, and we demonstrate that improvements in computational time for typical XANES calculations of about two orders of magnitude are possible with Lanczos MS algorithms on parallel computers. These approaches have been implemented in a generalization of the *ab initio* XAS/electronic structure code FEFF8 (version 8.2).[10] These developments thus largely overcome the time bottleneck of XANES calculations in complex materials and nanoscale systems with up to about $10^3$ atoms.

The remainder of this paper is outlined as follows. In Sec. II., we discuss the fast Lanczos algorithms for solving the MS equations and in Sec. III the MPI parallel processing approach. Section IV contains a summary and conclusions. Technical details are presented in the Appendix.

## II. ITERATIVE MS ALGORITHMS

### A. Lanczos type methods

In 1950 Lanczos[4] discovered a powerful three term recursion operation which transforms an arbitrary complex matrix $A$ to symmetric, tridiagonal form [Eq. (A1)]. Many efficient ways to solve sparse systems of linear equations are based on this transformation. These are referred to as Lanczos-type methods and have been reviewed by Gutknecht.[8] They include various versions of the biconjugate gradient (BiCG) method, the generalized minimum residual (GMRES) method, etc. Recently, additional steps have been made to improve or overcome various problems with these Lanczos procedures.[11,12]

In condensed matter physics, the Lanczos recursion method was extensively developed by Haydock *et al.*, especially for applications to tight-binding electronic structure calculations.[13] In their approach Haydock *et al.* usually use a continued-fraction representation which gives a single element of the inverse matrix $A_{11}^{-1}$. The continued-fraction recursion method was also applied to $K$-shell XANES calculations by Filipponi.[14] A great advantage of the Lanczos/continued-fraction representation is its lack of sensitivity to large eigenvalues of the matrix $A$. The reason seems to be that the procedure systematically incorporates all large eigenvalues into the solution. Indeed, the calculations of Filipponi[14] showed that this representation does not have such eigenvalue sensitivity, even when several eigenvalues are significantly larger than unity. In contrast, many other Lanczos algorithms are less stable. We found that some versions of the biconjugate gradient method need preconditioning when some eigenvalues are greater than unity and others failed to converge for the large Si calculations discussed below. The continued fraction representation has other advantages as well; for example, the spectral distribution after $n$ steps correctly gives the first $n$ moments of the distribution.[13]

A limitation of the continued fraction method is that it only gives a single element of the inverse matrix. However, for general XAS calculations, e.g., for XPS and XPD several or many elements of the inverse matrix $A^{-1}$ are needed. Such calculations can be reduced to solving a few systems of linear equations of the generic type[11,12]

$$A|x\rangle = |b\rangle, \tag{6}$$

where $A = \mathbf{1} - G^0 T$, and $|b\rangle$ has a single nonzero $j$th component, e.g., $b_i = \delta_{i,j}$ in the original basis. This is the approach adopted in this work. In this method, the vector $|x\rangle$ is then the $j$th column of the desired inverse matrix $A^{-1}$ with components $A_{ji}^{-1}$, as can be verified by direct substitution. Recently, de Abajo *et al.*[5] developed an alternative Lanczos algorithm for such calculations. Their approach calculates various inverse matrix elements recursively, in terms of the leading moments of the matrix $B = G^0 T$ obtained during the Lanczos procedure.

There have been many other attempts to speed up MS calculations. Early approaches were based, for example, on reordering the MS series,[15] but these methods appear to have only mixed success. Wu and Tong[16] introduced an iterative solution to the matrix inverse, with $|x\rangle = (\mathbf{1} + B + B^2 + \cdots)|b\rangle$ and suggested a simple mixing scheme of new and previous iterations to stabilize the approach for photoelectron diffraction. However, such a solution may not converge when the matrix $B$ has eigenvalues larger than unity, which is often the case in XANES. Another promising method is based on repartitioning the matrix $B$ to improve convergence.[17] This method splits the problem into regions with stronger and weaker scattering with different treatments appropriate to each region. Like the path expansion, this approach has the advantage of having a clear physical interpretation of the various MS contributions and can also treat strong scattering. However, it appears to be less amenable to automation than the Lanczos approach.

### B. Lanczos/LU method

As noted above, a knowledge of the complete inverse matrix $[\mathbf{1} - G^0 T]^{-1}$ is not required for all MS applications, nor is a single element sufficient. To address this problem we introduce here a combined Lanczos/LU method which generalizes the continued fraction approach to multiple $A|x\rangle = |b\rangle$ problems. This approach differs from those discussed above in that we employ the Lanczos procedure directly to calculate an entire column of the inverse matrix. In our approach, the Lanczos tridiagonalization is followed by the LU (i.e., lower upper) decomposition procedure in the transposed biorthonormal Lanczos vector space. Details are given in the Appendix. After $n$ steps of the Lanczos process the method yields an iterative solution $|x_n\rangle$ to $|x\rangle$ with components $x_i = A_{i1}^{-1}$ (for the choice $|1\rangle = |b\rangle$). The value of the first component $x_1$ agrees with the $n$-tier continued fraction result for $A_{11}^{-1}$ at each iteration. The method is computationally efficient, since the time required to obtain the entire column of the inverse matrix is the same as that for the

continued-fraction estimate of a single element $A_{11}^{-1}$. Moreover, it appears to be more direct than the recursion/moment procedure of de Abajo *et al.*[5]

In our applications the matrix/vector indices $i$ of the original basis label the combined atomic sites $R$ and angular momenta states $L=(l,m)$, i.e., $|i\rangle \equiv |LR\rangle$, and should not be confused with the Lanczos basis states $|n\rangle$ labeled by integers $n$. The vector $|b\rangle$ is chosen to define a particular seed state $|1\rangle$ for the Lanczos procedure, e.g., to obtain a particular projected density of states on some site or (e.g., through the dipole selection rules) a particular channel of the x-ray absorption process. For example, for the sDOS on the central atom, $|b\rangle = |(0,0)0\rangle$, and only one component of $|x\rangle$ needs to be calculated for that $|b\rangle$. For an unoriented powder sample or a cubic system at the $K$ absorption edge, (a common case of experimental interest) one can calculate the polarization-averaged absorption within the dipole approximation for an initial $s$ state, using at most three columns of the inverse matrix for three orthogonal vectors $|b\rangle$ corresponding to polarization components $l=1$ and $m=0,\pm 1$ at the central site.

All iterative methods that solve the $A|x\rangle = |b\rangle$ problem have the potential to give faster calculations for large clusters than the usual LU algorithm for the inverse, apart from some additional calculational overhead. Thus the exact LU matrix inversion algorithm can perform better only for small $N$ or nonsparse matrices. An exact LU solution of a system of linear equations with Lanczos methods is also formally a $O(N^3)$ operation process, and hence comparable in complexity (although slower in practice) than the exact LU algorithm. However, if a system is sparse, i.e., there are a number of nonzero elements $N_z$ in the matrix $A$, the total number of operations will scale only as $O(N_z N)$. Thus if $(N_z \ll N^2)$, the Lanczos approach will be faster than LU, even for an exact solution. Thus, due to the mean free path, one expects for large clusters that $N_z \propto N$, i.e., direct propagation only within a cluster of radius a few $\lambda_k$ is important. Yet another advantage of Lanczos methods is that they tend to reduce the residual error $|r_n\rangle$ in the inverse matrix elements with increasing $n$, where

$$|r_n\rangle \equiv |b\rangle - A|x_n\rangle. \tag{7}$$

It is natural to terminate the procedure once the residual becomes smaller than a given tolerance. Thus one generally reaches a desired precision with a number of iterations $N_{it} < N$, leading to $O(N_z N_{it})$ scaling and superior convergence properties. In the case of the MS expansion, the number $N_{it}$ corresponds to the maximum order of the expansion needed for convergence.

### C. Sample applications

Below, we compare the speed of various iterative Lanczos methods against the conventional LU method of matrix inversion.[19] We have used the calculation of the $K$ edge XAS of Si as a key test case to evaluate the effectiveness of these various alternatives. Real space MS XANES calculations of low $Z$ materials such as Si are often notoriously ill convergent, and heretofore, have not yielded good agreement with experiment. This is due partly to their open structure, strong scattering, and very long photoelectron mean free paths (i.e., long core-hole lifetime) at threshold for low $Z$ absorbers. Nonspherical potential corrections can also be important within a few eV of threshold, where they can strongly affect scattering properties. Thus these are cases that could benefit from improvements to the $N^3$ scaling of the LU algorithm. Indeed, we find that a reasonable agreement with XANES experiment for Si can be obtained only for clusters of about 600–1000 atoms.

We have implemented several Lanczos-type algorithms for the inversion of the matrix $A = \mathbf{1} - G^0 T$ (see the Appendix), in an effort to determine the optimal choices for the RSGF approach to electronic structure and x-ray spectra. These include our Lanczos/LU approach, which is similar to the biconjugate Lanczos approach (i.e., BiCG or Lanczos/Orthodir), and a stabilized version due to Van der Vorst (BiCGStab).[11]

As noted above, the matrix $G^0 T$ is short ranged due to the finite mean free path, and hence is semisparse. The presence of inelastic losses also implies that the usual Lanczos and continuum fraction algorithms for Hermitian matrices must be generalized. Thus to reduce computation time, matrix elements of $G^0 T$ are set to zero once they become smaller than some tolerance $t_1$ times the largest matrix element for a given energy. This effectively reduces the number of essentially nonzero matrix elements $N_z$. We also stop the calculation, once *all* components of the residual become less than a second tolerance $(t_2)$ for unit vector $|b\rangle$, which can be used to reduce the number of iterations $N_{it}$. Typically we set $t_1 = t_2 \approx 0.001$. We found that this is a more appropriate criterion for convergence than the norm of the residual, since the size of the matrix $\mathbf{1} - G^0 T$ changes with cluster size.

For initial tests, we carried out XAS calculations at the Si $K$ edge using a 191 atom cluster, and compared with the computation time for LU matrix inversions (taken to be 100% in the following comparisons). The LU decomposition algorithm[19] was originally used by default in the FEFF8 code, since it is one of the fastest general algorithms for exact matrix inversion. Our Lanczos/LU version of BiCG took only 57% of the time to reach convergence within the linewidth of the LU calculations $(t_2 = 0.001)$. Only the BiCGStab (Ref. 11) algorithm was found to converge faster (38% of LU time), while the Lanczos/Orthodir[12] was slightly slower. This was expected due to its similarity with our method. However, BiCGStab appeared to lose stability for some very large clusters (see below). Surprisingly, several algorithms which were expected[8] to perform as well or better than BiGCStab did not work well for our applications. Possibly this is because our matrix $A$ is both non-Hermitian and not very sparse. For example, the so-called TFQMR approach achieved convergence in 64% of LU time, while BiCGStab2 (Ref. 8) was even slower than LU. Moreover, the GMRES version of BiCG (Ref. 19) was about two times *slower* than LU and sometimes failed to converge for larger clusters.

As a result of these trials, all of our sample calculations reported below for bigger Si clusters were performed with the BiCGStab or our Lanczos/LU methods, and the results are summarized in Table I. For a 381 Si atom cluster these

TABLE I. Comparison of LU matrix inversion to iterative Lanczos type algorithms (BiCGStab and Lanczos/LU), in terms of hours of computation time on a SGI Octane 200 MHz workstation for several cluster sizes $N_R$.

| $N_R$ | 99 | 191 | 381 | 597 |
|---|---|---|---|---|
| LU | 0.28 | 1.99 | 20.8 | 60.7 |
| Lanczos/LU | 0.21 | 1.13 | 7.52 | 20.8 |
| BiCGStab | 0.16 | 0.75 | 4.24 | 11.2 |

were faster than LU by factors of 4.9 and 3.0, respectively. For 597 atoms, BiCGStab outperformed LU by a factor of 5.5.

Results for the Si *K* XAS from the largest of our calculations are shown in Fig. 1, and compared with high quality experimental data, both of which were collected using total electron yield detection at normal incidence and room temperature.[20,21] Because the sets of experimental data are essentially identical within one or two linewidths, only one is shown for clarity. The close agreement between the experimental data sets and the convergence with respect to cluster size suggests that the discrepancy between theory and experiment in the near edge region is likely due to limitations of the one-electron theory, spherical muffin-tin potentials, and electron gas self-energy used in our treatment, and not the MS expansion or limited cluster size. Although part of the discrepancy is likely due to experimental resolution, no additional broadening, Debye-Waller factors, or edge shifts were included in our XANES calculations. Also shown in Fig. 1 beyond 1855 eV is the EXAFS, as calculated with the path expansion, but with Debye Waller factors from the correlated Debye model at room temperature and Debye temperature $\Theta_D = 625$ K. These calculations demonstrate that the MS expansion for Si converges well beyond about 1875 eV. Note the discrepancy near 1846 eV between the results

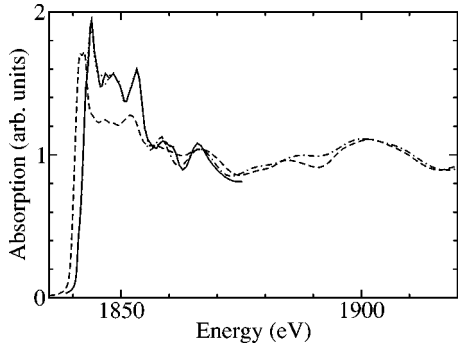


FIG. 1. Lanczos/LU XANES calculations for the Si *K* edge of 849 (solid) and 597 (dots) atom clusters, compared to room temperature experimental data (Refs. 20,21) (dashes). No additional broadening, Debye Waller factors or edge shifts were included in the XANES calculations. Also shown beyond 1855 eV is the Si *K* edge EXAFS calculated with the path expansion (dash-dots), with Debye-Waller factors from the correlated Debye model (see text). The calculations for the 849 atom cluster were done using 32 processors of the NERSC IBM SP machine with the Lanczos/LU method.

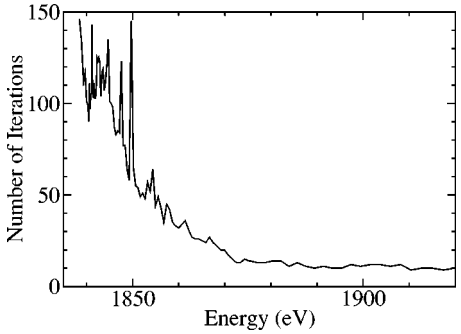


FIG. 2. Number of iterations $N_{it}$ needed for convergence of the Lanczos/LU algorithm vs energy for the 597 atom Si *K* edge XANES calculations. This number corresponds to the maximum order of the MS expansion needed for convergence. Note the approximate cross-over at about 1870 eV between XANES (where very high order MS is needed) and EXAFS (where a relatively low order MS path expansion suffices). This crossover is consistent with Fig. 1.

for 597 to 849 atoms clusters; this demonstrates that 600 atom clusters are generally sufficient for convergence of the MS expansion for Si, but that clusters of 1000 or more atoms are needed at some energies. The calculations even for 597 atoms are very time consuming, requiring about 21 hours on a single processor SGI with a 200 MHz R10K processor, and about 6 h on a dual Alpha EV-6 processor machine with our parallel MPI implementation (see Sec. III). The calculations for 849 atoms with BiCGStab failed to converge for several energy points, and were slower than those with Lanczos/LU method, which did not show any signs of such instabilities. The calculations in Fig. 1 for 849 atoms were done with 32 processors on the NERSC IBM SP computer, and took only 1.86 wallclock hours; according to Table I one would expect a calculation time of order 54 hours on a 250 MHz SGI Octane workstation. The scaling of computation time both for BiCGStab and Lanczos/LU methods varies approximately as $N^{2.5}$ in this limited cluster size range. The scaling exponent is 3.2 for LU, 2.6 for Lanczos/LU and 2.4 for BiCGStab. The reason is that the number of nonzero matrix elements $N_z$ in $G^0T$ still scales roughly as $N^2$ due to the large mean free path in Si, and the time savings are due to the fact that $N_{it}$ scales as $N^{0.5}$. Eventually, i.e., for thousands of atoms in the cluster, one would expect the number of nonzero elements in $G^0T$ to scale linearly with $N$, since distant pairs of atoms would not contribute, and for the same reason the $N_{it}$ should eventually saturate. However, for Si even with almost a thousand atoms we did not reach this limit.

In Fig. 2 we plot the number of matrix-vector multiplications $N_{it}$ required for the BiCGStab method to converge to a tolerance $t_2$ for each energy point. This number has a physical interpretation as the maximum order of MS expansion necessary to reach convergence. For Si, one sees that $N_{it}$ is of about 100 at threshold (1838 eV), but drops rather quickly to about 50 at 15 eV and then to 12 or less about 35 eV above the edge for the chosen tolerance $t_2 = 0.001$. By inspection of the EXAFS calculations in Fig. 1, these values of $N_{it}$ may be overestimates, since the EXAFS appears to be reasonably converged beyond about 35 eV of threshold. In-

terestingly, the variation of $N_{it}$ with energy appears to be correlated with the magnitude of the fine structure, which is a measure of overall scattering strength. Thus it is largest near threshold, becomes small at sufficiently high energy, and exhibits a ''fine structure'' of its own. Hence these results clearly show how the full MS expansion applied to Si crosses over to the high-order MS expansion about 35 eV above threshold, consistent with the results shown in Fig. 1. For comparison, we checked that for fcc Cu metal, the maximum number of iterations is at most 8 throughout the XANES, showing that the high order path expansion for Cu is always valid. This was not completely unexpected, since it has been verified numerically that the MS expansion converges for Cu,[3] but it is also known that some eigenvalues of $G^0T$ for Cu exceed unity near threshold.[22] Of course, the maximum MS order can also depend on cluster size, but this value is expected to tend to a limit for clusters larger than a cutoff of order the mean free path. For example, if $N_{it}$ becomes less than 10 above a certain energy, one expects that the MS path expansion will converge well beyond that energy.

## III. PARALLEL PROCESSING

### A. MPI parallel calculations of XANES

As discussed in the Introduction, a series of similar MS calculations must be done at a large number (typically of order $10^2$) of energy points to obtain a complete XANES spectrum. This number is determined by the natural energy resolution (due to lifetime broadening and inelastic losses) and the range of the XANES region (typically below 30 eV of threshold) for which full MS calculations are needed. Thus it is reasonable to consider doing these similar MS calculations in parallel.

Parallel algorithms generally depend on specifying independent tasks that can be executed simultaneously by different processors, and can be implemented in several ways: (1) using the natural parallelism intrinsic to independent physical processes; (2) using independent repeated elementary mathematical operations; and (3) using independent computational tasks, such as rewriting matrix inversion routines to execute in parallel on a set of processors. The first approach is particularly advantageous when it can be applied. Since we aim to model the physical process of x-ray absorption, it is natural to exploit the intrinsic task-parallelism (or physical parallelism) in this problem, namely, that the x-ray absorption at a given x-ray energy is independent of the absorption at other energies, assuming they are separated by the inherent energy resolution (typically a fraction of an eV). Thus a natural way to parallelize the full spectral calculation is simply to distribute the energy points among an available set of processors. The results can the be assembled to obtain the full absorption spectrum. The second approach listed above is hardware specific, and exploits the characteristics of particular processors (e.g., vector processors such as Altivec in PowerPC G4 processor or MMX in Intel processors). The third approach can also be fruitful in special applications, but can require substantial revisions to existing code and/or large amounts of communication time between processors. We do not address methods (2) and (3) further here, primarily because they require demanding recoding, and tend to be machine specific and hence not portable.

Thus in this work we exploit only the natural parallelism in XAS. To this end we have used the message passing interface (MPI) protocol.[9] MPI is now a standard library for implementing parallel processing, and is used with FORTRAN, C, or C++ and standard TCP/IP ethernet communications. It leads to a fast, portable system for parallel processing of problems with intrinsic parallelism. Using MPI, we have developed a parallel version of the *ab initio* full MS XANES code FEFF8,[2] based on the RSGF approach as briefly described in the Introduction. This parallel code (here dubbed FEFFMPI) compiles and runs without changes on all currently available operating systems tried to date (e.g., LINUX, WINDOWS NT, IBM-AIX, SGI, CRAY, . . . ). To realize such a parallel processing code, we began with the recent single-processor version of FEFF8 (version 8.10). FEFF8 is written in portable FORTRAN77, and uses a number of computational strategies for efficient calculations. Our goal is to implement a parallel processing version that retained all the advantages and portability of the the single-processor version, while gaining a significant improvement in speed on parallel machines. We also wanted a single code base, so improvements such as the Lanczos algorithms and other future developments can be incorporated without major recoding. The resultant code FEFFMPI runs on any parallel processing cluster or multiple-cpu machine that supports MPI. Moreover, such systems need not be homogeneous and can use distributed or shared memory (or even a mixture).

The starting point for ''parallelizing'' any code is to determine which parts of the calculation are the most time consuming. Profiling tests showed that three small sections of the 33 000 line FEFF8 code dominated the computational time: the calculation of self consistent potentials (SCF), the optional calculation of the local density of electronic states (LDOS), and the XANES calculation itself. All of these sections involve repeated full MS calculations (i.e., large matrix inversions) and altogether these steps account for about 97–98 % of the total runtime in typical XANES calculations. Altering these calculations to run in parallel is straightforward with MPI, because each step involves similar calculations at a different energy and utilizes identical matrix inversion routines. Thus the main computational bottleneck in FEFF8.1 is the LU matrix inversion algorithm used in solving the full MS problem.

By concentrating on a few ''hot spots'' in the code, we left most (over 99.5%) of the original single-processor code unchanged. FEFF8 reads a single plain-text free-formatted input file (FEFF.INP), which contains the atomic positions and atomic numbers defining the system as well as other relevant input data. We retained the same input file in FEFFMPI by running the parallel code from a single cross-mounted directory on a single *executive* node. Thus, all read and write statements go to the *executive* node on the MPI cluster containing $N_p$ processors. The parts of the code that still run sequentially are still executed by all nodes simultaneously. This results in a small amount (typically a few percent) of redundant calculation, but no reduction in overall wall clock

runtime. When the subroutine that executes a FMS calculation is reached, the MPI libraries are used to designate some cluster node as the executive node, and the remaining $N_p - 1$ nodes as *workers*. Each worker is assigned a unique integer identification number or MPI ''*rank*.'' In the cyclic loop over the $N_e$ x-ray energies (labeled by the points $i_e$ in the SCF, LDOS, and XANES calculations, each node (executive or worker) executes a fraction $\approx 1/N_p$ of the FMS calculations, at the energy points rank, rank$+N_p$, rank $+2N_p$, . . . . That is, the main loop seen by each processor has the behavior

$$\text{Do } i_e = \text{rank},\ N_e,\ N_p. \tag{8}$$

After each processor completes its part of the task, the results are sent back to the executive (e.g., by ethernet communication). This approach has the following properties: (1) exactly the same executable code is run on every node in the cluster, because the only distinction made between processors is the processor *rank*; (2) all of the changes to the single-processor FEFF8 are confined within a few subroutines; (3) the FEFFMPI version of the code is virtually identical to the single-processor version of FEFF8; and (4) communication between executive and worker processors is kept to a minimum. Thus the only fundamental difference is that each clone of the FEFFMPI process is aware (by virtue of its rank) that it is on a different node of a MPI cluster of $N_p$ processors. With this code structure, we succeeded in using a single code-base for both the single processor and MPI versions of FEFF8. For the single-processor version, this required the substitution of dummy MPI libraries into the original code.

Only one section of the original FEFF8 code had to be rewritten in our MPI implementation. Since this illustrates a common problem in implementing task parallelism, it is worth a brief comment. As noted above the key to implementing task parallel calculations is that the tasks must be independent of one another. However, in the original FEFF8 algorithm for self-consistent potentials, a ''smart search'' procedure was used to determine the Fermi level. Such a search cannot be run in parallel because it is an iterative algorithm, with successive steps depending on previous ones. To make the SCF calculation run in parallel, we had to replace the iterative search with a grid search method. Although the resultant algorithm is somewhat slower on single processors (i.e., by a factor of about 1.6), it can be task parallelized and yields excellent scaling of the calculation speed with cluster size. This algorithm change amounted to only about 100 lines of recoding (out of 33 000).

### B. Parallelization with fast Lanczos algorithms

It is straightforward to replace the default LU algorithm for inverting the multiple-scattering matrix $\mathbf{1}-G^0T$ with the fast Lanczos algorithms investigated in this work, simply by substituting subroutine calls. However, since fewer Lanczos iterations are needed at high energies, this led to a situation in which different execution times were needed by the various processors. Ideally, for the parallel execution, one wants to spread jobs evenly between processors or ''load-level'' the calculation. To compensate for this load imbalance, we assigned each processor a widely spaced fraction of the energies, designed to cover the total spectral range, as in the cyclic loop in Eq. (8).

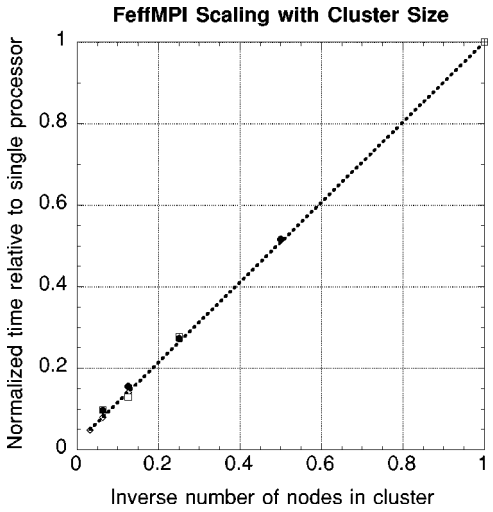


FIG. 3. FEFF8MPI total runtime scaling vs inverse cluster size for LINUX, SGI, WINDOWS NT, and IBM SP2 clusters. Once the calculation times are rescaled by a factor $T_0$ to account for single processor speed (see text), the FEFFMPI scaling is independent of processor and cluster. This scaling can be used to predict the speedup of FEFF8 on any MPI cluster, relative to a single processor in that cluster.

### C. Scaling with cluster size

To evaluate how well the parallel algorithm succeeds in FEFFMPI, we first conducted tests of XANES and LDOS calculations for an 87 atom GaN system on six computer systems. As representative single-processor systems, we used a 450 MHz AMD K6-3 running SuSe LINUX 6.1, and a 450 MHz Apple PowerMac G4 running OS 9. For the MPI clusters we used: (1) a cluster of 48 Pentium II 500 MHz systems running REDHAT LINUX; (2) a similar cluster of 400 MHz Pentium III machines running WINDOWS NT; (3) a cluster of SGI R12K machines running IRIX 6.5; and (4) a 16 processor IBM SP3 running AIX. All of these systems were connected via 100 MB ethernet. In these tests, the fastest clusters completed the total calculation about 50 times faster than the single processor Linux system. However, despite the disparate nature of these machines, we found that the relative processing speed could be fit to a simple scaling law as function of MPI cluster size, given by

$$T = T_0[0.03 + 0.97/N_p], \tag{9}$$

where $T$ is the net runtime, $T_0$ is a constant which represents the speed of a given processor type and the efficiency of the compiler, and $N_p$ is the number of processors in the MPI cluster, as shown in Fig. 3.

Once the single processor execution time $T_0$ is determined, the speed of every MPI cluster scales almost identically. As cluster size is increased, the part of the code that runs in parallel changes from the dominant part of the runtime, to a small or insignificant fraction of the total. In the limit of large MPI clusters, the runtime is then dominated by

the 3% of the original runtime which at present still executes sequentially. Thus in extremely large clusters, we would expect no further increase in speed because the runtime is totally dominated by sequentially executing code, and large clusters can even *increase* the runtime due to communications overhead.

### D. Sample applications of FEFFMPI

In this section we give some examples of how the fast turnaround of XANES calculations using FEFFMPI can be used to advantage. For this sample application, we have chosen a simulation of the changes in XANES in thin films of $BaTiO_3$ as a function of deposition conditions. In these materials, $BaTiO_3$ films are deposited on silicon or MgO substrates that are held at relatively low temperatures. Because of the low substrate temperatures and incomplete control of the deposition process, the actual structure of the films departs from that of ideal $BaTiO_3$ in unknown ways. In an attempt to determine the structure, we first used FEFFMPI to calculate the XANES for a series of possible trial structures and then used these structures as a starting point for fitting the EXAFS. The input to FEFFMPI is a cluster of 119 atoms around a central Ti atom, which includes full occupancy of all atomic sites in the $BaTiO_3$ structure to a distance of 7 Å from the central Ti atom. Our original hypothesis about the film structure was that there is a variation in the oxygen coordination around the Ti atom, and that the films contained regions of fourfold-, fivefold, and sixfold oxygen coordination. This hypothesis was motivated by the observed correlation between energy position and peak size shifts of the small ''pre-edge peak'' at the Ti K absorption edge in previous studies of various Ti-O compounds.[18] This peak appears to be due to a hybridization between *p* and *d* states on Ti largely mediated by multiple scattering. However, the $BaTiO_3$ films showed a much smaller peak size change than is found in empirical standards, as the Ti-O geometry changes from tetrahedral to square-pyramidal to octahedral. Using FEFFMPI we had fast ($\approx$20 min) turnaround on full MS XANES calculations, and were able to try out many different structural models. X-ray fluorescence (XRF) measurements showed that many of the films departed significantly from the ideal 1:1 Ba-Ti stoichiometry of $BaTiO_3$, so we eventually tried models that preserved the $BaTiO_3$ structure but included randomly distributed Ba vacancies in amounts consistent with the XRF. Figure 4 shows the experimental data from the $BaTiO_3$ films and the XANES calculated from a model that assumes an intact $BaTiO_3$ structure with full octahedral O coordination about the Ti atoms, but with an increasing number of Ba vacancies.

Clearly there is good qualitative agreement between the simulation series and the data. From this starting point, we were able to model the EXAFS data and then determine that the film structural variation is clearly due to Ba vacancies around octahedral Ti-O structural units. Details of this analysis will be given elsewhere.

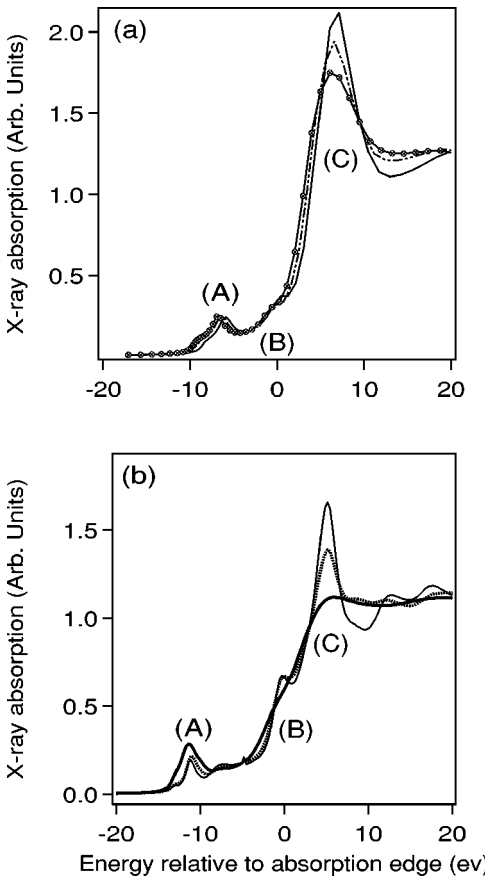


FIG. 4. Ti *K* edge XANES of (a) theoretical FEFFMPI calculations of the $BaTiO_3$ structure, assuming full Ba occupancy (solid line), two vacancies (dash-dots), and four vacancies (circles and solid line) in the Ba second shell. Note the slight shift of the pre-edge peak (A) to lower energy, the change in the inflection point (B) and the decrease in peak (C); (b) experimental data from $BaTiO_3$ films deposited at various substrate temperatures and with varying Ba:Ti ratios. XANES from these Ba-deficient samples show similar trends in the three feature changes seen in the FEFFMPI calculations.

## IV. CONCLUSIONS AND FUTURE PROSPECTS

We have demonstrated how parallel processing using the MPI protocol, combined with modern Lanczos type MS algorithms can speed up real-space XANES and electronic structure calculations by about two orders of magnitude. Similar techniques can be applied to many other spectroscopies. Our Lanczos/LU algorithm is both efficient and stable. It is an improvement on other Lanczos approaches in electronic structure calculations such as the continued fraction representation, in that it yields many elements (i.e., an entire column vector) of the matrix inverse simultaneously without additional calculation. Moreover, the approach also defines a natural crossover energy between the applicability of full and finite MS calculations, thus unambiguously distinguishing the near edge and extended regimes.

These combined developments largely overcome the computational time bottleneck of XANES calculations in large, complex systems. In particular the MPI approach alone can yield typically a 30-fold speed increase compared to an equivalent single processor system. The Lanczos/LU algorithm yields an additional factor typically between 3–5. These speed-up factors are system dependent, depending for example, on the size of the XANES region (or equivalently

the net scattering strength), mean free path, and experimental resolution. Thus the speed-up factor can be significantly greater when the MS expansion converges rapidly. The parallel processing algorithm exploits the natural or physical parallelism implicit in a XANES calculation and scales well for clusters of up to about 48 processors in our GaN test. Above that number, the execution time becomes dominated by the part of the original FEFF code (presently about 3%) which still runs sequentially.

The combination of both MPI and Lanczos algorithms changes the overall scaling parameters of the calculation, since a larger fraction of the total time is then taken by unavoidable sequential part. Indeed, using both MPI and Lanczos can sometimes totally reverse the situation that existed with the single processor version of FEFF8. The matrix inversion steps that previously dominated the execution time can now become irrelevant. This occurs, for example, at high energies (see Fig. 2), when the calculation converges in a few iterations, just like in path-expansion approach to EXAFS.

The new algorithms developed here have been incorporated into the *ab initio* FEFF8 code for calculations of XAS and electronic structure in version 8.2. The resultant FEFFMPI code is nearly as portable as the original FEFF8 code, since MPI libraries are available for most modern platforms. We have demonstrated an inverse scaling of FEFFMPI with the number of processors on several MPI CPU clusters. The code is also compatible with shared or distributed memory clusters, and with combinations of both memory architectures, i.e., networks with shared memory machines on each node. Thus the code also works with heterogeneous clusters, although the speed in that case is limited by the slowest processor. The present version of FEFFMPI uses only task parallelism, by giving each processor the same fraction of the energy values required for the full XANES and LDOS calculations, and approximately the same computational workload. Obviously, the scalability of this approach ends when each processor has only one FMS calculation to deal with; this will occur in cluster sizes of about 100 processors for typical XANES spectra. But a similar parallel approach could be employed to generate the entire EXAFS spectra as well, a step that typically requires an additional 100 energy points.

To obtain even faster XANES calculations, there are two obvious next steps to consider. One is to examine the FEFFMPI code for secondary hot spots. For example, the present code does not take advantage of the considerable redundancy in the elements of the large matrix $G^0T$. Nor do we take advantage of the separation of $G^0$ into energy independent rotation matrices (depending on bond angles) and energy dependent radial quantities, as in the Rehr-Albers approach. This has been used to advantage in the FEFF path expansion algorithms,[3] and also by de Abajo *et al.*[5] in their Lanczos algorithms. These redundant calculations likely account for a large fraction of the remaining sequential part of the calculation. Although implementing this requires recoding, it should lead to a considerable improvement both in execution time and in in overall storage requirements. Another advantage of the Lanczos procedure, is that it permits a formulation in which storage can be reduced considerably by storing only the results of matrix-vector multiplications at each stage, rather than the entire matrix. Yet another step would be to better *load level* the parallel calculations, e.g., by assigning more of the calculation to processors whose tasks finish more quickly. In addition, the Lanczos algorithms can determine the cross-over energy between XANES and EXAFS. With knowledge of the cross-over energy, one can now considering automatically replacing the FMS calculations with the more efficient path expansion beyond that energy, or alternatively, doing parts of the path expansion with the fast Lanczos approach. It may also be possible in the future to bypass the Lanczos algorithm, and explicitly parallelize the matrix inversions. We did not investigate this latter possibility, here, since the routines available for this purpose are not generally portable and tend to have large communication overhead.

With the improved efficiency now in hand, it becomes feasible to carry out XANES calculations in an entirely different manner than heretofore possible, and hence many applications can now be treated which otherwise would be impracticable. For example, a few days of calculations on the 48 processor Linux cluster can now complete a calculation that would now take a year on a current single processor workstation. Systems such as complex minerals, oxide compounds and biological structures, and other nano-scale systems are obvious targets for this type of improved computational capability. The improved speed should be very useful, for example, for magnetic materials, which often have a large number of inequivalent sites of the absorbing atom, requiring many separate calculations to produce a full XANES or XMCD (x-ray magnetic circular dichroism) spectrum. In addition, FEFFMPI should be more useful for routine applications of XAS, since it allows users to test different models rapidly and to examine subtle variations between models. Finally the availability of rapid calculations now permits efficient closed loop fitting of XANES spectra both to physical and chemical properties.

## ACKNOWLEDGMENTS

We acknowledge the support of C. Spangler and H. Fang for their efforts in the development of the cluster systems used in part of this work, and thank P. Lagarde, A. M. Flank, and A. P. Hitchcock for the experimental Si data shown here. We also thank G. de Abajo, G. Bertsch, A. Canning, T. Fujikawa, R. Haydock, M. Newville, B. Ravel, C. Reschke, and T. Schulthess for useful comments. This work was supported in part by U.S. DOE Grant No. DE-FG03-97ER45623 (JJR) and DE-FG03-98ER45718 (ALA), and was facilitated by the DOE Computational Materials Sciences Network. Certain commercial products are identified in the article for the sake of completeness; however, this does not constitute an endorsement by the National Institute of Standards and Technology.

## APPENDIX: LANCZOS/LU ALGORITHM FOR A SYSTEM OF LINEAR EQUATIONS

In this appendix we present the details of our Lanczos/LU version of the BiCG method to solve a system of linear equa-

tions $A|x\rangle=|b\rangle$ for a complex, nonsymmetric matrix $A$. Although the continued fraction expression for $A^{-1}_{1,1}$ is highly successful in many practical applications, e.g., x-ray absorption,[14] electronic structure,[13] and vibrational motion,[23] it is limited to a single element of the inverse matrix. Thus we aim to extend the method to obtain the full column vector $A^{-1}_{n,1}$. This is equivalent to solving a general system of linear equations. In order to keep the same result for $A^{-1}_{1,1}$ as the continued fraction we first perform a Lanczos transformation of $A$ to the same tridiagonal form, and then carry out its LU decomposition. As a result we obtain recursive expressions for both the iterative solution of the inverse matrix elements and the residual, $|r\rangle=|b\rangle-A|x\rangle$.

The essence of the Lanczos algorithms is a three-term recursion relation[4] Eq. (A1), which by construction tridiagonalizes an arbitrary complex (not-necessarily Hermitian) matrix $A$ in a biorthonormal set of basis states $|n\rangle$ and $\langle n|$, $n=1,2,\ldots,N$, with the same coefficients $a_n$ and $b_n$, i.e.,

$$b_n|n+1\rangle=A|n\rangle-a_n|n\rangle-b_{n-1}|n-1\rangle,$$

$$b_n\langle n+1|=\langle n|A-a_n\langle n|-b_{n-1}\langle n-1|. \tag{A1}$$

In this basis, $\langle n|n'\rangle=\delta_{nn'}$, and the only non zero matrix elements are $\langle n|A|n\rangle=a_n$ and $\langle n+1|A|n\rangle=\langle n|A|n+1\rangle=b_n$. For Hermitian matrices $A$, only one of the above equations is needed, and the coefficients $a_n$ and $b_n$ are real, but we cannot take advantage of that simplification in this work.

The LU decomposition of the resulting tridiagonal matrix can be also updated on each iteration. Both the $L$ and $U$ matrices are bidiagonal, and their only nonzero matrix elements are given by

$$L_{n,n}=1,$$

$$U_{n,n}=\alpha_n=a_n-b_{n-1}\alpha_{n-1},$$

$$L_{n+1,n}=\beta_n=b_n/\alpha_n,$$

$$U_{n,n+1}=b_n. \tag{A2}$$

After $n$ steps we find the iterative solution $|x_n\rangle$ of the system of linear equations $A|x\rangle=|b\rangle$, by using the incomplete bidiagonal $L$ and $U$ matrices in two steps, solving sequentially (i) the $L|y\rangle=|b\rangle$ and (ii) the $U|x_n\rangle=|y\rangle$ problems. The intermediate vector $|y\rangle=\Sigma\gamma_n|n\rangle$ is found by forward substitution, and on the $n$th iteration only $\gamma_n$ changes from zero to the calculated value, while all previous components remain the same, i.e.,

$$|y_{n+1}\rangle=|y_n\rangle+\gamma_{n+1}|n+1\rangle, \tag{A3}$$

$$\gamma_{n+1}=-\beta_n\gamma_n.$$

The components $|x_n\rangle$ and the residuals $|r_n\rangle$ are found from $|y_n\rangle$ by backward substitution using the complimentary vectors $|z_n\rangle$ and $|s_n\rangle$,

$$|z_{n+1}\rangle=|n+1\rangle-(b_n/\alpha_n)|z_n\rangle,$$

$$|s_{n+1}\rangle=A|n+1\rangle-(b_n/\alpha_n)|s_n\rangle. \tag{A4}$$

Finally the following recursion relations are obtained for the iterative solution $|x_n\rangle$ and the residual $|r_n\rangle=|b\rangle-A|x_n\rangle$:

$$|x_{n+1}\rangle=|x_n\rangle+(\gamma_{n+1}/\alpha_{n+1})|z_{n+1}\rangle,$$

$$|r_{n+1}\rangle=|r_n\rangle-(\gamma_{n+1}/\alpha_{n+1})|s_{n+1}\rangle. \tag{A5}$$

This algorithm for the solution of a general system of linear equations can be summarized in the following pseudocode: START Lanczos/LU: Set

$$|x_0\rangle=0,$$

$$|1\rangle=|z_1\rangle=|r_0\rangle=|b\rangle-A|x_0\rangle,$$

$$|s_1\rangle=A|1\rangle,$$

$$\langle 1|=|1\rangle^\dagger/\langle 1|1\rangle,$$

$$\alpha_1=a_1=\langle 1|A|1\rangle,$$

$$\gamma_1=1,$$

$$|x_1\rangle=|x_0\rangle+(\gamma_1/\alpha_1)|z_1\rangle,$$

$$|r_1\rangle=|r_0\rangle-(\gamma_1/\alpha_1)|s_1\rangle. \tag{A6}$$

DO $n=1$, nitx.

(1) Perform Lanczos transformation: get $b_n$ – the dot product of right hand sides of Eq. (A1) gives $b_n^2$; new basis vectors $|n+1\rangle$ and $\langle n+1|$; and finally $a_{n+1}$.

(2) Update LU decomposition: $\beta_n$, $\alpha_{n+1}$ and $\gamma_{n+1}$ using Eqs. (A2) and (A3).

(3) Use recursion relations (A4),(A5) to update solution $|x_{n+1}\rangle$ and residual vector $|r_{n+1}\rangle$.

(4) STOP when all components of residual are less than a tolerance: $|r_{n,i}|<t_1$.

ENDDO

This procedure tends to run out of precision typically after about 100 iterations (with single-precision arithmetic), and needs to be restarted with $|x_0\rangle=|x_n\rangle$ in Eq. (A6).

Notice that the most time consuming part of the algorithm is the Lanczos transformation itself [Eq. (A1)], which requires two multiplications of a vector by the matrix $A$ per iteration, unless $A$ is Hermitian, in which case only one multiplication is needed. For an arbitrary matrix $A$, this algorithm actually solves both the $A|x\rangle=|b\rangle$ and the $\langle x|A=|b\rangle^\dagger$ problems. By construction, our method gives the same result as the continued-fraction expression for $A^{-1}_{1,1}$ when the vector $|b\rangle$ components are $b_i=\delta_{i,1}$, where $i$ refers to the original basis states (i.e., $|i\rangle=|LR\rangle$, and $|1\rangle$ is the chosen initial state of the Lanczos procedure. However, Lanczos/LU also yields an entire vector or column of the inverse matrix, $A^{-1}_{i,1}$ at no additional computational cost, since it can be found from iterative solution as the component in the original basis of the approximate solution $|x_n\rangle\approx|x\rangle$, i.e.,

$$A^{-1}_{i,1}=\langle i|x\rangle. \tag{A7}$$

Notice that the first row of the matrix $A$ can also be found at no additional cost if desired, since Eq. (A3)–(A5) hold for the bra vectors as well as for ket vectors and

$$A^{-1}_{1,i} = \langle x | i \rangle, \tag{A8}$$

i.e., the $i$th component of the first row is given by the $i$th component of the vector $\langle x_n |$.

The Lanczos/LU algorithm is mathematically similar to other forms of the Lanczos biorthogonal conjugate gradient (BiCG) approach, like Lanczos/Orthodir, Lanczos/Orthomin, Lanczos/Orthores, and BiO algorithms.[12,24] All of these methods differ in various details, and the main distinction of our Lanczos/LU approach is that the Lanczos transformation is carried out explicitly, which maintains the continued-fraction value for the component $A^{-1}_{1,1}$, while in other versions this equivalence is only implicit. Remarkably the Lanczos/LU algorithm appears to be more stable numerically than most other Lanczos type algorithms investigated here. Thus we had to restart the Lanczos/LU iterations only after about 100 iterations (for single precision complex matrix $A$) due to degraded numerical precision, while with some others we had to restart the iterative process about every 20 iterations. Only the BiCGStab method was usually faster than Lanczos/LU for our MS calculations; however, this method was found to lack stability for large matrix dimensions and many other Lanczos-type methods sometimes failed to converge.


[1] A. Gonis, *Green Functions for Ordered and Disordered Systems* (North Holland, Amsterdam, 1992).

[2] A. L. Ankudinov, B. Ravel, J. J. Rehr, and S. D. Conradson, Phys. Rev. B **58**, 7565 (1998).

[3] J. J. Rehr and R. C. Albers, Rev. Mod. Phys. **72**, 621 (2000).

[4] C. Lanczos, J. Res. Natl. Bur. Stand. **45**, 255 (1950); **49**, 33 (1952).

[5] F. J. Garcia de Abajo, M. A. Van Hove, and C. S. Fadley, Phys. Rev. B **63**, 075404 (2001).

[6] Gordon E. Moore, Electronics **38**, No. 8 (1965).

[7] J. J. Rehr and A. L. Ankudinov, J. Synchrotron Radiat. **8**, 61 (2001).

[8] M. H. Gutknecht, Acta Numer. **6**, 271 (1997).

[9] W. Gropp, E. Lusk, and A. Skjellum, *Using MPI: Portable Parallel Programming with The Message-Passing Interface* (MIT Press, Cambridge, 1994).

[10] For details about the code see the FEFF Project URL http://leonardo.phys.washington.edu/feff/

[11] H. A. Van der Vorst, SIAM (Soc. Ind. Appl. Math.) J. Sci. Stat. Comput. **13**, 631 (1992).

[12] C. Brezinski, M. Redivo-Zaglia, and H. Sadok, J. Comput. Appl. Math. **123**, 241 (2000).

[13] R. Haydock, Phys. Rev. B **49**, 10845 (1994); J. Phys. C **18**, 2235 (1985).

[14] A. Filipponi, J. Phys.: Condens. Matter **3**, 6489 (1991).

[15] D.D. Vvedensky, D.K. Saldin, and J.B. Pendry, Surf. Sci. **156**, 845 (1985).

[16] Huasheng Wu and S. Y. Tong, Phys. Rev. B **59**, 1657 (1999).

[17] T. Fujikawa, J. Phys. Soc. Jpn. **62**, 2155 (1993); T. Fujikawa, K. Nakamura, S. Nagamatsu, and J. J. Rehr, *ibid.* **71**, 357 (2002).

[18] F. Farges, G. E. Brown, Jr., and J. J. Rehr, Phys. Rev. B **56**, 1809 (1997).

[19] W. H. Press, et al., *Numerical Recipes in FORTRAN* (Cambridge University Press, Cambridge, 1992), p. 77.

[20] A. P. Hitchcock, T. Tyliszczak, P. Aebi, J. Z. Xiong, T. K. Sham, K. M. Baines, K. A. Mueller, X. H. Feng, J. M. Chen, B. X. Yang, Z. H. Lu, J. M. Baribeau, and T. E. Jackman, Surf. Sci. **291**, 349 (1993); J. C. Aubry, T. Tyliszczak, A. P. Hitchcock, J.-M. Baribeau, and T. E. Jackman, Phys. Rev. B **59**, 12 872 (1999).

[21] A. M. Flank (private communication).

[22] B. Ravel, Ph.D. thesis, University of Washington, Seattle, 1999.

[23] A. Poiarkova and J. J. Rehr, Phys. Rev. B **59**, 948 (1999).

[24] K. C. Jea and D. M. Young, Linear Algebr. Appl. **52/53**, 399 (1983).