Parallel Density Functional Theory Energies using the Fourier Transform Coulomb Method[†]

Jon Baker,[‡,§] László Füsti-Molnar,[§] and Peter Pulay*[,§]

Parallel Quantum Solutions, 2013 Green Acres Road, Suite A, Fayetteville, Arkansas 72703, and Department of Chemistry, University of Arkansas, Fayetteville, Arkansas 72701

Received: September 30, 2003; In Final Form: November 7, 2003

The recently described Fourier transform Coulomb (FTC) algorithm for fast and accurate calculation of density functional theory (DFT) energies (Füsti-Molnar, L; Pulay, P. *J. Chem. Phys.* **2002**, *117*, 7827) has been parallelized. We present several calculations showing the speed and accuracy of our new parallel FTC energy code, comparing its performance with our standard all-integral DFT code. Although it still contains significant serial code, the parallel FTC algorithm is up to 6 times faster overall than our parallel all-integral algorithm, and well over an order of magnitude faster for computation of the Coulomb terms, with essentially no loss in accuracy. Proposed improvements should significantly increase these factors. The Coulomb energy in DFT energy calculations can now be computed *accurately* for large molecules and/or basis sets faster than the exchange-correlation energy.

Introduction

Density functional theory is now well established as the premier computational method in ab initio quantum chemistry. Density functional theory (DFT) in its modern form derives from the X_α method of Slater and Johnson,[1] and from the 1965 paper by Kohn and Sham.[2] Much of the progress since then, in both methodology and the development of new functionals, has originated in the physics community, and it was not until the early 1990s that quantum chemists began to look seriously at the method, a key paper at this time being the performance study of a number of different density functionals by Johnson, Gill, and Pople.[3] DFT was subsequently taken up in earnest by the quantum chemistry community, most notably by the Handy[4] and Ahlrichs[5] groups. The quantum chemists injected a degree of rigor into the field that had for the most part been lacking, and the introduction of DFT methods into well established ab initio program packages (such as GAUSSIAN,[6] CADPAC,[7] and TURBOMOLE[8]) brought DFT into the mainstream of computational chemistry, contributing significantly to its current popularity.

In their initial DFT implementations quantum chemists leveraged the years of experience that had gone into the then existing Hartree−Fock-based codes, producing programs that treated DFT essentially as a variant to standard self-consistent Hartree−Fock theory. The exchange-correlation energy and the corresponding contributions to the Fock matrix were calculated numerically over atom-centered grids,[9] similarly to DFT codes developed by physicists, but the Coulomb term was evaluated as in Hartree−Fock theory, i.e., by calculating analytically all the necessary two-electron integrals. This is in marked contrast to DFT codes developed in the physics community, in which a variety of alternative methods were used, such as density fitting—also called "resolution of the identity" (RI-DFT)—an old favorite of DFT researchers,[10] which involves expansion of the density in an auxiliary basis set (used, for example, in both DGAUSS[11] and DEMON[12]), or direct numerical solution of the Poisson equation (a method adopted at least partially in ADF[13] and fully in DMOL[14]). The two-electron integral approach is especially useful for hybrid functionals, which retain a portion of the exact Hartree−Fock exchange.

These alternative methods of evaluating the Coulomb potential are usually faster than computing analytically all the two-electron integrals, but in many programs speed was achieved at the expense of accuracy, and total energies computed with these alternative approaches, even for small molecules, were often significantly different from energies calculated using more traditional quantum chemistry codes, with errors typically in the mE_h range. This degree of error approaches the predictive accuracy of the method, and makes it difficult to compare calculations using different programs. Numerical and other errors in the energy naturally extend to gradients, and some of the early DFT programs had difficulties reliably converging molecular geometries during a geometry optimization, especially for larger, floppy molecules. One of the aims behind the Fourier transform Coulomb (FTC) method was to achieve the speed of these alternative approaches without the reduction in accuracy.

The essence of the FTC method is the evaluation of the Coulomb operator and its matrix elements in an intermediate plane wave basis, retaining the traditional Gaussian basis sets. This approach has been pioneered by Parrinello and co-workers under the acronym GAPW (Gaussian and augmented plane wave), initially for pseudopotential methods,[15] and subsequently for all-electron calculations.[16] The FTC method shares the goals and the basic idea behind the GAPW approach, but is quite different technically, particularly in the treatment of core orbitals.

Plane wave basis sets are widely used in solid-state physics, especially for calculations on periodic systems, which have been their main applications. At first sight, they would appear to be entirely unsuitable for molecular chemistry as they have a number of obvious disadvantages: (1) they necessarily describe an infinite periodic system, and the periodic "ghost" images cause errors for molecules; (2) they lead to divergent expressions

[†] Part of the special issue "Fritz Schaefer Festschrift".
[‡] Parallel Quantum Solutions.
[§] University of Arkansas.

10.1021/jp036926l CCC: $27.50 © 2004 American Chemical Society
Published on Web 01/16/2004

for charge distributions with net charge; (3) plane waves can handle compact charge distributions, e.g., the region around the nucleus, only at very high cost; (4) good-quality plane wave basis sets are huge (our calculations typically use several million plane waves); (5) the large dimensions can cause severe problems with the determination of the eigenvalues and eigenvectors of the Fock matrix. However, they have one overwhelming advantage, namely, that calculation of the Coulomb integrals is greatly simplified and can be achieved very rapidly. The electron–electron Coulomb operator is diagonal in momentum space, and quantities can be calculated very efficiently in the appropriate (momentum or real space) representation, using fast Fourier transformation to switch between the representations.

In the past, the most prominent plane wave applications have been Car–Parrinello molecular dynamics (CPMD) and solid-state DFT, both with pseudopotentials and at a limited level of accuracy. CPMD takes optimum advantage of plane waves. Pseudopotentials take care of the region around the atomic nuclei, and because high accuracy is not needed, the periodic images can be neglected. No diagonalization is required because the wavefunction is propagated, and calculation of the forces is very fast. Solid-state DFT is also appropriate as space-filling structures can use the plane wave basis efficiently and periodic images do not have to be eliminated.

How can plane waves be used efficiently in accurate, all-electron molecular calculations? The essential idea is to start with a normal Gaussian-type basis set, expand the *valence* part of the density in plane waves,[16] compute the Coulomb potential in the plane wave basis, and then transform back to the original Gaussian basis when the Fock matrix elements are formed. The core contributions have to be calculated separately, either by a standard integral code or by some other method. Switching to a Gaussian basis removes problems due to the enormous dimension of the plane wave basis, as all matrices are represented in the modest Gaussian space, the dimension of which is at most a few thousand. Only one-dimensional quantities need to be stored and handled in the plane wave basis. In previous work we have shown how to circumvent *all* of the disadvantages of plane waves; in particular we have developed a simple technique which *exactly* eliminates the periodic ghost images and removes the divergences present in charged systems.[17,18]

Overview of the FTC Method

Before discussing the parallel algorithm, we provide a brief overview of the method, concentrating on the computational steps involved. Full technical details will not be given, as most of these have been discussed in the original presentation of the method.[19]

As noted above, the basic idea is to expand as much of the original basis set as possible in terms of plane waves, compute the electronic density on a direct-space grid, transform to momentum space to compute the Coulomb potential, transform back to the real space grid, and determine the Fock matrix elements (in the space of the original Gaussian basis set) by numerical quadrature. In general the original basis cannot be expanded *fully* in plane waves as basis functions with large exponents (those representing the core region) cannot be represented sufficiently accurately in a plane wave basis.

We first partition the original Gaussian basis set into two classes depending on the exponent value; those with small exponents (which we term *diffuse*, d) and those with large exponents (which we term *compact*, c). The exponent cutoff depends somewhat on the angular momentum (i.e., s, p, d, etc.) of the basis function, and also on the quality of the grid (i.e., the number of plane waves in the expansion) but is around 2.5–3.0 a_0^{-2}. Partitioning the basis in this way results in the following classes of integrals that need to be evaluated (using the Mulliken notation):

$$(1)\ (\mathrm{cc|cc});\ (2)\ (\mathrm{cc|cd});\ (3)\ (\mathrm{cd|cd});$$
$$(4)\ (\mathrm{cc|dd});\ (5)\ (\mathrm{cd|dd});\ (6)\ (\mathrm{dd|dd})$$

Compact basis functions cannot be properly expanded in plane waves, so we treat the first four integral types using a variant of our standard integral package, i.e., essentially in exactly the same way as in a "normal" SCF integral code. Integrals of type 6 can be fully handled in plane wave space, as can those of type 5 (because the high momentum components of the charge density, cd, do not interact with the diffuse charge density, dd). In theory, integrals of type 4 can also be taken over into plane waves, but we have not yet done this.

The molecule is placed in a box sufficiently large to contain essentially all the electron density. For simplicity, the box can be considered as a cube of sides L, but in the actual program it is a parallelepiped, adapted to the molecular dimensions. We introduce a standard rectangular grid in our box, whose grid density, d—the number of plane waves in one Cartesian direction per atomic unit—characterizes the plane wave basis. The grid spacing is $h = d^{-1}$, and the grid points range from $-L/2$, $-L/2 + h$, ..., to $L/2 - h$ in each Cartesian direction. The efficiency of Fourier transform and plane wave methods derives from the fast Fourier transform (FFT), which allows almost effortless switching between the momentum and coordinate representations. For quantities which can be exactly represented by the plane wave basis, the two descriptions are isomorphic.[20]

Evaluation of the Coulomb potential in plane waves is extremely fast, so we want as many basis functions as possible to be partitioned into the plane wave part of our space. Normally integrals of types 5 and 6 will dominate, especially in larger basis sets containing high angular momentum and diffuse functions.

Having partitioned the basis set, the steps followed to compute the Coulomb Fock matrix elements during each SCF cycle are as follows.

(1) For integrals of types 1–4, determine the Fock matrix elements in the traditional way, i.e., compute the integrals and contract them with the appropriate density matrix elements. This is currently done using a minor variant of our standard direct SCF code. However, as discussed later, most contributions from these integrals can be calculated very efficiently using a multipole approximation.

(2) Compute the Coulomb contribution arising from the (dd|dd) integrals. This involves the following steps (order with respect to system size is shown in parentheses): (i) calculation of the "diffuse" density on the real space grid, $O(N)$, i.e., at each grid point ($\mathbf{r}$), $\rho(\mathbf{r}) = \sum \mathbf{d}_{\alpha\beta} g^{\mathrm{d}}_{\alpha}(\mathbf{r})\, g^{\mathrm{d}}_{\beta}(\mathbf{r})$, $\mathbf{d}$ = density matrix; (ii) FFT to momentum space, $O(N \log N)$; (iii) calculation of the potential in momentum space, $O(N)$; (iv) reverse FFT back to real space, $O(N \log N)$; (v) computation of Fock matrix elements by numerical quadrature, $O(N)$.

(3) Compute the Coulomb contribution arising from the (cd|dd) integrals. This involves essentially the same steps as in (2), above, except that the "mixed" density $\rho(\mathbf{r}) = \sum \mathbf{d}_{\alpha\beta} g^{\mathrm{c}}_{\alpha}(\mathbf{r})\, g^{\mathrm{d}}_{\beta}(\mathbf{r})$ is constructed, where one of the indices (α) corresponds to a compact Gaussian function. In practice several steps from (2) and (3) are combined in the interest of program efficiency.

As indicated in (2), the scaling of the various steps is either $O(N)$ or $O(N \log N)$ for the Fourier transform steps. In practice, the FFT steps are very fast; we use the excellent FFTW package of Frigo and Johnson.[21] The rectangular nature of the grid allows for very efficient screening and precomputation of many quantities.

The exchange-correlation part of the calculation is currently handled in the same way as in our integral-driven DFT code, i.e., by numerical quadrature over spherical, atom-centered grids, using techniques pioneered in this context by Becke.[9] We have already improved this part of the code (see, e.g., ref 22), but we plan a further overhaul with a different quadrature allowing us to use much of the grid already generated and in place for the Coulomb term. We do not discuss the exchange-correlation term in detail in this paper, our main focus being parallelization of the code used to compute the Coulomb Fock matrix elements via our new FTC method.

Parallelization

Our initial parallelization of the FTC code has been completed using the parallel virtual machine (PVM) toolkit.[23] Parallelization can also be accomplished using the message-passing interface (MPI), and indeed our standard parallel code has an MPI version, although the PVM variant is somewhat more convenient to use, particularly for explicitly correlated methods such as second-order Møller−Plesset (MP2) theory. Below we discuss our parallelization strategy for each stage of the FTC method.

A. Traditional Electron Repulsion Integrals. Computation of those integrals (classes 1−4, above) that are currently evaluated classically is handled in a similar way as in our existing, all-integral code. Each possible shell *pair* (initially in blocks of 20−50, later individually to ensure good load balance) is passed in a round-robin fashion to each slave, and all integrals associated with that shell pair are computed, contracted with the appropriate density matrix elements, and added directly to a local copy of the Fock matrix. The parallel efficiency of this part of the FTC code is very high.

B. Precomputation of Compact Basis Function Values over the Real Space Grid. These values are precomputed and written to disk. Each compact basis function has only a limited spatial extent, and values are stored only over those grid points which have a "nonzero" contribution. Parallelization is carried out over compact shells, which are passed to each slave in a round-robin fashion. Values are written to disk on the slave that computed them. Parallel efficiency is very high, as only the shell indices need to be communicated, and the amount of disk storage required is typically less than 1 GB (in total, and distributed over all nodes). Each slave knows which compact shell indices are stored on its disk, so no additional communication is required during the SCF procedure once the basis function values have been calculated and stored. The entire process has recently been dramatically improved using a new "core-cut" procedure.[24]

C. Calculation of the Diffuse Density on the Real Space Grid. This is parallelized by selecting one grid dimension (X in our case), passing the grid points along this axis to each slave in a round-robin fashion, and computing the density over all Y, Z grid points. The computed densities are passed back and accumulated on the master. As an example, for the yohimbine molecule ($C_{21}H_{26}N_2O_3$) the box dimensions, L, are approximately 45 au; using our standard grid density of 3.75 points/au, this results in 169 grid points in each direction, equivalent to a total of $169^3 \approx 4.8$ million plane waves. The 169 grid points in the X direction are passed to each slave, either in small blocks or one at a time, and the densities on all Y, Z grid points ($169^2 = 28561$ for each X value) are calculated and passed back to the master.

Despite the large number of plane waves (typically between two and six million), the memory requirements for this step are modest. The master requires storage for the complete grid (16−48 MB), but the slaves only need storage for the number of Y, Z grid points (times the number of X values being computed at the same time). The only real communication between the nodes is during the transfer of the computed density; consequently the parallel efficiency of this step is also high.

D. Calculation of the Diffuse Coulomb Potential by Fast Fourier Transform. This step involves the FFT of the diffuse density to momentum space, calculation of the Coulomb potential in momentum space, and the FFT back to real space. As previously discussed, we use the serial version of the FFTW package of Frigo and Johnson[21] essentially "as is", and we have made no attempt to parallelize this code at this time. Note that this step also includes the code needed to eliminate the periodic images,[17] which involves a doubling of the box dimensions and takes significantly more time than the FFT itself. The FFT step is very fast, but it *is* serial in the current implementation, and it contributes to the serial overhead of our parallel FTC code. We plan to parallelize this step later.

E. Calculation of the Mixed Density on the Real Space Grid and the Fock Matrix Contribution from the Compact Functions. Parallelization of the mixed density calculation cannot be accomplished in the same way as for the diffuse density, because the compact orbitals cannot be handled in plane wave space (i.e., via the Fourier transform). Instead a full "partial" mixed density is formed on each slave using the precomputed core functions that are stored on that slave, and the total mixed density is accumulated on the master using the operation *pvmfreduce*. This requires each slave to hold a local copy of the full grid containing its contribution to the density. These grids are collected and summed on the master. At the same time, each slave computes the direct contribution to the Fock matrix from its own partial mixed density and the precomputed compact basis functions. This requires the Coulomb operator over the full grid which was obtained from the diffuse density via the FFT in step D, above.

At the start of this step, the diffuse Coulomb operator over the full grid is broadcast from the master to each slave, and at the end the mixed density is accumulated on the master. This stage is the most computationally demanding part of the current FTC algorithm. Each slave requires storage for two full grids (16−48 MB each), plus Fock and density matrixes, as well as other work arrays, and there is considerable data transfer over the nodes, both in the initial broadcast (which, unfortunately, is not a true broadcast in the current version of PVM, and therefore, the time taken increases linearly with the number of slaves) and, especially, in the final accumulation of the mixed density on the master. On the other hand, the data transfer is only about the same as that required to accumulate a large Fock matrix (say 2500 basis functions, 50 MB) from each slave to the master, which has to be done in any case. Overall the parallel efficiency of this step is only moderate; there is a speedup, but it is likely to fall off quite rapidly with an increasing number of slaves due to the relatively high communication overhead compared to the rest of the FTC code. Possible improvements in this part require the division of the compact basis functions into sets localized in different regions of the molecule, and assigning each slave one of these regions. In this case, the diffuse

TABLE 1: Timing Data for a Single-Point BLYP Energy for Aspirin ($C_9H_8O_4$)

	elapsed time			
job step	basis 1 (306/233)[a]		basis 2 (555/482)	
number of processors	2	4	2	4
A. classical integrals	49 s	26 s	316 s	161 s
C. diffuse density	12 s	6 s	64 s	33 s
D. FFT 1	24 s	25 s	42 s	42 s
E. mixed density + compact Fock	21 s	18 s	60 s	40 s
F. FFT 2	24 s	25 s	42 s	42 s
G. all diffuse Fock contributions	10 s	6 s	50 s	26 s
total FTC Coulomb	2.3 min	1.7 min	9.6 min	5.8 min
1-el integrals				
miscellaneous	0.1 min	0.1 min	0.4 min	0.3 min
exchange correlation (DFT)	1.2 min	0.6 min	7.3 min	3.9 min
total SCF time (FTC)	3.6 min	2.4 min	17.3 min	10.0 min
Coulomb only (all-integral)	3.4 min	1.7 min	68.2 min	33.3 min
total SCF time (all-integral)	4.7 min	2.4 min	75.9 min	37.5 min
all-integral/FTC (total job)	1.3	1.0	4.4	3.8
all-integral/FTC (Coulomb only)	1.5	1.0	7.1	5.7
SCF energy (all-integral)	−646.959156		−648.7196121	
SCF energy (FTC)	−646.959141		−648.7196124	
total error	0.000015		0.0000003	
error per atom	0.7 μE_h		0.014 μE_h	

[a] Basis set description (*xxx*/*yyy*): the first number refers to the total number of basis functions, the second to the number of diffuse functions (see the text).

Coulomb operator and the mixed density are needed only over a subset of the full grid.

F. Calculation of the Mixed Coulomb Potential by Fast Fourier Transform. This is essentially a repeat of step D, above, only using the mixed density computed on the slaves in step E. As with step D it is not parallel and contributes to the serial overhead.

G. Calculation of the Fock Matrix Contributions from the Diffuse Functions. This is the last part of the parallel FTC algorithm and involves calculating the contribution to the Fock matrix from the diffuse functions by numerical quadrature. The diffuse and mixed Coulomb potentials over the grid (derived from the FFT of the diffuse and mixed densities, respectively) are combined, and transmitted to the slaves. As in step C, this is parallelized over the X direction grid points in a round robin, with the potential over all Y, Z grid points sent with each X point. Each slave sums up the relevant contributions into its own local copy of the Fock matrix. The amount of data transmitted between the nodes is essentially the same as in step C, as are the memory requirements, and the parallel efficiency should be the same as well.

At the end of all these steps, each slave's local copy of the Fock matrix is summed up on the master.

Results

We demonstrate the capabilities of our new parallel FTC code on four molecules, aspirin ($C_9H_8O_4$), yohimbine ($C_{21}H_{26}N_2O_3$), paclitaxel (taxol) ($C_{47}H_{51}NO_{14}$), and chlorophyll ($C_{55}H_{72}N_4O_5$-Mg), and two basis sets, a valence triple-ζ basis with a single set of polarization functions for all non-hydrogen atoms (a double set for hydrogen) denoted VTZP[25] and a larger basis derived from the Pople-type 6-311G(2df,2pd) basis by uncontracting those primitive Gaussians which can be treated using the FTC method.[26] Henceforth, these will be designated basis 1 and basis 2, respectively. The modification of the Pople basis set was made because our current code is not yet able to calculate components of the same contraction with different algorithms (i.e., FTC and traditional integrals). This is a purely technical problem and will be addressed in future versions of our code. Comparisons are made between the all-integral, fully direct code in the current release of our PQS ab initio program[27] and our new FTC code.

Tables 1−4 give a detailed breakdown of the elapsed time spent in the various steps of the parallel FTC algorithm for aspirin, yohimbine, paclitaxel, and chlorophyll, run on two and four nodes, respectively, of one of our current systems (a QS8-2400C, using 2.4 GHz Xeon processors and 2 GB of PC2100 ECC memory per node), together with comparisons with the all-integral code. Note that timings for precomputing the compact basis functions over the real space grid (step B, above) are not given as this step is extremely fast. There are four main points to focus on: (1) the ratio between the time to compute the Coulomb term for the all-integral compared to the FTC code; (2) the ratio between the total SCF time for the two codes; (3) the relative accuracy of the FTC code; (4) the overall parallel efficiency of the FTC code.

We look first at the parallel efficiency. As an example, we take the timings for aspirin, basis 2 (our comments apply equally well to all the other calculations). Comparing the reported elapsed timings on two and four nodes, it is clear that the classical integral step, computation of the diffuse density, and the final Fock matrix construction are all very efficiently parallelized. As expected, the mixed density/compact Fock step does not parallelize well due to the high communication overhead (however, the parallel performance of this step improves as both the system and basis set size increases). As already noted, the FFT steps are not parallel, and this contributes significantly to the serial overhead. For aspirin, the effect is marked (on four processors the FFT time is 1.4 min out of a total FTC time of 5.8 min), but—as with the mixed density/compact Fock step—becomes less with increasing system and/

TABLE 2: Timing Data for a Single-Point BLYP Energy for Yohimbine ($C_{21}H_{26}N_2O_3$)

	elapsed time			
job step	basis 1 (702/546)[a]		basis 2 (1275/1119)	
number of processors	2	4	2	4
A. classical integrals	326 s	167 s	1973 s	1000 s
C. diffuse density	42 s	22 s	254 s	129 s
D. FFT 1	57 s	57 s	98 s	98 s
E. mixed density + compact Fock	58 s	42 s	165 s	106 s
F. FFT 2	57 s	57 s	98 s	98 s
G. all diffuse Fock contributions	37 s	21 s	208 s	110 s
total FTC Coulomb	9.7 min	6.1 min	46.7 min	25.8 min
1-el integrals	0.2 min	0.2 min	0.4 min	0.4 min
miscellaneous	0.6 min	0.6 min	3.1 min	3.0 min
exchange correlation (DFT)	6.1 min	3.2 min	45.9 min	23.4 min
total SCF time (FTC)	16.6 min	10.1 min	96.1 min	52.6 min
Coulomb only (all-integral)	22.5 min	11.3 min	474 min	239 min
total SCF time (all-integral)	29.4 min	15.3 min	523 min	266 min
all-integral/FTC (total job)	1.8	1.5	5.4	5.1
all-integral/FTC (Coulomb only)	2.3	1.9	10.1	9.3
SCF energy (all-integral)	−1148.494496		−1150.970847	
SCF energy (FTC)	−1148.494463		−1150.970849	
total error	0.000033		0.000002	
error per atom	0.6 μE_h		<0.04 μE_h	

[a] Basis set description (*xxx*/*yyy*): the first number refers to the total number of basis functions, the second to the number of diffuse functions (see the text).

TABLE 3: Timing Data for a Single-Point BVWN Energy for Paclitaxel ($C_{47}H_{51}NO_{14}$)

	elapsed time			
job step	basis 1 (1575/1214)[a]		basis 2 (2860/2499)	
number of processors	2	4	2	4
A. classical integrals	2342 s	1189 s	16088 s	8159 s
C. diffuse density	146 s	75 s	898 s	454 s
D. FFT 1	100 s	103 s	138 s	141 s
E. mixed density + compact Fock	166 s	107 s	570 s	307 s
F. FFT 2	100 s	102 s	137 s	140 s
G. all diffuse Fock contributions	139 s	78 s	742 s	394 s
total FTC Coulomb	50.0 min	27.7 min	310 min	160 min
1-el integrals	1.7 min	1.7 min	4.3 min	4.3 min
miscellaneous	6.8 min	6.8 min	30.1 min	31.5 min
exchange correlation (DFT)	28.7 min	14.7 min	211.6 min	108.5 min
total SCF time (FTC)	87.2 min	50.9 min	556 min	304 min
Coulomb only (all-integral)	163 min	80.8 min	3337 min	1675 min
total SCF time (all-integral)	200 min	104 min	3583 min	1819 min
all-integral/FTC (total job)	2.3	2.0	6.4	6.0
all-integral/FTC (Coulomb only)	3.3	2.9	10.8	10.5
SCF energy (all-integral)	−2945.727664		−2952.295540	
SCF energy (FTC)	−2945.727429		−2952.295571	
total error	0.000235		0.000031	
error per atom	2 μE_h		<0.3 μE_h	

[a] Basis set description (*xxx*/*yyy*): the first number refers to the total number of basis functions, the second to the number of diffuse functions (see the text).

or basis set size (e.g., for paclitaxel, basis 2, the FFT time is just 4.7 min out of a total FTC time of 160.1 min on four processors).

Overall, the parallel efficiency of our new FTC code is reasonably satisfactory, especially for the small-scale parallelism that is our primary focus (i.e., systems with 4−16 processors). However, for good efficiency on larger numbers of processors it is clear that the FFT steps need to be parallelized, and the communication overhead in the calculation of the mixed contributions needs to be reduced.

The relative performance of the FTC code compared to our standard all-integral code can be seen by examining the all-integral/FTC timing ratios reported in Tables 1−4. For the total job time, on four processors, these range from 1.0 for aspirin, basis 1 (i.e., the FTC algorithm is the same speed as the all-integral code), to 6.0 (i.e., 6 times faster) for paclitaxel, basis

TABLE 4: Timing Data for a Single-Point OLYP Energy for Chlorophyll ($C_{55}H_{72}N_4O_5Mg$)

	elapsed time			
job step	basis 1 (1826/1429)[a]		basis 2 (3309/2912)	
number of processors	2	4	2	4
A. classical integrals	2654 s	1373 s	13221 s	6753 s
C. diffuse density	193 s	102 s	966 s	487 s
D. FFT 1	239 s	246 s	272 s	275 s
E. mixed density + compact Fock	204 s	154 s	461 s	279 s
F. FFT 2	240 s	245 s	266 s	268 s
G. all diffuse Fock contributions	177 s	102 s	821 s	445 s
total FTC Coulomb	62.0 min	37.2 min	267 min	142 min
1-el integrals	2.6 min	2.7 min	6.7 min	6.7 min
miscellaneous	10.9 min	11.3 min	60.8 min	61.7 min
exchange correlation (DFT)	36.5 min	20.1 min	186.5 min	97.6 min
total SCF time (FTC)	112 min	71.3 min	521 min	308 min
Coulomb only (all-integral)	186 min	90.5 min	3208 min	1593 min
total SCF time (all-integral)	236 min	121 min	3462 min	1759 min
all-integral/FTC (total job)	2.1	1.7	6.6	5.7
all-integral/FTC (Coulomb only)	3.0	2.4	12.0	11.2
SCF energy (all-integral)	−2928.898581		−2934.349474	
SCF energy (FTC)	−2928.898554		−2934.349454	
total error	0.000027		0.000020	
error per atom	<0.2 μE_h		<0.15 μE_h	

[a] Basis set description (*xxx*/*yyy*): the first number refers to the total number of basis functions, the second to the number of diffuse functions (see the text).

2. For the Coulomb term alone, the speedup is well over an order of magnitude for the larger systems. For smaller systems with modest basis sets, there are no real advantages in using the current FTC code, but as the system size and, especially, the number of basis functions increase, the advantage of the FTC method increases.

A further advantage of the FTC method is its excellent scaling with increasing basis set size at constant system size, a property not shared by many other fast DFT codes. For example, the continuous fast multipole method developed by Gill and Head-Gordon,[28] and available in the recent releases of GAUSSIAN[6] and Q-CHEM[29] programs, is dominated for larger systems by the calculation of the near-field integrals[30] which have a steep $O(N^4)$ scaling. The calculation of the Coulomb matrix elements in the density fitting (DF or RI-DFT) method scales formally only as $O(N^2)$, but the calculation of the fitting coefficients involves a cubic $O(N^3)$ step. Methods based on the numerical solution of the Poisson equation should scale in principle quadratically. However, our own experience, as well that of others,[31] shows that it is difficult if not impossible to achieve high accuracy with these methods. With the FTC code the scaling is essentially quadratic, as can be seen from Table 1. For example, for aspirin, increasing the number of basis functions from 306 (basis 1) to 555 (basis 2) on two nodes increases the Coulomb evaluation time by a factor of around 4 (less if we allow for the serial FFT overhead) with the FTC code, but by over 20 with the all-integral code. (This latter factor is even greater than expected from the $O(N^4)$ scaling, reflecting the increased integral evaluation time due to the greater number of high angular momentum basis functions in the larger basis.) Essentially the same scaling is seen for the other three molecules as well (Tables 2−4). If the time spent evaluating the remaining classical integrals is removed, then the plane wave manipulations in the FTC algorithm scale almost linearly with increasing basis size.

Also shown in the tables are total SCF energies, the total error in the FTC method (assuming that our standard all-integral code is exact), and the error per atom. Total errors are on the order of a few tens of μE_h, corresponding to well under 1 μE_h per atom, in all cases except for paclitaxel, basis 1, where the total error is 0.2 mE_h with an error per atom of 2 μE_h. Perhaps surprisingly, the errors are smaller with the larger basis set. The main source of error is in selecting the cutoff for partitioning the basis into diffuse and compact functions; the more functions there are in the diffuse partition that have exponents close to the cutoff, the greater the error. In basis 1, the oxygen atom has a p-function with an exponent (2.28) which is close to the default cutoff, and this shows the maximum error for paclitaxel, which has the most oxygen atoms. The errors for all molecules, and especially paclitaxel, can be further reduced—at the expense of a slight increase in job time—by moving this function into the compact set. Note that errors in the Coulomb term with the FTC method for these systems are typically 1 or 2 orders of magnitude *less* than the integration error in the DFT exchange-correlation energy.

The favorable scaling properties of the FTC method, for both increasing basis set size and increasing system size, are further demonstrated in Tables 5 and 6, which present timings for a series of BLYP calculations on various alanine chains, (alanine)$_n$. Table 5 presents single-point energies for (alanine)$_5$ with seven basis sets, ranging from 6-31++G (419 basis functions) to 6-311++G(3df,3pd) (1500 basis functions). Table 6 presents single-point energies with the 6-311++G** basis set for five systems, from a single alanine (181 basis functions) through (alanine)$_{15}$ (2211 basis functions). The first series of calculations shows the scaling with increasing basis set size at constant system size, while the second shows the scaling with increasing system size at constant basis set quality. All jobs were run on eight processors of one of our QS8-2400C QuantumCube systems, using 2.4 GHz PIV Xeon CPUs on dual-processor motherboards, running both processors per node.

In both Tables 5 and 6 the first row for a given basis set/alanine chain gives timings for the standard all-integral code, while the second row gives timings with the FTC code. In the

TABLE 5: Timings (min) for Single-Point BLYP Energies for $(\mathrm{Alanine})_5$

basis	nbf	2-el	FTC	DFT	misc	total	E
6-31++G	419	11.4		6.8	0.2	18.4	−1312.2239303
		5.6	14.7	6.8	0.2	27.3	−1312.2239301
6-31++G*	575	29.1		9.7	0.5	39.3	−1312.5948321
		8.2	14.6	9.7	0.5	33.0	−1312.5948321
6-31++G**	656	37.0		11.5	0.6	49.1	−1312.6462435
		9.2	15.1	11.5	0.6	36.4	−1312.6462437
6-311++G**	761	62.4		17.5	1.0	80.9	−1312.9701514
		9.9	14.7	17.5	1.0	43.1	−1312.9701504
6-311++G(2d,2p)	972	150.3		26.9	1.7	178.9	−1313.0096019
		18.3	17.4	26.7	1.5	63.9	−1313.0096014
6-311++G(2df,2pd)	1289	399.0		34.9	3.8	437.7	−1313.0477576
		29.3	16.8	34.9	3.7	84.7	−1313.0477573
6-311++G(3df,3pd)	1500	869.2		62.8	5.4	937.4	−1313.0611576
		105.6	25.5	63.3	5.5	199.9	−1313.0611569

TABLE 6: Timings (min) for Single-Point BLYP/6-311++G Energies for $(\mathrm{Alanine})_n$**

system	nbf	2-el	FTC	DFT	misc	total	E
alanine	181	0.60		0.37	0.02	0.99	−323.7469289
		0.21	6.58	0.35	0.02	7.16	−323.7469290
$(\mathrm{alanine})_2$	326	3.51		1.24	0.10	4.85	−571.0526737
		0.78	8.18	1.23	0.09	10.28	−571.0526738
$(\mathrm{alanine})_5$	761	62.4		17.5	1.0	80.9	−1312.9701514
		9.9	14.7	17.5	1.0	43.1	−1312.9701504
$(\mathrm{alanine})_{10}$	1486	537.8		88.3	6.4	632.5	−2549.4617713
		73.6	32.6	89.1	6.4	201.7	−2549.4617682
$(\mathrm{alanine})_{15}$[a]	2211	2696.9		312.2	26.9	3036.0	−3785.9364673
		224.4	78.4	204.1	24.6	531.5	−3785.9364604

[a] The all-electron calculation (first row) took more SCF cycles to converge than the FTC calculation for $(\mathrm{alanine})_{15}$; for all other systems the number of SCF cycles was the same in each case.

latter case, the timing under the "2-el" column refers to the timing for step A, i.e., those integrals that are handled classically, while the timing under the "FTC" column refers to all other steps in the FTC code. The timing in the "total" column refers to the total elapsed time for the complete SCF calculation.

Turning first to Table 5, we see that, for $(\mathrm{alanine})_5$, for the smallest basis set (6-31++G), the FTC code is *slower* than the all-integral code, but for all other basis sets the FTC code is faster, with the speedup increasing with increasing basis set size, becoming over 5 times faster with the 6-311++G(2df,2pd) basis. With the largest 6-311++G(3df,3pd) basis, the speedup falls off to around 4.7; this is primarily due to paging as the memory capacity of the system has been reached. If one considers just the time to compute the Coulomb term (the sum of the timings in the "2-el" and "FTC" columns), the speedup is of course even greater. From the timings given in Table 5 the various steps scale with increasing basis set size according to the following powers: DFT, 1.46; miscellaneous, 2.62; classical integrals (all-electron), 3.16; classical integrals (FTC), 1.47. In the FTC calculation, the FTC part itself is almost independent of basis set size. (There is a slight increase with increasing basis size, but the scaling is clearly sublinear; however, this is certainly influenced by the degree of parallelism in the FTC code.) The overall scaling for the all-electron code is 2.62, while the FTC code clearly shows linear scaling.

The situation is similar in Table 6. For small alanine chains (one and two alanines) the FTC code is again slower than the all-integral code, but by the time we reach five alanines (and certainly before this) the FTC code is faster, with the speedup increasing with increasing system size (over a factor of 3 for $(\mathrm{alanine})_{10}$ and almost a factor of 6 for $(\mathrm{alanine})_{15}$, although part of the latter speedup is due to the decrease in the number of SCF cycles). The overall scaling is 3.2 for the all-electron code, while the FTC code shows quadratic scaling. However, the scaling of the plane wave part of the FTC code is itself near-linear.

In considering the results shown in Tables 5 and 6, it should be borne in mind that the basis set contains diffuse functions, which in most other methods negatively influence scaling properties because of their relatively large spatial extent. Additionally, the timing comparison is unfavorable for the FTC code, as the all-integral code has a greater parallel efficiency. As already discussed, the FFT part of the plane wave code is currently not parallel, and this is a significant overhead, especially for smaller systems and basis sets. The nonparallel overhead is the main reason the FTC scaling is sublinear in Table 1. For a single alanine molecule (Table 6), the time taken for the FTC step (under the "FTC" column) is dominated by the FFT, which comprises 73% of the total; for $(\mathrm{alanine})_{15}$ the FFT time falls to 25% of the total, noticeably less (but still significant).

Tables 5 and 6 also further demonstrate the accuracy of the FTC method (if anything even more so in this regard than Tables 1−4). The difference between the computed all-integral and FTC *total* energies is in the μE_h or sub-μE_h range. For example, for $(\mathrm{alanine})_{10}$ the difference is 3 μE_h, which is less than 0.03 μE_h per atom.

Conclusions

Our previous serial FTC algorithm[19] has been parallelized. Despite the fact that the Fourier transform step is still serial in the current parallel algorithm, the parallel efficiency in the plane wave part is reasonably good, particularly for larger molecules and basis sets, where the FFT step is relatively unimportant in terms of the elapsed time. The parallel FTC code is up to 6 times faster than our standard all-integral code for computing DFT energies, and well over an order of magnitude faster for computing the Coulomb term. Unlike several other fast DFT codes using alternative approaches, the speedup over the all-integral code is achieved with essentially *no loss in accuracy*. The scaling properties of the method are very favorable, especially for increasing basis set size at constant system size, where it is genuinely near-linear; furthermore, gains can be realized even for small molecules (e.g., aspirin, as shown in this work).

The FTC method shows great promise. Extension to gradients is straightforward (and is currently under way[24]), and further extensions to analytical second derivatives (and hence vibrational frequencies) and NMR chemical shifts are planned. The efficiency of the method, relative to the traditional all-integral programs, should be even higher for these derivative techniques, as they implicitly extend the basis set, and can profit from the advantageous scaling of FTC with increasing basis set size. As noted, the current FTC code is many times faster than the corresponding all-integral code, and there are several obvious improvements that can be made, such as parallelizing the FFT steps and handling the (cc|dd) integrals in plane wave space.

The most important potential improvement concerns those integrals that are evaluated classically in the current FTC algorithm. Although they comprise only a few percent of the total, they typically take more time than all remaining FTC steps combined. Omitting the (cc|dd) integrals discussed above, these integrals are of the form (ca|c′a′), where a and a′ denote any basis function. In other words, these integrals involve a compact function for each electron. Such charge distributions are compact, even if a or a′ is diffuse. Consequently, most of the remaining classical integrals are highly amenable to evaluation via a multipole expansion, and we estimate that implementing this could reduce the current classical integral evaluation time by an order of magnitude or more for most systems. Work is currently in progress on a multipole-based integral evaluation algorithm.

The above-mentioned improvements will further increase the performance of the FTC code relative to our all-integral SCF code, making it the method of choice for all but the smallest of systems. Once all these changes are in place, the time required to compute the Coulomb energy should be noticeably less than that required to evaluate the exchange-correlation energy. Additionally, we are already seeing that for larger systems (such as chlorophyll, Table 4) miscellaneous serial overhead during the SCF procedure (principally diagonalization of the Fock matrix) is significantly impacting the overall job time. For further reductions in elapsed parallel job times for DFT energy calculations on large systems, these non-Coulomb issues clearly need to be addressed.

One further point that should be noted is that the FTC method as outlined in this paper—as with all other fast-DFT methods—achieves its speedup over traditional all-integral code by dramatically reducing the time needed to evaluate the Coulomb term. The standard Hartree–Fock exchange term, which is a component in all of the so-called hybrid DFT functionals (e.g., B3LYP) is unaffected. Consequently the method as currently formulated cannot be used to great advantage with hybrid functionals, only with “pure” density functionals which contain no “exact” exchange. The HF exchange term is much more “local” than the Coulomb term (it has been shown to scale linearly with system size in insulators[32]), so the FTC method should be substantially less expensive than current all-integral techniques even with hybrid functionals for very large systems, provided code currently used to evaluate the HF exchange term is suitably modified (e.g., using appropriate cutoffs after removal of the Coulomb contribution). It could also reduce the computational cost in other ways; for example, geometries can be preoptimized with less expensive pure DFT methods, switching to the hybrid functional only in the last few cycles.

Acknowledgment. This work was supported by the U.S. National Science Foundation via an SBIR Phase I award (DMI-0214787) to Parallel Quantum Solutions, and by Grant CH-0111101 to the University of Arkansas, Fayetteville.

References and Notes

(1) Slater, J. C. *Phys. Rev.* **1951**, *81*, 385. Johnson, K. H. *J. Chem. Phys.* **1966**, *45*, 3085.
(2) Kohn, W.; Sham, L. J. *Phys. Rev.* **1965**, *A140*, 1133.
(3) Johnson, B. G.; Gill, P. M. W.; Pople, J. A. *J. Chem. Phys.* **1993**, *98*, 5612.
(4) Murray, C. W.. Handy, N. C.; Laming, G. J. *Mol. Phys.* **1993**, *78*, 997.
(5) Treutler, O.; Ahlrichs, R. *J. Chem. Phys.* **1995**, *102*, 346.
(6) Frisch, M. J.; Trucks, G. W.; Head-Gordon, M.; Gill, P. M. W.; Wong, M. W.; Foresman, J. B.; Johnson, B. G.; Schlegel, H. B.; Robb, M. A.; Replogle, E. S.; Gomperts, R.; Andres, J. L.; Raghavachari, K.; Binkley, J. S.; Gonzalez, C.; Martin, R. L.; Fox, D. J.; Defrees, D. J.; Baker, J.; Stewart, J. J. P.; Pople, J. A. *Gaussian 92*; Gaussian, Inc.: Pittsburgh, PA, 1992.
(7) CADPAC5: Amos, R. D.; Alberts, I. L.; Andrews, J. S.; Colwell, S. M.; Handy, N. C. Jayatilaka, D.; Knowles, P. J.; Kobayashi, R.; Koga, N.; Laidig, K. E.; Maslen, P. E.; Murray, C. W.; Rice, J. E.; Sanz, J.; Simandiras, E. D.; Stone, A. J.; Su, M.-D., Cambridge University, Cambridge, U.K., 1992.
(8) Ahlrichs, R.; Bär, M.; Häser, M.; Horn, H.; Kölmel, C. *Chem. Phys. Lett.* **1989**, *162*, 165.
(9) Becke, A. D. *J. Chem. Phys.* **1988**, *88*, 2547.
(10) Dunlap, B. I. *Phys. Rev.* **1990**, *A42*, 1127.
(11) Andzelm, J.; Wimmer, E. *J. Chem. Phys.* **1992**, *96*, 1280.
(12) St-Amant, A.; Salahub, D. R. *Chem. Phys. Lett.* **1990**, *169*, 387.
(13) Ravenek, W.; Baerends, E. J. *J. Chem. Phys.* **1984**, *81*, 865.
(14) Delley, B. *J. Chem. Phys.* **1990**, *92*, 508.
(15) Lippert, G.; Hutter, J.; Parrinello, M. *Theor. Chim. Acta* **1999**, *103*, 124.
(16) Krack, M.; Parrinello, M. *Phys. Chem. Chem. Phys.* **2000**, *2*, 2105.
(17) Füsti-Molnar, L.; Pulay, P. *J. Chem. Phys.* **2002**, *116*, 7795.
(18) Hockney, R. W. In *Plasma Physics*; Alder, B., Fernbach, S., Rothenberg, M., Eds.; Methods of Computational Physics, Vol. 9; Academic: New York, 1970; p 136.
(19) Füsti-Molnar, L.; Pulay, P. *J. Chem. Phys.* **2002**, *117*, 7827.
(20) Light, J. C.; Hamilton, I. P.; Lill, J. V. *J. Chem. Phys.* **1985**, *82*, 1400.
(21) Frigo, M.; Johnson, S. G. *Proceedings of the 1998 ICASSP Conference*; Casual Productions Pty Ltd.: Adelaide, Australia. Vol. 3, p 1381. www.causalproductions.com.
(22) Mitin, A. V.; Baker, J.; Wolinski, K.; Pulay, P. *J. Comput. Chem.* **2003**, *24*, 154.
(23) PVM, a message-passing toolkit developed by Oakridge National Laboratory and Tennessee State University.
(24) Füsti-Molnar, L. *J. Chem. Phys.* **2003**, *119*, 11080.
(25) Our VTZP basis is loosely derived from Dunning's correlation-consistent cc-pVTZ basis set. It is 3s2p on H, 4s3p1d on C, N, and O, and 6s5p1d on Mg.
(26) Slightly decontracted 6-311G(2df,2pd) basis. In the Pople notation, it is 51-2111G(2df) for C, 51-311G(2df) for N and O, and 2111(2pd) for H.
(27) PQS version 2.5, Parallel Quantum Solutions, 2013 Green Acres Rd., Suite A, Fayetteville, AR 72703. E-mail: sales@pqs-chem.com. URL: http://www.pqs-chem.com.
(28) White, C. A.; Johnson, B. G.; Gill, P. M. W.; Head-Gordon, M. *Chem. Phys. Lett.* **1994**, *230*, 8.
(29) QCHEM 2.0: Kong, J.; White, C. A.; Krylov, A. I.; Sherrill, D.; Adamson, R. D.; Furlani, T. R.; Lee, M. S.; Lee, A. M.; Gwaltney, S. R.; Adams, T. R.; Ochsenfeld, C.; Gilbert, A. T. B.; Kedziora, G. S.; Rassolov, V. A.; Maurice, D. R.; Nair, N.; Shao, Y.; Besley, N. A.; Maslen, P. E.; Dombroski, J. P.; Dasche, H.; Zhang, W.; Korambath, P. P.; Baker, J.; Byrd, E. F. C.; Voorhis, T. V.; Oumi, M.; Hirata, S.; Hsu, C.-P.; Ishikawa, N.; Florian, J.; Warshel, A.; Johnson, B. G.; Gill, P. M. W.; Head-Gordon, M.; Pople, J. A. *J. Comput. Chem.* **2000**, *21*, 1532.
(30) Strain, M. C.; Scuseria, G. E.; Frisch, M. J. *Science* **1996**, *271*, 51.
(31) Termath, V.; Handy, N. C. *Chem. Phys. Lett.* **1994**, *230*, 17.
(32) Schwegler, E.; Challacombe, M.; Head-Gordon, M. *J. Chem. Phys.* **1997**, *106*, 9708.