# Scalable Parallel 3d FFTs for Electronic Structure Codes

Andrew Canning

CRD, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA
Department of Applied Science, University of California, Davis, CA 95616, USA

**Abstract.** First-principles methods based on Density Functional Theory (DFT) where the wavefunctions are expanded in plane waves (Fourier components) are the most widely used approach for electronic structure calculations in materials science. The scaling of this method depends critically on having an efficient parallel 3d FFT that minimizes communications and calculations. We present an implementation and performance data of a parallel 3d FFT specifically designed for electronic structure calculations that scales to thousands of processors on leading parallel and vector computer platforms (IBM SP, Cray XT, NEC SX).

## 1 Parallel Implementation of 3d FFT

First-principles methods based on Density Functional Theory (DFT) in the Kohn-Sham (KS) [1] formalism are the most widely used approach for electronic structure calculations in materials science. The most common implementation of this approach involves the expansion of the wave functions in plane waves (Fourier components) and the use of pseudopotentials to replace the nucleus and core electrons. In this implementation we require parallel 3d FFTs to transform the electronic wavefunctions from Fourier space to real space to construct the charge density. Parallel 3d FFTs are also required in other parts of the code e.g. to transform potential terms from real space to Fourier space. This gives a computationally very efficient approach with a full quantum mechanical treatment for the valence electrons, allowing the study of systems containing hundreds of atoms on modest-sized parallel computers. Taken as a method DFT-based codes are one of the largest consumers of scientific computer cycles around the world with theoretical chemists, biologists, experimentalists etc. now becoming users of this approach. Parallel 3d FFTs are very demanding on the communication network of parallel computers as they require global transpositions of the FFT grid across the machine. The ratio of calculations to communications for 3d FFTs is of order $\log N$ where $N$ is the grid dimension (compared to a ration of $N$ for a distributed matrix multiply of matrix size $N$) which makes it one of the most demanding algorithms to scale on a parallel machine. A scalable parallel 3d FFT is critical to the overall scaling of plane wave DFT codes.

The Kohn-Sham formalism of DFT within the Local Density Approximation (LDA) requires that the wavefunctions of the electrons $\{\psi_i\}$ satisfy

$$[-\frac{1}{2}\nabla^2 + \sum_R v_{ion}(r-R) + \int \frac{\rho(r')}{|r-r'|} d^3r' + \mu_{xc}(\rho(r))]\psi_i = \varepsilon_i \psi_i \tag{1}$$

where $v_{ion}(r)$ is the ionic pseudopotential, $\rho(r)$ is the charge density and $\mu_{xc}(\rho(r))$ is the LDA exchange-correlation potential. We use periodic boundary conditions, expanding the wavefunctions in plane waves (Fourier components),

$$\psi_{j,\mathbf{k}}(\mathbf{r}) = \sum_{\mathbf{g}} a_{j,\mathbf{k}}(\mathbf{g}) e^{i(\mathbf{g}+\mathbf{k}).\mathbf{r}} \; . \tag{2}$$

The selection of the number of plane waves is determined by a cutoff $E_{cut}$ in the plane-wave kinetic energy $\frac{1}{2}|\mathbf{g}+\mathbf{k}|^2$ where $\{\mathbf{g}\}$ are reciprocal lattice vectors. This means that the representation of the wavefunctions in Fourier space is a sphere or ellipsoid with each $\mathbf{g}$ vector corresponding to a Fourier component (see figure 1(a)). The $\mathbf{k}$'s are vectors sampling the first Brillouin Zone (BZ) of the chosen unit cell (or supercell). The Kohn-Sham equations are usually solved by minimizing the total energy with an iterative scheme, such as conjugate gradient (CG), for a fixed charge density and then updating the charge density until self-consistency is achieved (for a review of this approach see reference [2]). Some parts of the calculation are done in Fourier space and some in real space transforming between the two using 3d FFTs. Our particular implementation in PARATEC (PARAllel Total Energy Code) [3] is based on a Grassmann conjugate gradient minimization [4] where all bands are minimized simultaneously. This allows us to use efficient BLAS3 routines for many parts of the calculation and also to block the communications to ensure MPI messaging is not latency dominated in the 3d FFTs.

The two most important criteria driving the choice of any parallelization strategy are equal division of the computational workload among the processors (load balancing) and minimization of the communications. We distribute the $\mathbf{g}$ vectors for each band among the processors by giving out columns of $\mathbf{g}$ vectors to each processor (see figure 1(a)). These columns are of different length depending on where they are in the sphere with the longest columns cutting the center of the sphere. The computations in PARATEC that are performed in Fourier space (e.g. non-local pseudopotential and orthogonalization) are load balanced by assigning each processor approximately the same number of $\mathbf{g}$ vectors. The load-balancing algorithm first orders the columns in descending order, and then distributes them among the processors such that the next-available column is assigned to the processor containing the fewest $\mathbf{g}$ vectors. The number of $\mathbf{g}$ vectors a processor has corresponds to the total length of columns it holds. It is necessary to distribute complete columns of $\mathbf{g}$ vectors to each processor as the first step in the 3d FFT performs 1d FFTs on columns of $\mathbf{g}$ vectors. The real-space data layout of the wavefunctions is on a standard Cartesian grid, where each processor holds a contiguous part of the space arranged in columns, as shown in figure 1(f). Each processor holds the same number of columns (to within one column) which load balances the real space part of the calculation. The charge density is constructed by performing 3d FFTs on each wavefunction to obtain the wavefunction on the larger real space grid. The wavefunctions are then squared and summed on this grid to produced the charge density that is then used in the calculation of the potential for the next self-consistent step in the solution of the Kohn-Sham equations.

A 3d FFT consists of three sets of 1d FFTs in the x,y and z directions with transpositions of the data between each set of 1d FFTs. Only two transposes are needed if the final data layout is not required to have the same x,y,z order in both spaces. Since the **g** vectors are distributed across the processors these two transposes can require global communications across the parallel computer and are the most communication intensive part of the whole calculation. We have therefore written a specialized 3d FFT to minimize the amount of communications. This 3d FFT is different from a standard 3d FFT as we have a sphere of points in Fourier space rather than a standard grid. This 3d FFT takes advantage of the fact that the real space grid is usually about twice the diameter of the sphere and at each of the three sets of 1d FFTs this sphere is in a sense expanding into the larger grid. We therefore only perform 1d FFTs and communications on the non-zero data elements which greatly reduces the amount of communications compared to using a standard library routine for the 3d FFT. Also when performing the second transpose to the final real space data layout (see figure 1) we choose the data layout to have as closely as possible complete planes of data on each processor so that the transpose is local and there is little data communication. In this way it is only the first transpose on the smaller data set where there is significant communication. Our 3d FFT can run on any number of processors for any grid and sphere size. If we used vendor supplied 3d FFTs we would have restrictions on grid sizes as well as performing more calculations and communications than our specialized 3d FFT since the grid size in Fourier and real space would have to be the same. The details of each step in our Fourier space to real space 3d FFT are (with z,y,x ordering in Fourier space and x,y,z in real space):

1. Each processor pads out the ends of each of the z-columns of $g$ vector coefficients that it holds with zeros to form full length z-columns on each processor. The complete data set is now a cylinder of length $2d$ and diameter $d$ where $d$ is the diameter of the original $g$ vector sphere and $2d$ is the cube size (see figure 1(b)).
2. Each processor performs one-dimensional FFTs on its set of z-columns.
3. The cylinder of data is now reorganized from z-columns to y-columns (ordered by their x,z indices) with each processor now holding a contiguous set of y-columns. Global data redistribution is required at this step (ie. going from figure 1(b) to figure 1(c)),as can be seen by the changes in color of the data elements. Each processor is given as closely as possible the same number of y-columns.
4. The y-columns (which are sections through the cylinder) are now padded with zeros at the ends to form full length columns. The complete data set is

now a slab of dimension $d$ in the x direction and $2d$ in the other directions (see figure 1(d)).

5. Each processor performs one-dimensional FFTs on its set of y-columns.
6. The slab of data is now transformed from y-columns (x,z ordered) to x-columns (y,z ordered) with each processor now having a set of contiguous x-columns (ie. going from figure 1(d) to figure 1(e)). Each processor is given as closely as possible the same number of x-columns. Communications are minimized at this step since most of the transformations are local to the processor with only data at the interfaces of the colored blocks being communicated. In the ideal case where there are complete $(y, x)$ planes on each processor the transpose can be done locally on each processor and there are no communications. Due to our choice of data layouts in the FFT the main communications are in step 3 where the data set (the cylinder) is much smaller than the slab.
7. The x-columns are now padded at the ends with zeros so the global data set is now the complete cube of side $2d$ (see figure 1(f)).
8. Each processor performs one-dimensional FFTs on its set of x-columns producing the final distributed real space representation of the wavefunction in x,y,z order.

An inverse 3d FFT is the reverse of these steps. While it is important to minimize the amount of data transfer in 3d FFTs, communication latency can also become a major issue. In the first transpose in the 3d FFT all the processors are sending data to all the other processors so the data packet size (for a fixed size physical system ) scales as the inverse of the number of processors squared. Therefore as we scale up to thousands of processors the data packets can become very small and communication latencies can dominate the code. To avoid this problem in our code we use an all-band method that allows us to perform many 3d FFTs at the same time and block the communications. There is an input parameter in our code which chooses the number of 3d FFTs to be performed at the same time. In this way, at the cost of using more memory, we can increase the packet size of the communications in the 3d FFTs to avoid the latency problem. For machines with higher latency like the IBM SP we have found that this can increase the speed of the code by 50-100% on runs in the hundreds of processor regime. For large processor counts we typically do up to fifty 3d FFTs at the same time which greatly reduces the latency problem.

## 2 Code Details and Performance

PARATEC is written in F90 and MPI and is designed primarily for massively parallel computing platforms, but can also run on serial machines. The code has run on many computer architectures and uses preprocessing to include machine specific routines such as the one dimensional FFT calls which are used in our specialized 3d FFTs. For the parallel vector platforms (Cray X1 and NEC SX) an efficient vector implementation of the one dimensional FFT libraries was required. The standard vendor supplied 1D FFT routines (on which our own

| $P$ | Bassi<br>IBM SP P5<br>Gflop/P | <br><br>%Pk | Thunder<br>Itanium2<br>Gflop/P | <br><br>%Pk | Phoenix<br>Cray X1E<br>Gflop/P | <br><br>%Pk | ES<br>NEC SX6<br>Gflop/P | <br><br>%Pk | Franklin<br>Cray XT4<br>Gflop/P | <br><br>%Pk |
|---|---|---|---|---|---|---|---|---|---|---|
| 64 | — | — | — | — | 4.88 | 27 | — | — | — | — |
| 128 | 5.49 | 72 | 2.84 | 51 | 3.80 | 21 | 5.12 | 64 | — | — |
| 256 | 5.52 | 73 | 2.63 | 47 | 3.24 | 18 | 4.97 | 62 | 3.36 | 65 |
| 512 | 5.13 | 67 | 2.44 | 44 | 2.22 | 12 | 4.36 | 55 | 3.15 | 61 |
| 1024 | — | — | 1.77 | 32 | — | — | 3.64 | 46 | 2.93 | 56 |
| 2048 | — | — | — | — | — | — | 2.67 | 33 | 2.65 | 46 |

**Table 1.** PARATEC results for a 488 atom CdSe quantum dot on the different platforms. The real space grid size for the 3d FFTs is 252*3. Bassi is an IBM SP with eight Power 5 processors per node, located at the NERSC computer center, Lawrence Berkeley National Laboratory. Th under is an Intel Itanium2 cluster with four processors per node and a Quadrics interconnect, located at Lawrence Livermore National Laboratory. Phoenix is a Cray X1E vector architecture located at Oak Ridge National Laboratory. The ES is the Earth Simulator which is a custom designed NEC SX6 located at the Earth Simulator Center, Yokohama. Franklin is a Cray XT4 with dual core Opteron processors and a 3d Torus interconnect, located at the NERSC computer center, Lawrence Berkeley National Laboratory.

specialized 3D FFTs are written) run at a relatively low percentage of peak. Code transformation was therefore required to rewrite our 3D FFT routines to use simultaneous (often called multiple) 1D FFT calls, which allow effective vectorization across many 1D FFTs. Additionally, compiler directives were inserted to force the vectorization and multistreaming (on the X1) for loops that contained indirect addressing. The main communications in the code are performed in the parallel 3d FFTs with most of the other parts of the code performing dense linear algebra on their local data. For the data presented in this paper the 3d FFTs typically take about 30% of the total runtime. Table 1 presents performance data for 3 CG steps of a 488 atom CdSe (Cadmium Selenide) quantum dot and a standard Local Density Approximation (LDA) run of PARATEC with a 35 Ry cut-off using norm-conserving pseudopotentials. The real space grid size for the 3d FFTs is 252 cubed and the calculation is for 709 bands. A typical calculation would require at least 60 CG iterations to converge the charge density for a CdSe dot. CdSe quantum dots are luminescent in the optical range at different frequencies depending on their size and can be used as electronic dye tags by attaching them to organic molecules. They represent a nanosystem with important technological applications.

As can be seen from Table 1 PARATEC obtains a high percentage of peak on both superscalar and vector based architectures. The machines with the best communication networks and lowest latency, such as the Cray XT4, have the best scaling to large processor counts for the 3d FFT and hence the whole code.

The Power 5 chip has the highest per processor performance for this code. This code makes heavy use of Cache in the FFTs as well as the other dense linear algebra operations so that RISC type architectures obtain a percentage of peak that is similar to vector machines. The ES achieved the highest peak performance of 5.5 Tflops on 2048 processors with the Cray XT4 being only a few percent slower. The Cray X1E obtained a lower percentage of peak due in part to some non-vectorizable sections of the code that run on the slow scalar processor. The NEC ES has a relatively faster scalar processor.

## 3 Discussion and Conclusions

In this paper we have present an efficient implementation of a parallel 3d FFT specifically designed for plane wave electronic structure codes. We have shown that with this 3d FFT our electronic structure code can scale well to thousands of processors on a variety of different computer architectures ranging from vector to superscalar. The limiting factor to scaling to larger processor counts is the communications in the 3d FFTs and we are investigating different communication schemes, such as using more collective operations, to allow us to scale to larger processor counts.

## 4 Acknowledgments

The author would like to thank L. Oliker J. Shalf and J. Carter for useful discussions and help in gaining access to different computer platforms. The author would also like to thank the staff of the Earth Simulator Center, especially Dr. T. Sato. This research used resources of the National Energy Research Scientific Computing Center, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098. This research used resources of the Center for Computational Sciences at Oak Ridge National Laboratory, which is supported by the Office of Science of the Department of Energy under Contract DE-AC05-00OR22725.

## References

1. W. Kohn and L.J. Sham, Phys. Rev. **140**, A1133 (1965).
2. M. Payne, M.P. Teter, D.C. Allan, T.A. Arias and J.D. Joannopoulos, Rev. Mod. Phys. **64**, 1045 (1992).
3. PARATEC (PARAllel Total Energy Code) www.nersc.gov/projects/paratec/ by B. Pfrommer, D. Raczkowski, A. Canning, S.G. Louie, Lawrence Berkeley National Laboratory (with contributions from F. Mauri, M. Côté, Y. Yoon, C. Pickard and P. Haynes).
4. D. Raczkowski, C.Y. Fong, P.A. Schultz, R.A. Lippert and E.B. STechel, Phys. Rev. B, **64**. 155203 (2001).

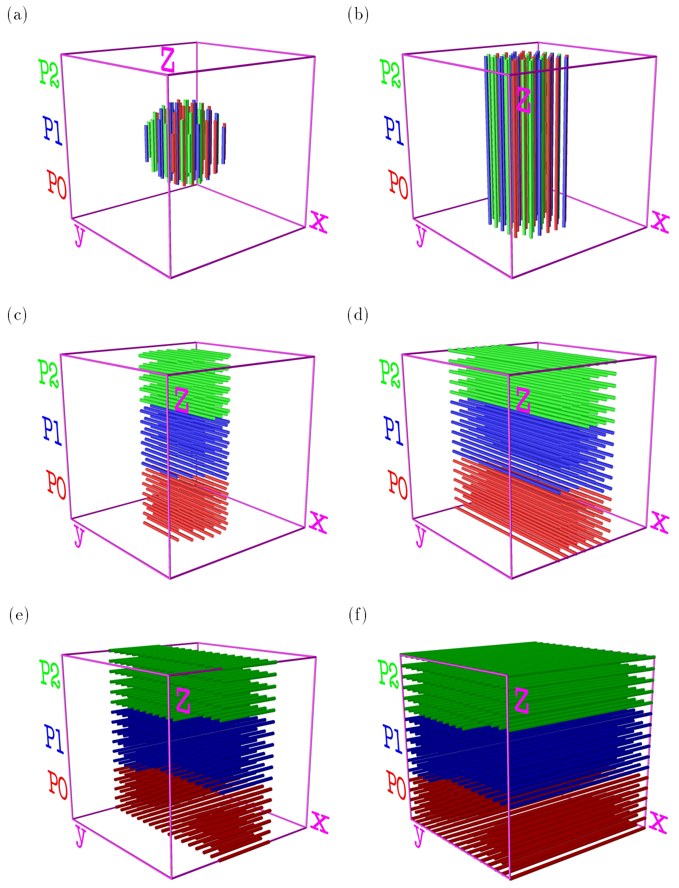

**Fig. 1.** Parallel three dimensional FFT. This figure shows which processors deal with which part of the grid during the three dimensional FFT. The colors red, blue and green correspond to the part of the grid that resides on processors zero to two