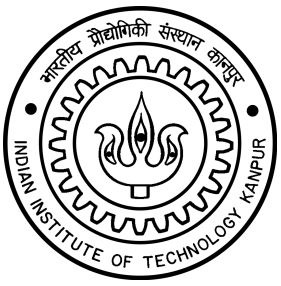
SURGE 2017 END-TERM REPORT

10 July, 2017



Indian Institute of Technology, Kanpur
Department of Chemistry

ON

Development of Density Functional Theory Methods for Molecular Computations

Project undertaken by

Soumyadeep Datta
Roll No: S170143
Undergraduate, Department of Electrical Engineering
Indian Institute of Technology, Kanpur

Project supervised by

Dr. Nisanth N. Nair
Associate Professor
Department of Chemistry

ABSTRACT

The Fourier transform is an important mathematical tool with widespread applications in science and engineering. In this project, we made an attempt to implement core subroutines for computing Fourier transforms of wavefunctions and potentials in three dimensions using Fast Fourier Transform (FFT) algorithms.

We modelled lattices in the form of real-space grids, optimized for a given energy cut-off. Then we fitted real-valued input functions on the real-space grids to generate input data. The G-space grids are generated to model the reciprocal lattice corresponding to the input lattice.

Using the popular and efficient FFTW library, we then computed the Fourier transforms of the input grids, which are essentially fitted over the G-space grids. We also recovered the input from the Fourier transform and found it to be identical to the original input.

Finally, we shifted the entire program from serial to parallel architecture to improve performance in terms of time and memory.

Contents

Chapter 1

INTRODUCTION

Density functional theory (DFT) is a computational quantum mechanical modelling method to investigate electronic structure of many-body systems, particularly atoms, molecules and condensed phases.[4] It derives its name from its usage of functionals of spatially dependent electron density to predict properties of many-electron systems. DFT is among the most popular and versatile methods used in physics, chemistry and material science. [5] DFT has been popularly used in solid-state physics calculations since 1970. The calculated results were quite satisfactorily close to experimental data and computational costs were significantly less compared to traditional approaches based on multi-electron wavefunctions, such as the Hartree-Fock theory and its descendants. However, it was not regarded to be accurate enough for quantum chemistry calculations until around 1990. Around this time, significant improvements were made in approximations to incorporate exchange and correlation corrections. Despite these improvements, difficulties arise when DFT is used to precisely describe intermolecular interactions, including van der Waals forces (dispersion), charge transfer excitations, transition states, global potential energy surfaces and in calculations of the band gap in semiconductors. DFT does not model the phenomenon of dispersion precisely and hence is not suitable, at least in isolation, for the treatment of dispersion dominated systems like interacting noble gas atoms. It is also not suitable for systems where dispersion competes significantly with other effects, e.g., in biomolecules. Contemporary research in this area focusses on developing new DFT methods to address these shortcomings by altering the functional or including additional terms to account for dispersion effects.

Chapter 2

THEORY

2.1 Supercells and the plane-wave basis sets [1]

2.1.1 Introduction

We can define unit cells of a periodically repeating system by Bravais lattice vectors $\mathbf{a}_1$, $\mathbf{a}_2$, $\mathbf{a}_3$ which can be combined into a 3x3 matrix $\mathbf{h} = [\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3]$. The volume of the cell can be calculated as

$$\Omega = det\mathbf{h}$$

The position vectors $\mathbf{r}$ are related to $\mathbf{h}$ via the scaled coordinates $\mathbf{s}$ as

$$\mathbf{r} = \mathbf{h}.\mathbf{s}$$

Reciprocal lattice vectors b_i are defined as

$$\mathbf{b}_i.\mathbf{a}_j = 2\pi\delta_{ij}$$

and can also be written as a 3x3 matrix

$$[\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3] = 2\pi(\mathbf{h}^T)^{-1}$$

Plane waves form a complete orthonormal basis

$$f_{\mathbf{G}}^{PW}(\mathbf{r}) = \frac{1}{\sqrt{\Omega}} \exp[i\mathbf{G}.\mathbf{r}]$$

where the reciprocal space vectors are given by

$$\mathbf{G} = 2\pi(\mathbf{h}^T)^{-1}\mathbf{g}$$

, where $\mathbf{g} = [i, j, k]$ is an integer triplet. In this basis, any periodic function can be expanded as

$$\varphi(\mathbf{r}) = \varphi(\mathbf{r} + \mathbf{L}) = \frac{1}{\sqrt{\Omega}} \sum_{\mathbf{G}} \varphi(\mathbf{G}) \exp[i\mathbf{G}.\mathbf{r}]$$

, $\varphi(\mathbf{r})$ and $\varphi(\mathbf{G})$ being related by a three-dimensional Fourier transform.
$\mathbf{L}$ is the periodicity of the direct lattice vector.

2.1.2 Plane wave expansions

For a periodic system, the Kohn-Sham potential exhibits the same periodicity as the direct lattice

$$V^{KS}(\mathbf{r}) = V^{KS}(\mathbf{r} + \mathbf{L})$$

In general Bloch form, the Kohn-Sham orbitals can be written as

$$\phi_j(\mathbf{r}, \mathbf{k}) = \exp[i\mathbf{k}.\mathbf{r}] u_j(\mathbf{r}, \mathbf{k})$$

where $\mathbf{k}$ is a vector in the first Brillouin zone. The functions $u_j(\mathbf{r}, \mathbf{k})$ have the same periodicity as the direct lattice

$$u_j(\mathbf{r}, \mathbf{k}) = u_j(\mathbf{r} + \mathbf{L}, \mathbf{k})$$

The index j runs over all states, each having a relative weight $f_j(\mathbf{k})$ associated with them. The periodic functions $u_j(\mathbf{r}, \mathbf{k})$ can now be extended in plane wave basis

$$u_j(\mathbf{r}, \mathbf{k}) = \frac{1}{\sqrt{\Omega}} \sum_{\mathbf{G}} c_j(\mathbf{G},\ \mathbf{k}) \exp[i\mathbf{G}.\mathbf{r}]$$

the Kohn-Sham orbitals being

$$\phi_j(\mathbf{r}, \mathbf{k}) = \frac{1}{\sqrt{\Omega}} \sum_{\mathbf{G}} c_j(\mathbf{G},\ \mathbf{k}) \exp[i(\mathbf{G} + \mathbf{k}).\mathbf{r}]$$

where $c_j(\mathbf{G},\mathbf{k})$ are complex numbers. Using this expansion, we can expand the charge density as

$$n(\mathbf{r}) = \frac{1}{\Omega} \sum_j \int f_j(\mathbf{k}) \sum_{\mathbf{G},\mathbf{G'}} c_j^*(\mathbf{G'}.\mathbf{k}) c_j(\mathbf{G},\mathbf{k}) \exp[i(\mathbf{G} + \mathbf{k}).\mathbf{r}] d\mathbf{k}$$

where the sum over $\mathbf{G}$ vectors extends over twice the range given by the wave function expansion.

2.1.3 Cutoffs and k-points

In reality the infinite sums over **G** vectors and cells have to be truncated. Moreover, the integral has to be approximated over the Brillouin zone by a finite sum over k-points.

$$\int_{BZ} d\mathbf{k} \Rightarrow \sum_{\mathbf{k}} w_k,$$

where w_k are weights of the integration points.

The Kohn-Sham potential $V^{KS}(\mathbf{G})$ converges rapidly with increasing modulus of **G**. Hence, at each **k**-point, only those **G** vectors within a certain kinetic energy cutoff are permitted.

$$\frac{1}{2}|\mathbf{k}+\mathbf{G}|^2 \leq E_{cut}$$

The number of plane waves for a given cutoff depends on the k-points and the unit cell. A rough estimate at the centre of the Brillouin zone is

$$N_{PW} = \frac{1}{2\pi^2}\Omega E_{cut}^{3/2}$$

, where E_{cut} and Ω are in atomic units (Hartree and $Bohr^3$, respectively).

2.1.4 Real space grid and fast Fourier transforms

We can define a function given as a finite linear transformation of plane waves as a set of functional values on an equally spaced grid in real space, the maximal grid spacing given by the sampling theorem to be, say, $N_s, s = x, y, z$. The real space sampling points are

$$\mathbf{R} = \mathbf{hNq},$$

where **N** is a diagonal matrix with the entries $1/N_s$ and **q** is a vector of integers from 0 to $N_s - 1$. For the sampling theorem, N_s must be bigger than $2max\mathbf{g}_s + 1$. A periodic function is calculated at the real space grid points **R**

$$f(\mathbf{R}) = \sum_{\mathbf{G}} f(\mathbf{G}) \exp[i\mathbf{G}.\mathbf{R}]$$

The function $f(\mathbf{G})$ is zero outside the cutoff region. The functions $f(\mathbf{R})$ and $f\mathbf{G}$ are related by three dimensional Fourier transforms as

$$f(\mathbf{R}) = inv\_FT[f(\mathbf{G})]$$

$$f(\mathbf{G}) = fw\_FT[f(\mathbf{R})]$$

2.2 The Fast Fourier Transform(FFT)[2]

2.2.1 Introduction

A *fast Fourier transform (FFT)* computes the discrete Fourier transform(DFT) of a sequence, or its inverse. Widely used in various applications in engineering, science and mathematics, it is regarded as one of the most important algorithms of the 20th century. Typically, the FFT computes the discrete Fourier transform with a time complexity of $\mathbf{O}(n * logn)$ as opposed to $\mathbf{O}(n^2)$ obtained using the definition for a DFT.

2.2.2 The Danielson-Lenczos Lemma

Danielson and Lanczos "rediscovered" the FFT in 1942 when they showed that a discrete Fourier transform of length N can be written as the sum of two independent discrete Fourier transforms of length *N/2*. One of them is formed by the even-numbered terms, the other by the odd-numbered terms. The proof is as follows:

$$F_k = \sum_{j=0}^{N-1} e^{2\pi ijk/N} f_j$$

$$= \sum_{j=0}^{N/2-1} e^{2\pi i(2j)k/N} f_{2j} + \sum_{j=0}^{N/2-1} e^{2\pi i(2j+1)k/N} f_{2j+1}$$

$$= \sum_{j=0}^{N/2-1} e^{2\pi ijk/(N/2)} f_{2j} + W^k \sum_{j=0}^{N/2-1} e^{2\pi ijk/(N/2)} f_{2j+1}$$

$$= F_k^e + W^k F_k^o$$

where $W \equiv e^{2\pi i/N}$, F_k^e is the kth component of the Fourier transform of length *N/2* formed by the even numbered components of the $f_j s$ and F_k^o is that formed by the odd numbered components, also of length *N/2*.

This algorithm can be used recursively, to reduce the problem size further to *N/4* and so on. The F_k^e and F_k^o can be further decomposed into F_k^{ee}, F_k^{eo}, F_k^{oe} and F_k^{oo} respectively. The easiest case to deal with is when the original N is a power of 2. Hence, it is highly recommended to round the input to the *nearest power of 2* before beginning computation. Other algorithms, using relations analogous to the Danielson-Lenczos Lemma, enable us to deal with inputs which can be factorized completely into small primes (2,3,5,7,11,13...).

Chapter 3

IMPLEMENTATION

The entire implementation has been done for a *simple cubic(SCC)* lattice.

3.1 Determining appropriate number of grid points

Given an energy cutoff (E_{cut}), we proceeded to calculate the appropriate cell dimensions using the following algorithm.

Step 1 : Start with $N_x = N_y = N_z = 0$

Step 2 : Compute $G_{max} = \sqrt{(2.0 * E_{cut})}$

Step 3 : Increment N_x, N_y, N_z all by 1

Step 4 : Compute the current dimensions of the reciprocal lattice $b_1 = 2\pi(N_x/L_x), b_2 = 2\pi(N_y/L_y), b_3 = 2\pi(N_z/L_z)$

Step 5 : IF $(b_1 * b_2 * b_3 < (4/3)\pi G^3_{max}$[6]) THEN GOTO Step 3
ELSE GOTO Step 6

Step 6 : Calculate NR_x, NR_y, NR_z, the closest estimates to N_x, N_y, N_z which can be factorized completely into small primes (2,3,5,7,11...)[7]

L_x, L_y, L_z = Dimensions of supercell in x, y and z dimensions
N_x, N_y, N_z = Number of grid points in x, y and z directions

3.2 Creating real space grids[3]

The following expressions were used to generate grid points for the direct lattice:

$$x = \{0, 1, 2, ..., N_x - 1\}(L_x/N_x)$$

$$y = \{0, 1, 2, ..., N_y - 1\}(L_y/N_y)$$

$$z = \{0, 1, 2, ..., N_z - 1\}(L_z/N_z)$$

3.3 Fitting functions over real space grids

Now we define a function $f(x, y, z)$ over the real space grids, essentially sampling a continuous function over the grid points.

3.4 Creating reciprocal lattices[3]

The following expressions were used to generate grid points for the reciprocal lattice:

$$x = \{0, 1, 2, ..., N_x - 1\}(L_x/N_x)$$

$$y = \{0, 1, 2, ..., N_y - 1\}(L_y/N_y)$$

$$z = \{0, 1, 2, ..., N_z - 1\}(L_z/N_z)$$

3.5 Computing the Fourier transform

We used highly optimized subroutines of the "Fastest Fourier Transform in the West" (FFTW) library[8] to compute the three-dimensional Fourier transforms of the real valued function defined over the direct lattice (real space grid). The three-dimensional Fourier transforms are naturally defined on the reciprocal lattice (G-space grid).
We also recovered the input from the Fourier transforms. The recovered input was found to be nearly identical to the original input.

3.6 Shifting to parallel architecture

We are currently shifting the entire program from a serial architecture to a parallel architecture using the *Message Passing Interface(MPI)[9]* standards.

Chapter 4

RESULTS

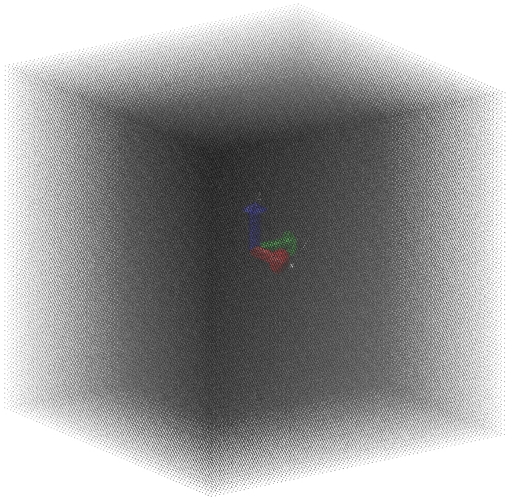


Figure 4.1: Real space grids

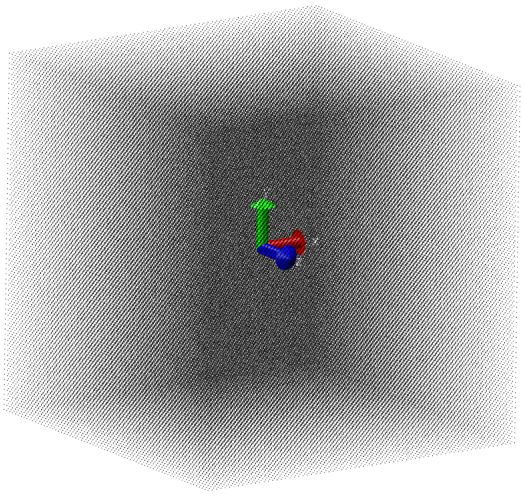


Figure 4.2: G space grids

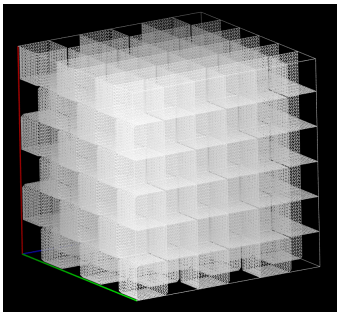


Figure 4.3: $f(x, y, z) = sinx * siny * sinz$ fitted over the direct lattice.

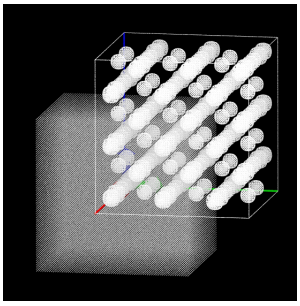


Figure 4.4: The three dimensional Fourier transform

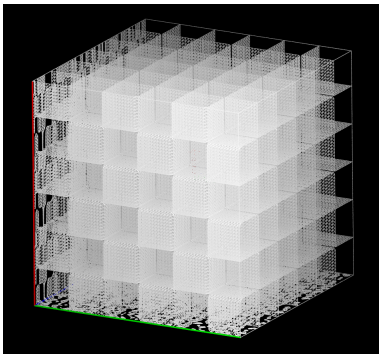


Figure 4.5: The recovered input

The results were obtained for an energy cutoff $E_{cut} = 25.0$ Hartree. The input function chosen was $f(x, y, z) = sinx * siny * sinz$, which was sampled over the direct lattice. The cube file generated shows the isosurfaces at an isovalue of 0, i.e., the nodal planes. The output is the three dimensional Fourier transform. The cube file shows the grid points together with the output, represented as an isosurface at isovalue 1519.27. As can be easily observed, the recovered input, represented again as an isosurface with isovalue 0, is nearly identical to the original input, in accordance with expectations.

Chapter 5

FUTURE PROSPECTS

We have already done one form of parallelization of the entire process by parallelizing the formation of the grids, the generation of the input data(samples of a real valued function over the direct lattice) and the three dimensional Fourier transform. However, the technique we used is, though easy to implement, inefficient in terms of memory usage. The following image shows an efficient way of parallelization: [10]

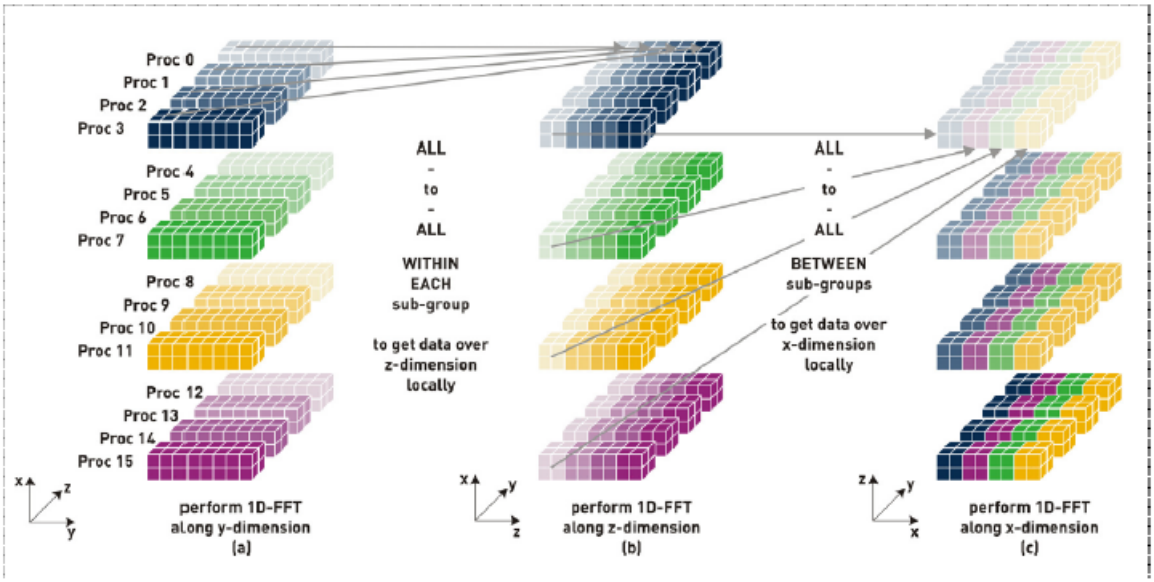


Figure 5.1: 2D decomposition of 3D FFT. Axes rotated for clarity

As the figure suggests, the 3D FFT can be computed by the following steps:

1. 1D FFT along a local dimension

2. Global transpose within the same subgroup

3. 1D FFT along second dimension

4. Global transpose

5. 1D FFT along third dimension

Existing packages using this method of decomposition of a 3D FFT into 2D and 1D FFTs include P3DFFT by Dmitry Pekurovsky[11], which uses the MPI_Alltoallv subroutine.

Any further improvement made in this crucial method of computing FFTs shall be of immense significance to the entire scientific and engineering community. A reduction in time complexity of calculation of Fourier transforms shall immensely speed up several crucial calculations in various spheres of sciences, engineering and mathematics.

The final goal of this project is to invent a linear scaling Density Functional Theory method. This, once finally implemented, shall bring about sweeping increase in efficiency of computational chemistry and physics codes.

Chapter 6

ACKNOWLEDGEMENT

I would like to take this opportunity to express my heartiest gratitude to the authorities of the Summer Undergraduate Research and Graduate Excellence (SURGE) 2017 Programme for providing us such a wonderful platform to explore our interests and interact with some of the brightest brains in the nation.

I sincerely thank Dr. Nisanth N. Nair, Associate Professor, Department of Chemistry, for guiding me constantly at every step and providing me with all the resources I would need from time to time. Working in his lab has been a memorable experience.

I would also like to thank Mr. Satyendra Soni, Project Associate under Dr. Nair, for collaborating with me in spite of his own ongoing projects. I could have done nothing without his help.

I would also like to thank all the other members of Dr. Nair's research group. It has been a nice experience working in the same lab and I will cherish the memories.

Bibliography

[1] Dominik Marx and Jurg Hütter. *Ab Initio Molecular Dynamics: Basic Theory and Advanced Methods*. Cambridge University Press, Cambridge, UK, 2009.

[2] et al W.H. Press, Brian P. Flannery. *Numerical Recipes in Fortran 77*. Cambridge University Press, Cambridge, UK, 1992.

[3] Nisanth N. Nair. Lecture notes for r.a.r.e. 2014.

[4] Wikipedia.

[5] Computational chemistry wiki.

[6] Sverre Froyen. Real space grids and fast fourier transforms in pseudopotential solid state calculations. *Physical Review B*, 1991.

[7] Carl-parinello molecular dynamics. "http://www.cpmd.org/".

[8] The fftw library. "http://fftw.org/".

[9] Message passing interface tutorials. "https://computing.llnl.gov/tutorials/mpi/".

[10] Roland Schulz. 3d fft with 2d decomposition.

[11] Dmitry Pekurovsky. P3dfft: A framework for parallel computations of fourier transforms in three dimensions. *Society for Industrial and Applied Mathematics*, 2012.