DFT Implementation Using STO-3G Basis Sets: A Computational Approach to Electronic Structure Calculations

Magane Tamandja

March 25, 2025

1 abstract

This report presents an efficient implementation of Density Functional Theory (DFT) using the STO-3G basis set for electronic structure calculations. We develop analytical expressions for the Laplacian operator, electron repulsion terms, and exchange-correlation potentials within the Local Density Approximation (LDA) framework. Our implementation demonstrates [briefly mention a key result or advantage of your approach]. The mathematical framework established here provides a foundation for more advanced implementations that could significantly reduce computational costs while maintaining accuracy for molecular systems.

2 Introduction

1. Introduction

Density Functional Theory (DFT) has emerged as one of the most powerful and widely used methods for electronic structure calculations in computational chemistry and materials science. The popularity of DFT stems from its favorable balance between computational efficiency and accuracy, making it suitable for studying systems ranging from small molecules to extended materials.

Despite its widespread use, implementing DFT algorithms that are both accurate and computationally efficient remains challenging. The treatment of exchange-correlation functionals, basis set selection, and numerical integration schemes all significantly impact the performance and reliability of DFT calculations.

In this report, we present a detailed implementation of a DFT algorithm using the STO-3G basis set. We focus specifically on three critical components: 1) The mathematical treatment of the Laplacian operator 2) The computation of electron repulsion integrals 3) The implementation of exchange-correlation potentials within the Local Density Approximation (LDA)

Our goal is to develop a computational framework that balances accuracy with efficiency for electronic structure calculations, providing a foundation for future extensions to more complex molecular systems.

3 The Khon-Sham equation

In the following sections we will develop the algorithms given by the KS equation given by :

$$\left(-\frac{1}{2}\nabla^2 + \left[\sum_j^N \int \frac{|\varphi_j(\vec{r_2})|^2}{r_{12}} d\vec{r_2} + V_{XC}(\vec{r_1}) - \sum_A^M \frac{Z_A}{r_{1A}}\right]\right)\varphi_i = \epsilon_i \varphi_i$$

The strategy is to get to an eigen value problem. So we have to transform the expression $\left(-\frac{1}{2}\nabla^2 + \left[\sum_j^N \int \frac{|\varphi_j(\vec{r_2})|^2}{r_{12}} d\vec{r_2} + V_{XC}(\vec{r_1}) - \sum_A^M \frac{Z_A}{r_{1A}}\right]\right)$ into a matrix operator.

3.1 Treatment of the Laplacian

Given the Laplacian operator $-\frac{1}{2}\nabla^2$,we choose to use the STO-3G basis set, which is given by :

$$\psi_{\text{STO-3G}}(s) = c_1\phi_1 + c_2\phi_2 + c_3\phi_3$$

where

$$\phi_1 = \left(\frac{2\alpha_1}{\pi}\right)^{3/4} e^{-\alpha_1 r^2}$$
$$\phi_2 = \left(\frac{2\alpha_2}{\pi}\right)^{3/4} e^{-\alpha_2 r^2}$$
$$\phi_3 = \left(\frac{2\alpha_3}{\pi}\right)^{3/4} e^{-\alpha_3 r^2}$$

So we can rewrite the Laplacian as

$$\begin{aligned}\frac{d^2(c_1\phi_1 + c_2\phi_2 + c_3\phi_3)}{d^2r} &= \frac{d^2(c_1\left(\frac{2\alpha_1}{\pi}\right)^{3/4} e^{-\alpha_1 r^2} + c_2\left(\frac{2\alpha_2}{\pi}\right)^{3/4} e^{-\alpha_2 r^2} + c_3\left(\frac{2\alpha_3}{\pi}\right)^{3/4} e^{-\alpha_3 r^2})}{d^2r} \\ &= (4\alpha_1^2 r^2 - 2\alpha_1)c_1\left(\frac{2\alpha_1}{\pi}\right)^{3/4} e^{-\alpha_1 r^2} \\ &+ (4\alpha_2^2 r^2 - 2\alpha_2)c_2\left(\frac{2\alpha_2}{\pi}\right)^{3/4} e^{-\alpha_2 r^2} \\ &+ (4\alpha_3^2 r^2 - 2\alpha_3)c_3\left(\frac{2\alpha_3}{\pi}\right)^{3/4} e^{-\alpha_3 r^2}\end{aligned}$$

In the spirit of constructing a matrix hamiltonian, we diagonilize the expression to get the expression

$$\frac{d^2(c_1\phi_1+c_2\phi_2+c_3\phi_3)}{d^2r} = \begin{bmatrix} (4\alpha_1^2r^2-2\alpha_1) & 0 & 0 \\ 0 & (4\alpha_2^2r^2-2\alpha_2) & 0 \\ 0 & 0 & (4\alpha_3^2r^2-2\alpha_3) \end{bmatrix} (c_1\left(\tfrac{2\alpha_1}{\pi}\right)^{3/4}e^{-\alpha_1r^2}+c_2\left(\tfrac{2\alpha_2}{\pi}\right)^{3/4}e^{-\alpha_2r^2}+ c_3\left(\tfrac{2\alpha_3}{\pi}\right)^{3/4}e^{-\alpha_3r^2}).$$

This provides a computable expression for the Laplacian. We still run into the elements of the of the matrix having the variable r. We therefore apply the formula:

$$L_{ij} = \int \phi_i^*(r)(-\frac{1}{2}\nabla^2)\phi_j(r)dr.$$

In our case, we have :

$$\begin{aligned}
L_{ij} &= \int \phi_i^*(r)(-\frac{1}{2}\nabla^2)\phi_j(r)dr \\
&= \int \phi_i^*(r)(4\alpha_i^2r^2-2\alpha_i)\phi_j(r)dr, \text{since we are working with a diagonal matrix and STO -3G is real,} \\
&= \int \phi_i(r)(4\alpha_i^2r^2-2\alpha_i)\phi_i(r)dr \\
&= \int \phi_i(r)^2(4\alpha_i^2r^2-2\alpha_i) \\
&= \int \left(\left(\frac{2\alpha_i}{\pi}\right)^{3/4}e^{-\alpha_ir^2}\right)^2(4\alpha_i^2r^2-2\alpha_i) \\
&= \int \left(\left(\frac{2\alpha_i}{\pi}\right)^{3/2}e^{-2\alpha_ir^2}\right)(4\alpha_i^2r^2-2\alpha_i) \\
&= \int \left(4\alpha_i^2\left(\frac{2\alpha_i}{\pi}\right)^{3/2}r^2e^{-2\alpha_ir^2} - 2\alpha_i\left(\frac{2\alpha_i}{\pi}\right)^{3/2}e^{-2\alpha_ir^2}\right) \\
&= \int 4\alpha_i^2\left(\frac{2\alpha_i}{\pi}\right)^{3/2}r^2e^{-2\alpha_ir^2} - \int 2\alpha_i\left(\frac{2\alpha_i}{\pi}\right)^{3/2}e^{-2\alpha_ir^2} \\
&= 4\alpha_i^2\left(\frac{2\alpha_i}{\pi}\right)^{3/2}\frac{1}{4\alpha_i}\sqrt{\frac{\pi}{2\alpha_i}} - 2\alpha_i\left(\frac{2\alpha_i}{\pi}\right)^{3/2}\sqrt{\frac{\pi}{2\alpha_i}} \\
L_{ii} &= -\alpha_i\left(\frac{2\alpha_i}{\pi}\right)^{3/2}\sqrt{\frac{\pi}{2\alpha_i}}.
\end{aligned}$$

Finally, we get the Laplacian :

$$\begin{bmatrix} -\alpha_1\left(\frac{2\alpha_1}{\pi}\right)^{3/2}\sqrt{\frac{\pi}{2\alpha_1}} & 0 & 0 \\ 0 & -\alpha_2\left(\frac{2\alpha_2}{\pi}\right)^{3/2}\sqrt{\frac{\pi}{2\alpha_2}} & 0 \\ 0 & 0 & -\alpha_3\left(\frac{2\alpha_3}{\pi}\right)^{3/2}\sqrt{\frac{\pi}{2\alpha_3}} \end{bmatrix}$$

The following C++ code is a function implementation that is needed to generate each element:

```
double y_feta(double c , double a){
    return c*((2*a)/M_PI)^(3/4);
}

double y_gaussian_integral(double a, double b){
    return sqrt(M_PI/(a+b));
}
```

3.2 Treatment of the Electron Repulsion Term

We start by the following assumption, the integral is defined by infinity, this make calculations convenient.
Given the term $|\varphi_j(\vec{r_2})|^2$ can be written as $(\varphi_j(\vec{r_2}))^2$ for a real basis function, we write:

$$
\begin{aligned}
|\varphi_j(\vec{r_2})|^2 &= \left[c_1\left(\frac{2\alpha_1}{\pi}\right)^{3/4} e^{-\alpha_1 r^2} + c_2\left(\frac{2\alpha_2}{\pi}\right)^{3/4} e^{-\alpha_2 r^2} + c_3\left(\frac{2\alpha_3}{\pi}\right)^{3/4} e^{-\alpha_3 r^2}\right]^2 \\
&= \left[c_1\left(\frac{2\alpha_1}{\pi}\right)^{3/4}\right]^2 e^{-2\alpha_1 r^2} + 2\left[c_2\left(\frac{2\alpha_2}{\pi}\right)^{3/4}\right]\left[c_1\left(\frac{2\alpha_1}{\pi}\right)^{3/4}\right] e^{-\alpha_1 r^2-\alpha_2 r^2} \\
&+ 2\left[c_3\left(\frac{2\alpha_3}{\pi}\right)^{3/4}\right]\left[c_1\left(\frac{2\alpha_1}{\pi}\right)^{3/4}\right] e^{-\alpha_1 r^2-\alpha_3 r^2} + \left[c_2\left(\frac{2\alpha_2}{\pi}\right)^{3/4}\right]^2 e^{-2\alpha_2 r^2} \\
&+ \left[c_3\left(\frac{2\alpha_3}{\pi}\right)^{3/4}\right]^2 e^{-2\alpha_3 r^2} + 2\left[c_2\left(\frac{2\alpha_2}{\pi}\right)^{3/4}\right]\left[c_3\left(\frac{2\alpha_3}{\pi}\right)^{3/4}\right] e^{-\alpha_2 r^2-\alpha_3 r^2}
\end{aligned}
$$

$$
\begin{aligned}
|\varphi_j(\vec{r_2})|^2 &= \left[c_1\left(\frac{2\alpha_1}{\pi}\right)^{3/4}\right]^2 e^{-2\alpha_1 r^2} + 2\left[c_2\left(\frac{2\alpha_2}{\pi}\right)^{3/4}\right]\left[c_1\left(\frac{2\alpha_1}{\pi}\right)^{3/4}\right] e^{(-\alpha_1-\alpha_2) r^2} \\
&+ 2\left[c_3\left(\frac{2\alpha_3}{\pi}\right)^{3/4}\right]\left[c_1\left(\frac{2\alpha_1}{\pi}\right)^{3/4}\right] e^{(-\alpha_1-\alpha_3) r^2} + \left[c_2\left(\frac{2\alpha_2}{\pi}\right)^{3/4}\right]^2 e^{-2\alpha_2 r^2} \\
&+ \left[c_3\left(\frac{2\alpha_3}{\pi}\right)^{3/4}\right]^2 e^{-2\alpha_3 r^2} + 2\left[c_2\left(\frac{2\alpha_2}{\pi}\right)^{3/4}\right]\left[c_3\left(\frac{2\alpha_3}{\pi}\right)^{3/4}\right] e^{(-\alpha_2-\alpha_3) r^2}
\end{aligned}
$$

$$\int |\varphi_j(\vec{r_2})|^2 dr = \int \left[c_1 \left(\frac{2\alpha_1}{\pi}\right)^{3/4}\right]^2 e^{-2\alpha_1 r^2} dr + \int 2 \left[c_2 \left(\frac{2\alpha_2}{\pi}\right)^{3/4}\right] \left[c_1 \left(\frac{2\alpha_1}{\pi}\right)^{3/4}\right] e^{(-\alpha_1-\alpha_2)r^2} dr$$
$$+ \int 2 \left[c_3 \left(\frac{2\alpha_3}{\pi}\right)^{3/4}\right] \left[c_1 \left(\frac{2\alpha_1}{\pi}\right)^{3/4}\right] e^{(-\alpha_1-\alpha_3)r^2} dr + \int \left[c_2 \left(\frac{2\alpha_2}{\pi}\right)^{3/4}\right]^2 e^{-2\alpha_2 r^2} dr$$
$$+ \int \left[c_3 \left(\frac{2\alpha_3}{\pi}\right)^{3/4}\right]^2 e^{-2\alpha_3 r^2} dr + \int 2 \left[c_2 \left(\frac{2\alpha_2}{\pi}\right)^{3/4}\right] \left[c_3 \left(\frac{2\alpha_3}{\pi}\right)^{3/4}\right] e^{(-\alpha_2-\alpha_3)r^2} dr$$
$$\int |\varphi_j(\vec{r_2})|^2 dr = \left[c_1 \left(\frac{2\alpha_1}{\pi}\right)^{3/4}\right]^2 \int e^{-2\alpha_1 r^2} dr + 2 \left[c_2 \left(\frac{2\alpha_2}{\pi}\right)^{3/4}\right] \left[c_1 \left(\frac{2\alpha_1}{\pi}\right)^{3/4}\right] \int e^{(-\alpha_1-\alpha_2)r^2} dr$$
$$+ 2 \left[c_3 \left(\frac{2\alpha_3}{\pi}\right)^{3/4}\right] \left[c_1 \left(\frac{2\alpha_1}{\pi}\right)^{3/4}\right] \int e^{(-\alpha_1-\alpha_3)r^2} dr + \left[c_2 \left(\frac{2\alpha_2}{\pi}\right)^{3/4}\right]^2 \int e^{-2\alpha_2 r^2} dr$$
$$+ \left[c_3 \left(\frac{2\alpha_3}{\pi}\right)^{3/4}\right]^2 \int e^{-2\alpha_3 r^2} dr + 2 \left[c_2 \left(\frac{2\alpha_2}{\pi}\right)^{3/4}\right] \left[c_3 \left(\frac{2\alpha_3}{\pi}\right)^{3/4}\right] \int e^{(-\alpha_2-\alpha_3)r^2} dr$$

Using the Gaussian integral formula:

$$\int_{-\infty}^{\infty} e^{-\alpha r^2} dr = \sqrt{\frac{\pi}{\alpha}};$$

we can rewrite the integral terms as :

$$\int |\varphi_j(\vec{r_2})|^2 dr = \left[c_1 \left(\frac{2\alpha_1}{\pi}\right)^{3/4}\right]^2 \sqrt{\frac{\pi}{2\alpha_1}} + 2 \left[c_2 \left(\frac{2\alpha_2}{\pi}\right)^{3/4}\right] \left[c_1 \left(\frac{2\alpha_1}{\pi}\right)^{3/4}\right] \sqrt{\frac{\pi}{\alpha_1+\alpha_3}}$$
$$+ 2 \left[c_3 \left(\frac{2\alpha_3}{\pi}\right)^{3/4}\right] \left[c_1 \left(\frac{2\alpha_1}{\pi}\right)^{3/4}\right] \sqrt{\frac{\pi}{\alpha_1+\alpha_3}} + \left[c_2 \left(\frac{2\alpha_2}{\pi}\right)^{3/4}\right]^2 \sqrt{\frac{\pi}{2\alpha_2}}$$
$$+ \left[c_3 \left(\frac{2\alpha_3}{\pi}\right)^{3/4}\right]^2 \sqrt{\frac{\pi}{2\alpha_3}} + 2 \left[c_2 \left(\frac{2\alpha_2}{\pi}\right)^{3/4}\right] \left[c_3 \left(\frac{2\alpha_3}{\pi}\right)^{3/4}\right] \sqrt{\frac{\pi}{\alpha_2+\alpha_3}}$$

This provides us with a relatively easy to compute a coulomb contribution. Using the point charge approximation.

$$V_c \approx \frac{Q_A Q_B}{R_{AB}}.$$

With $Q_A \approx \int |\phi_A(r)|^2 dr$, where ϕ is φ for all $c_i = 1$. A and B are nucleus centers. We implement the following function to perform these calculations:

```
double y_coulomb(double feta_1, double feta_2, double feta_3,
double gi1, double gi2, double gi3, double gi4, double gi5){
 return (((feta_1^2)*gi1)+(2*feta_2*feta_1*gi2)+(2*feta_3*
 feta_1*gi2)+((feta_2^2)*gi3)+((feta_3^2)*gi4)+
 (2*feta_2*feta_3*gi5));
}

double y_coulomb_interaction(double QA, double QB, double RAB){
return (QA*QB)/RAB;

}
```

3.3 Treatment of the exchange potential : LDA

Using the LDA approximation of the exchange potential, we get:

$$V_X^{LDA} = -\frac{3}{4}(\frac{3}{\pi})^{\frac{1}{3}}\rho^{\frac{1}{3}}(r)$$

Here we choose to treat ρ as constant average, we use the formula $\varphi = N/V$. where N is the the number of electron and V is the approximated volume calculated using the formula $V = \sum_A V_{total} - V_{overlap}$. In turn, $V = \frac{4}{3}\pi R_{vdw}^3$ and for $V_{overlap}$, $R = d_{bond}/2$. The corresponding c++ code is straight forward

```
double y_volumes(double rs[]){
    double solution = 0;
    for (int i=0; i < sizeof(rs); i++){
        solution = solution + (4/3)*M_1_PI*(rs[i])^3
    }
    return solution;
     }

double y_exchange_potential(double av_density){
return -(3/4)*((3/M_1_PI)^(1/3))*(av_density^(1/3))
       }
```

3.4 Treatment of the nuclear attraction term

The nuclear attraction term is given by the formula:

$$V_{nuclear} = -\sum_A^M \frac{Z_A}{r_{1A}}.$$

Given that the density is constant, the average electron-nucleus distance is $(3/4)R$. R being the van der Waals radius.

4 KS equation of CO2

What make DFT related calculation complex is the complexity of implementation. This is why using an actual molecule like CO2 as a prototype is very usefull. Here, we use less sofisticated methods like molecular mechanics to get inter nucleus distances. We find C=O bonds to be 1.1970 Armstrong.

4.1 Treatment of α and c

Using reliable sources like https://www.basissetexchange.org, we get the α and c parameters for each atom.

```
!----------------------------------------------------------------------
! Basis Set Exchange
! Version 0.11
! https://www.basissetexchange.org
!----------------------------------------------------------------------
!   Basis set: STO-3G
! Description: STO-3G Minimal Basis (3 functions/AO)
!        Role: orbital
!     Version: 1  (Data from Gaussian09)
!----------------------------------------------------------------------

C     0
S    3   1.00
      0.7161683735D+02        0.1543289673D+00
      0.1304509632D+02        0.5353281423D+00
      0.3530512160D+01        0.4446345422D+00
SP   3   1.00
      0.2941249355D+01       -0.9996722919D-01        0.1559162750D+00
      0.6834830964D+00        0.3995128261D+00        0.6076837186D+00
      0.2222899159D+00        0.7001154689D+00        0.3919573931D+00
****
O     0
S    3   1.00
      0.1307093214D+03        0.1543289673D+00
      0.2380886605D+02        0.5353281423D+00
      0.6443608313D+01        0.4446345422D+00
SP   3   1.00
      0.5033151319D+01       -0.9996722919D-01        0.1559162750D+00
      0.1169596125D+01        0.3995128261D+00        0.6076837186D+00
      0.3803889600D+00        0.7001154689D+00        0.3919573931D+00
****
```

We can now start computing the Laplacian matrix. As illustrated in the code below:

```
//guassian exponent values for carbon
double c_a1 = 71.61683735;
double c_a2 = 13.04509632;
double c_a3 = 3.53051216;
```

```
//contraction coefficient values for carbon
double c_c1 = 0.1543289673;
double c_c2 = 0.5353281423;
double c_c3 = 0.4446345422;

//guassian integral values for oxigen
double o_a1 = 130.7093200;
double o_a2 = 23.80886605;
double o_a3 = 6.443608313;

//contraction coefficient values for oxigen
double o_c1 = 0.1543289673;
double o_c2 = 0.5353281423;
double o_c3 = 0.4446345422;

//Laplacian matrix element values for carbon
c_Lap_1 = y_feta(-1, c_a1)*y_gaussian_integral(c_a1, c_a1);
c_Lap_2 = y_feta(-1, c_a2)*y_gaussian_integral(c_a2, c_a2);
c_Lap_3 = y_feta(-1, c_a3)*y_gaussian_integral(c_a3, c_a3);
```

4.2 Computing the electron repulsion term

We now compute the electron repulsion term. We use the following code to do so:

```
//construct the QC
double QC = (((y_feta(c_c1, c_a1))^2)*
            y_gaussian_integral(c_a1,c_a1) )+
            (2*y_feta(c_c2, c_a2)*y_feta(c_c1, c_a1))*
            y_gaussian_integral(c_a2,c_a1)+
            (2*y_feta(c_c3, c_a3)*y_feta(c_c1, c_a1))*
            y_gaussian_integral(c_a3,c_a1)+
            (((y_feta(c_c2, c_a2))^2)*
            y_gaussian_integral(c_a2,c_a2))+
            (((y_feta(c_c3, c_a3))^2)*
            y_gaussian_integral(c_a3,c_a3))+
            (2*y_feta(c_c2, c_a2)*y_feta(c_c3, c_a3))*
            y_gaussian_integral(c_a2,c_a3);

//construct the QO
double QO = (((y_feta(o_c1, o_a1))^2)*
            y_gaussian_integral(o_a1,o_a1) )+
            (2*y_feta(o_c2, o_a2)*y_feta(o_c1, o_a1))*
            y_gaussian_integral(o_a2,o_a1)+
            (2*y_feta(o_c3, o_a3)*y_feta(o_c1, o_a1))*
            y_gaussian_integral(o_a3,o_a1)+
            (((y_feta(o_c2, o_a2))^2)*
            y_gaussian_integral(o_a2,o_a2))+
            (((y_feta(o_c3, o_a3))^2)*
            y_gaussian_integral(o_a3,o_a3))+
            (2*y_feta(o_c2, o_a2)*y_feta(o_c3, o_a3))*
            y_gaussian_integral(o_a2,o_a3);
double ROC =2.2620020083;
double coulomb_interaction = y_coulomb_interaction(QC, QO, ROC);
```

4.3 Computing the exchange potential

We now compute the exchange potential. We use the following code to do so:

```
//bhor radius of the 3 atoms
double rs[] = {3.2125341806,2.8723835026,3.2125341806};

double r_s[] = {0.340150678,0.340150678};

//calculate volume of the system

//overall volume of the system
double volume_total = y_volumes(rs);

//correction volume
double volume_correction = y_volumes(r_s);

double volume = volume_total - volume_correction;

//calculate the density of the system
double density = (6+8+8)/volume;

//calculate the exchange potential
double exchange_potential = y_exchange_potential(density);
```

5 nuclear attraction term

We now compute the nuclear attraction term. We use the following code to do so:

```
    //calculate the exchange potential
    double exchange_potential = y_exchange_potential(density);

    //ccalculate the nuclear attraction term

    double r1c = (3/4)*3.2125341806;
    double r1o = (3/4)*2.8723835026;

    double ena_potential = 2*(ena(6,r1c)) + ena(8,r1o);
```