Modelling Radial Electron Densities of Atoms in Molecules

By

Devin G. S. Nippard

A thesis submitted to the School of Graduate Studies
in partial fulfilment of the requirements for the degree of
Master of Science (Computational Science)
The Department of Chemistry
Memorial University of Newfoundland

May, 2014

St. John's, Newfoundland

Abstract

This thesis looks at fitting the radial density of atoms in molecules using a least squares minimization. As proof of concept the radial electron densities of individual atoms of the first three periods were fitted using a linear combination of six Gaussian functions. The core electrons of the individual atom fits were used as an initial guess for the core electrons of the atoms in molecule fit while the now non-spherically symmetric valence shells of the atoms in molecules were fitted using non-nuclear centred Gaussian functions. The radial densities of atoms in molecules for each molecular orbital were also fitted. The fits were computed using Mathematica. The thesis shows that it is possible to use the radial electron density of the atoms in molecule fits to produce/reproduce molecular electron densities.

Acknowledgments

I would like to thank Dr. Raymond Poirier and Dr. Peter Warburton and their Theoretical/Computational Chemistry group for all their help and support on this Masters research project.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

A very basic concept of chemistry is that molecules are made up of atoms and the molecular properties are derived from the properties of the individual atoms. The properties of an atom in a molecule will be different than the atomic properties of individual free atoms. This is the theory of atoms in molecules (AIM).

A property of atoms and molecules is electron density. Electron density can be obtained experimentally from X-ray crystal diffraction or computationally using Wavefunction Theory or Density Functional Theory (DFT). In DFT the density is found by minimizing a density functional with respect to energy, whereas in wavefunction theory, such as Hartree-Fock, a wavefunction is found by solving the Schrödinger equation:

$$\hat{H}\Psi(r) = E\Psi(r) \tag{1.1.1}$$

where $\hat{H}$ is the Hamiltonian, $\Psi(r)$ is the wavefunction and E is the total energy of the system. Once you solve for the wavefunction, in Hartree-Fock theory, the electron density $\rho(r)$ can be expressed as:

$$\rho(r) = 2\sum_{a}^{N/2} \psi_a(r)\psi_a^*(r) \tag{1.1.2}$$

where the sum is over the molecular orbitals $\psi_a(r)$.

From the theory of AIM, the electron density of a molecule can be partitioned into atomic contributions (Equation 1.1.3). This thesis explores how to fit these atomic electron density contributions. The fits will then be stored for many atoms in molecules so that they may be used to build molecular electron densities (Equation 1.1.4).

$$\rho^{HF}(r) \rightarrow \sum_{A} \rho_A(r) \tag{1.1.3}$$

$$\rho^{fit}(r) \leftarrow \sum_{A} \rho_A^{fit}(r) \tag{1.1.4}$$

In Equation 1.1.3 $\rho^{HF}(r)$ is the electron density of the molecule found using Hartree-Fock Theory, while ρ_A is the atomic contribution of the A^{th} atom. The fits used to build a molecule would require that they be of atoms from a similar chemical environment (what other atoms does the atom in a molecule interact with, what atoms are it bonded too). For example, if you wish to construct a ketone (for instance propanone), the carbon atom that bonds to the oxygen should be at minimum a carbon atom double bonded to an oxygen atom and two other like atoms, ideally, in this case, two carbons.

1.2 Radial Electron Density

The radial electron density of an atom is defined as:

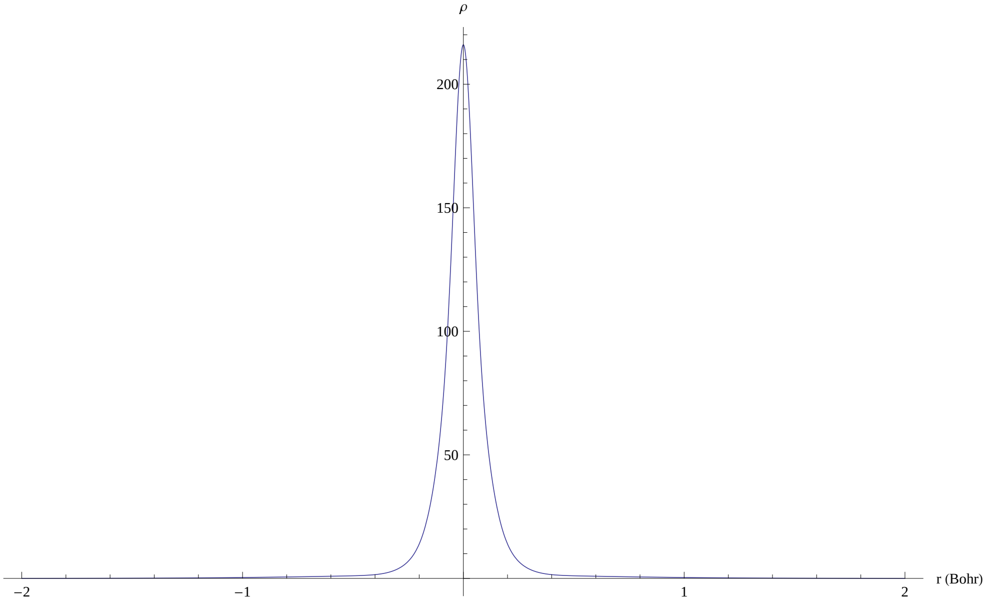
$$\rho_A^{rad}(r) = r_A^2 \rho_A(r) \tag{1.2.1}$$

where r_A is the distance from some point to atom A, and $\rho_A(r)$ is the density at that point. When looking at a molecule, however, the radial density is not as simply defined. Which center do you choose to calculate r_A? A simple solution to this problem comes from AIM theory. We know that the electron density of a molecule is the sum of the individual atomic electron densities. The radial electron density of a molecule is defined as:

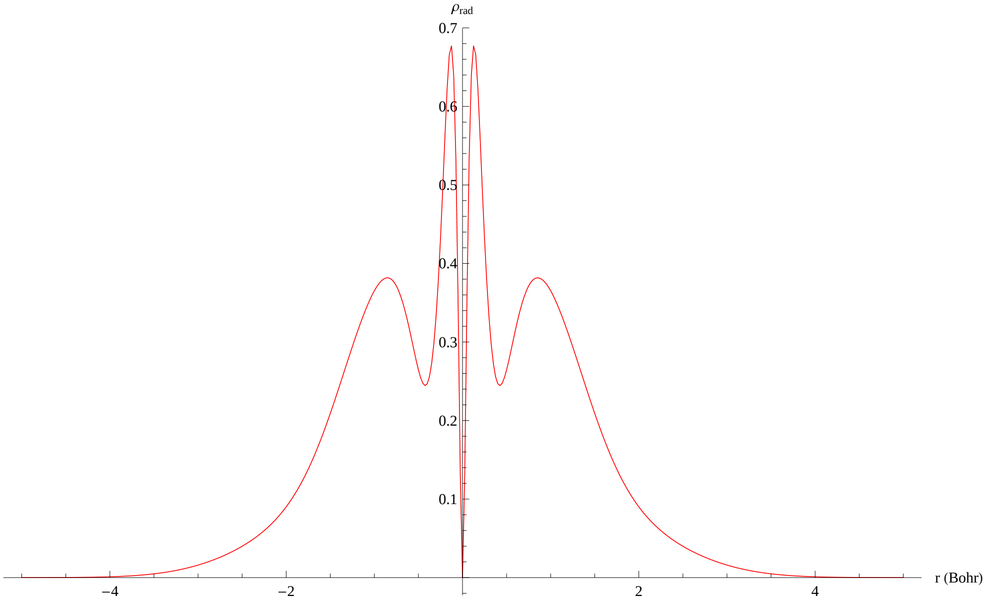
$$\rho^{rad}(r) = \sum_A r_A^2 \rho_A(r) \tag{1.2.2}$$

which shows that if the molecular electron density can be partitioned into their atomic contributions, the molecular radial electron density can be calculated.

Why use radial electron densities? From Figure 1.2.1a one can see that the electron density of an oxygen atom has a very large sharp peak at the nucleus. The density quickly becomes zero as you move away from the nucleus. These two features make it a challenge to fit, as most least squares fitting programs will focus on fitting the large peak. The small numbers (less than ten) of the valence shell region have little consequence when compared to the large numbers of the core (core values range from 100-10000, increasing as atomic number increases) to the numbers rapidly approaching zero as r increases. This is a problem since the most interesting regions of atoms in molecules are the valence regions where most of the bonding occurs. From Figure 1.2.1b we see the radial electron density of oxygen. The first thing to note is that it is no longer a single large peak at the nucleus. Instead there is a sharp peak close to the nucleus to represent the core electrons, and a more diffuse peak to represent the valence electrons beyond the core peak. So not only does the radial electron density have more of the character of the atom, but it should be easier to fit the valence electrons now that the radial electron density is more than one sharp peak at the nucleus. Another example showing the differences in electron and radial electron density is seen by comparing Figure 1.2.1a and 1.2.2a. Both graphs are almost identical with their only difference being the maximum reached by the peaks. Now consider Figure 1.2.1b and 1.2.2b. Oxygen only has one peak while chlorine has two sharp core peaks close to the nucleus. The valence shell of oxygen is sharper and closer to the nucleus than chlorine's valence shell which is more diffuse.

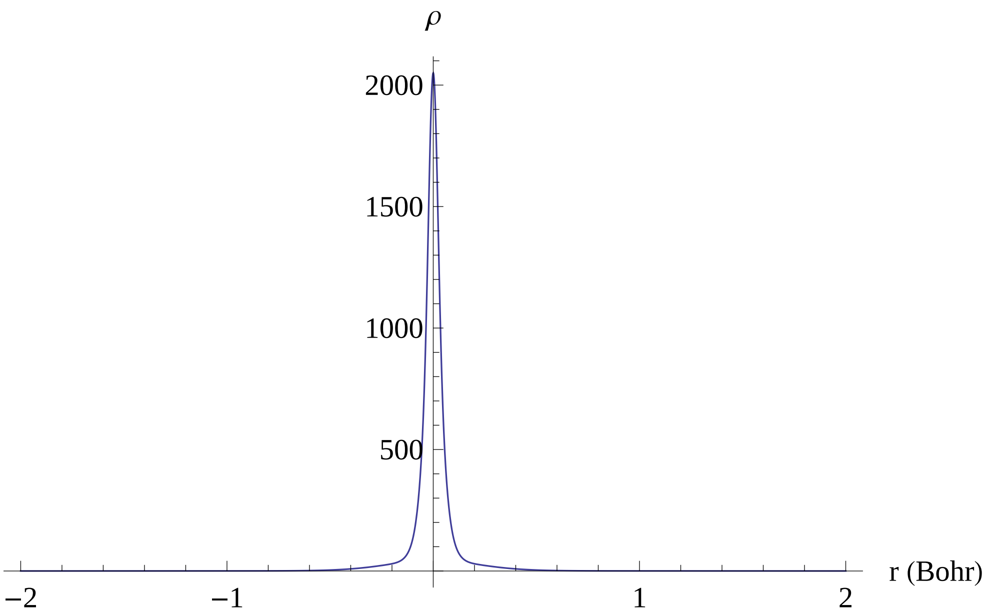


(a) Electron Density of O

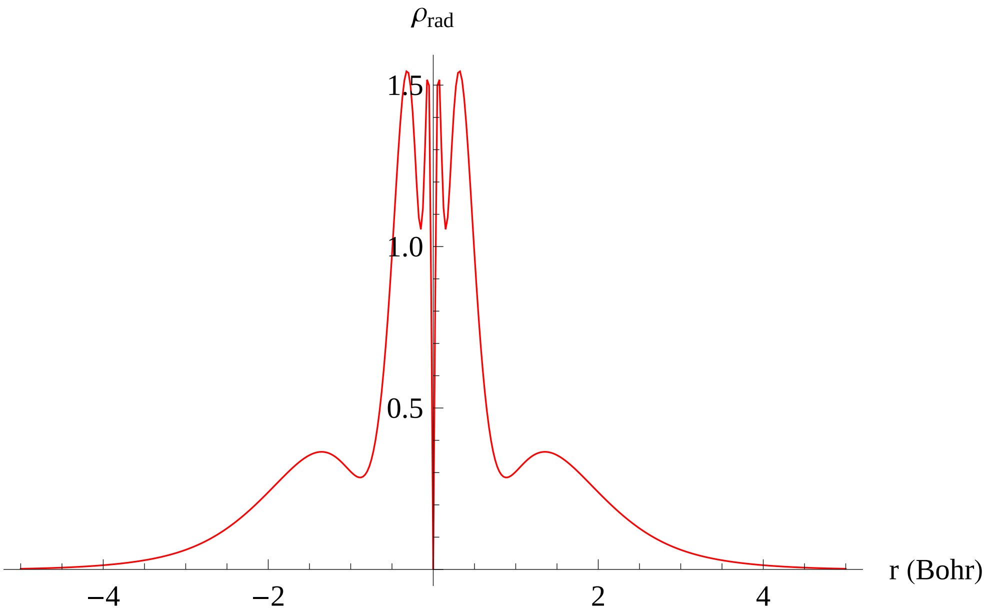


(b) Radial Electron Density of O

Figure 1.2.1: Electron density (a) and radial electron density (b) of an oxygen atom.



(a) Electron Density of Cl



(b) Radial Electron Density of Cl

Figure 1.2.2: Electron density (a) and radial electron density (b) of a chlorine atom.

1.3 Partition Schemes

A partitioning scheme is used to determine what portion of a molecular electron density belongs to an individual atom in that molecule. The atomic radial electron density of an atom in molecule is defined as:

$$\rho_A^{rad}(r) = W_A(r)\rho^{rad}(r) \tag{1.3.1}$$

where $W_A(r)$ is the weight or amount of radial density at point r belonging to atom A. In this case how one defines the weight is the partition scheme, though it should be noted that a weight is not needed to partition a molecule into its atomic contributions. An example of this is Bader's Quantum Theory of Atoms In Molecules. This is discussed in more detail in Section 1.4. In the case of the Iterated Stockholder Approach (ISA) [2] two equations that define the density of atom A and the weight of atom A are solved simultaneously and iteratively:

$$\rho_A(r) = \rho(r)\frac{w_A(r)}{\sum\limits_B w_B(r)} \tag{1.3.2}$$

$$w_A(r) = \langle \rho_A(r) \rangle_A \tag{1.3.3}$$

In Equation 1.3.1 $W_A(r)$ is defined as:

$$W_A(r) = \frac{w_A(r)}{\sum\limits_B w_B(r)} \tag{1.3.4}$$

In Equations 1.3.2 and 1.3.3 the densities are spherically symmetric and the density functions are normalized to the number of electrons. In Equation 1.3.2 the weight (or partitioning function) of atom A is found by dividing the weight function of A ($w_A(r)$) by the sum of all the weight functions of all the atoms in that molecule ($\sum\limits_B w_B(r)$). In ISA the weight functions are found by spherically averaging the electron density of atom A. This is Equation 1.3.3.

Another way to define the weight of atom A is to use the Becke weight formulation [3]. In the Becke weight formulation the weight of atom A is defined as it is in Equation 1.3.4 where $w_A(r)$ is the cell function of the A^{th} atom. The cell function can be thought of as a polyhedron that is made of the perpendicular bisectors of the vectors connecting atom A to every other atom. This leads to defining $w_A(r)$ as:

$$w_A(r) = \prod_{B \neq A} s(\mu_{AB}) \tag{1.3.5}$$

where μ_{AB} is one of the set of elliptical coordinates Becke uses to describe the boundary of the polyhedron. The range of μ_{AB} is between and including negative one and one. The coordinate μ_{AB} is defined as:

$$\mu_{AB} = \frac{r_{iA} - r_{iB}}{R_{AB}} \tag{1.3.6}$$

where R_{AB} is the distance between atom A and atom B, r_{iA} and r_{iB} are the distances to some grid point i from atom A and atom B respectively.

In Equation 1.3.5, $s(\mu_{AB})$ is a step function that Becke defines as:

$$s(\mu_{AB}) = \frac{1}{2}(1 - f(\mu_{AB})) \tag{1.3.7}$$

where $f(\mu_{AB})$ is the polynomial:

$$f(\mu_{AB}) = \frac{3}{2}\mu_{AB} - \frac{1}{2}\mu_{AB}^3 \tag{1.3.8}$$

The polynomial in Equation 1.3.8 defines a step function that is not steep enough for a boundary between atoms, so Becke defines the new function $h_i(\mu_{AB})$ as:

$$\begin{aligned} h_1(\mu_{ij}) &= f(\mu_{AB}) \\ h_2(\mu_{ij}) &= f(f(\mu_{AB})) \\ h_3(\mu_{ij}) &= f(f(f(\mu_{AB}))) \\ &\vdots \end{aligned} \tag{1.3.9}$$

Becke states that $h_3(\mu_{ij})$ is a steep enough step function. Substituting the new step function (h_3) into Equation 1.3.7:

$$s(\mu_{AB}) = \frac{1}{2}(1 - h_3(\mu_{ij})) = \frac{1}{2}(1 - f(f(f(\mu_{AB})))) \tag{1.3.10}$$

The Becke weight formulation previously presented is for homonuclear species. The general formulation for heteronuclear species replaces the μ_{AB} with ν_{AB} which is defined as:

$$\nu_{AB} = \mu_{AB} + a_{AB}(1 - \mu_{AB}^2) \tag{1.3.11}$$

where a_{AB} is defined as:

$$a_{AB} = \frac{u_{AB}}{u_{AB}^2 - 1} \tag{1.3.12}$$

where u_{AB} is defined as:

$$u_{AB} = \frac{\chi_{AB} + 1}{\chi_{AB} - 1} \tag{1.3.13}$$

where χ_{AB} is the ratio of the atomic radii of atom A and B respectively and can be written as:

$$\chi_{AB} = \frac{R_A}{R_B} \tag{1.3.14}$$

In his paper Becke [3] suggests using Bragg-Slater radii.

The Becke weight fails if the radii is greater than 2.4, in this case a_{AB} would fall outside its allowed range of $\pm 1/2$. This keeps $s(\mu_{AB})$ monotonic when transformed to $s(\nu_{AB})$. When using Bragg-Slater radii this happens mostly with atoms paired with hydrogen, which has a Bragg-Slater radius of 0.25. Becke suggests, as a solution, to increase the radius of hydrogen to 0.35, and in any cases where the ratio of the radii is greater than 2.4 or less than 0.42 (its inverse) then the ratio be set to that maximum or minimum.

1.4 Previous Work

There are a few schemes that partition molecules into their atomic contribution, these contributions are then studied to help further understand molecules and their properties. As mentioned in section 1.3 one can define a weight to partition a molecule and in those cases how the weight is defined is the partition scheme. There are also partition schemes that do not depend on defining a weight. One of these schemes is Bader's quantum theory of atoms in molecules (QTAIM) [4]. In QTAIM the electron density is partitioned by defining zero-flux surfaces. Zero-flux surfaces are found by following gradient paths, and can be viewed using contour graphs of electron density. The gradient of the electron density is:

$$\nabla\rho(x,y,z) = \frac{\partial\rho}{\partial x}\hat{\imath} + \frac{\partial\rho}{\partial y}\hat{\jmath} + \frac{\partial\rho}{\partial z}\hat{k} \tag{1.4.1}$$

where $\hat{\imath}$,$\hat{\jmath}$ and $\hat{k}$ are the unit vectors pointing in the x,y,z direction respectively. The vector described by the gradient points to the largest increase in density, which when followed, points to a maximum in electron density. Critical points are points in which the gradient is zero ($\nabla\rho(x,y,z) = 0$). One type of critical point occurs when electron density is a maximum. They are called nuclear attractors. Another important critical point is the bond critical point. This is the point where the gradient of the electron density goes to zero between two atoms, as seen in Figure 1.4.1, which shows the bond critical point between the oxygen and carbon atoms in CH_2O.

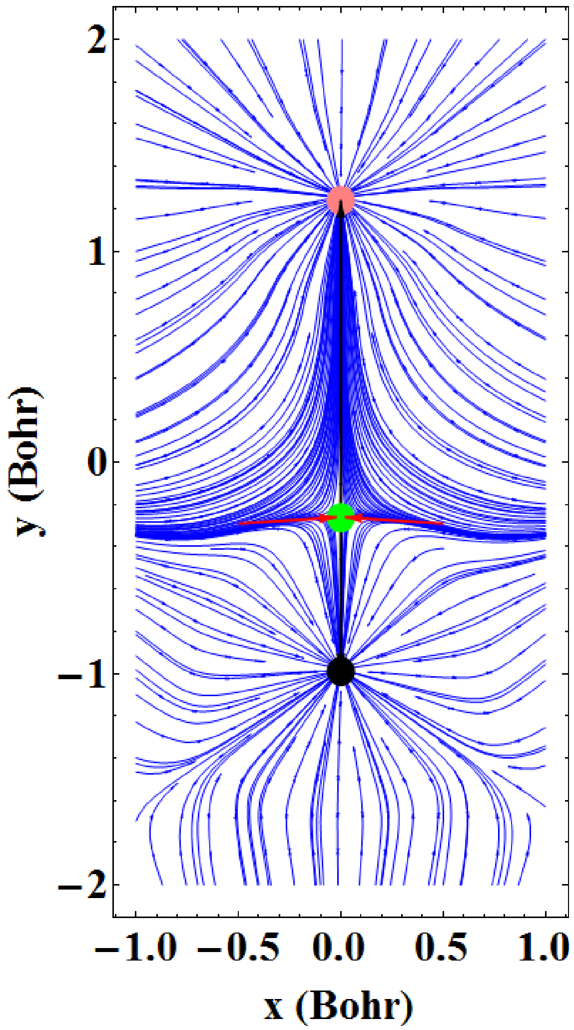


Figure 1.4.1: This Figure was taken from Jessica Besaw's Honours Thesis [1].The bond critical point (green) seen between a carbon (black) and oxygen (pink) in CH_2O os located where the gradients paths meet (red arrows). The atomic interaction line is denoted by black arrows.

The critical points help describe the properties of the atoms in molecules. The gradient paths and the contour map help to identify these boundaries as in Figure 1.4.2. For example, imagine the boundary as a barrier along the line between oxygen and the rest of the molecule. All of the density on the oxygen side of the barrier belongs to the oxygen atom, while none of the the density beyond the barrier belongs to oxygen. The boundary itself is made up of interatomic surfaces that separate the atoms. This of course means that atoms in QTAIM do not overlap.

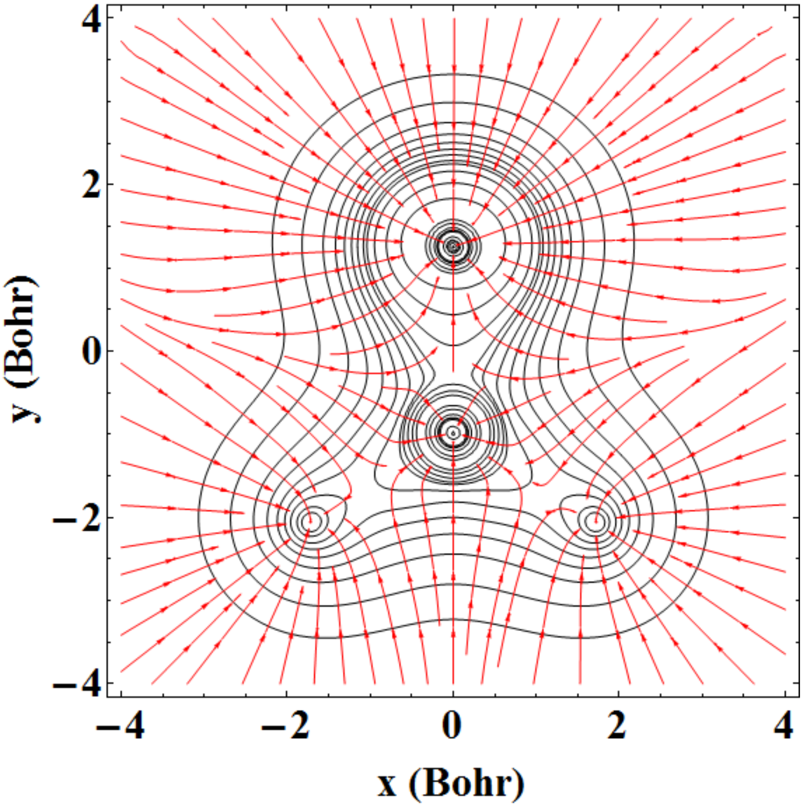


Figure 1.4.2: This Figure was taken from Jessica Besaw's Honours Thesis [1]. The gradient paths of electron density superimposed upon its contour plot in the symmetry plane of CH_2O.

As was mentioned, there are a few types of critical points, the critical points are identified by the rank of the hessian, which is:

$$\nabla\nabla\rho(r) = \begin{pmatrix} \frac{\partial^2\rho}{\partial x^2} & \frac{\partial^2\rho}{\partial x\partial y} & \frac{\partial^2\rho}{\partial x\partial z} \\ \frac{\partial^2\rho}{\partial y\partial x} & \frac{\partial^2\rho}{\partial y^2} & \frac{\partial^2\rho}{\partial y\partial z} \\ \frac{\partial^2\rho}{\partial z\partial x} & \frac{\partial^2\rho}{\partial z\partial y} & \frac{\partial^2\rho}{\partial z^2} \end{pmatrix} \tag{1.4.2}$$

The rank is the number of nonzero eigenvalues of the hessian. The critical points are also identified by their signature which comes from the Laplacian:

$$\nabla^2\rho(r) = \frac{\partial^2\rho}{\partial x^2} + \frac{\partial^2\rho}{\partial y^2} + \frac{\partial^2\rho}{\partial z^2} = \lambda_1 + \lambda_2 + \lambda_3 \tag{1.4.3}$$

where λ_i,$i \in 1, 2, 3$ are the eigenvalues of the hessian.

In QTAIM, molecules are partitioned into atomic contributions, The Atomic Orbitals of the Topological Atom [5], expands on QTAIM by creating molecular orbitals from linear combinations of atomic orbitals. To do this imagine a system of n doubly occupied orbitals, and a three dimensional space split up into N^A atomic domains denoted by D^A, a n x n Hermitian matrix can be formed with the elements:

$$Q^A_{ij} = \int\limits_{D^A} \varphi^*_i(r)\varphi_j(r)\mathrm{d}v \tag{1.4.4}$$

where $\varphi^*_i(r)$ and $\varphi_j(r)$ are molecular orbitals, making the matrix Q^A an overlap matrix. The molecular orbitals can be broken into intratomic parts such that:

$$\varphi^A_i(r) = \begin{cases} \varphi_i(r) & \text{if } r \in D^A \\ 0 & \text{if } r \notin D^A \end{cases} \tag{1.4.5}$$

The matrix Q^A can be diagonalized using the unitary matrix U^A. The nonzero diagonal elements of the transformed Q^A are used to define the basis set ("effective atomic orbitals"):

$$\chi_\mu^A(r) = \frac{1}{\sqrt{\lambda_\mu^A}} \sum_{i=1}^{n} U_{i\mu}^A \varphi_i^A(r) \tag{1.4.6}$$

where λ_μ^A are the nonzero diagonals of the transformed matrix Q^A. The index μ goes from one to n_A, the number of nonzero diagonals. The basis set produces orbitals with occupancy numbers that sum to give the correct number of electrons for the atoms as seen by QTAIM. The basis set may also be used to produce the correct bond orders.

Chapter 2

Computational Method

All densities and radial densities were generated on brasdor server of ACEnet using a copy of MUNgauss code [6]. Codes produced were written in Fortran90 and implemented using MUNgauss. The densities were calculated at HF/6-31G(d). All graphs and fits were generated using Mathematica 8.

Chapter 3

Mathematical Development

3.1 Least Squares Fit

The proposed least squares fit problem is:

$$(\rho_{HF}(r) - \sum_{A} \rho_A(r))^2 \tag{3.1.1}$$

where $\rho_{HF}(r)$ is the molecular electron density obtained using HF theory, and $\rho_A(r)$ is the fitted density of atom A. The fitted density sums to give the molecular electron density. As mentioned before, radial electron densities will be used which changes Equation 3.1.1 to:

$$\int (\sum_{A} r_A^2 W_A(r) \rho_{HF}(r) - \sum_{A} r_A^2 \rho_A(r))^2 \mathrm{d}r \tag{3.1.2}$$

In Equation 3.1.2 $W_A(r)$ is how the molecule is partitioned and in our case is from the Becke weight formulation as discussed in Section 1.3. Now we want to impose the constraint that the sum of the number of electrons of the atoms equals the total number of electrons (N) which can be expressed as:

$$\sum_A \int_0^\infty r_A^2 \rho_A(r) \mathrm{d}r - N = 0 \tag{3.1.3}$$

One method for solving a least squares problem is the Lagrange Undetermined Multiplier Method. The first step is to set the Lagrange expression, which in this case is:

$$\mathcal{L} = \int (\sum_A r_A^2 W_A(r) \rho_{HF}(r) - \sum_A r_A^2 \rho_A(r))^2 \mathrm{d}r + \lambda(\int_0^\infty \sum_A r_A^2 \rho_A(r) \mathrm{d}r - N) \tag{3.1.4}$$

Now expanding $\rho_A(r)$ as:

$$\rho_A(r) = \sum_a \eta_a^A |\psi_a^A(r)|^2 \tag{3.1.5}$$

where $\psi_a^A(r)$ is the molecular orbital a on atom A and η_a^A is the occupancy of that orbital. The Lagrange expression becomes:

$$\begin{split} \mathcal{L} = \int (\sum_A r_A^2 W_A(r) \rho_{HF}(r) - \sum_A r_A^2 \sum_a \eta_a^A |\psi_a^A(r)|^2)^2 \mathrm{d}r \\ + \lambda(\int_0^\infty \sum_A r_A^2 \sum_a \eta_a^A |\psi_a^A(r)|^2 \mathrm{d}r - N) \end{split} \tag{3.1.6}$$

To solve using the Lagrange method, the partial derivative is calculated and set to zero:

$$\begin{aligned}\delta\mathcal{L} = 2\eta_b^B\delta\psi_b^{B*}(r)\int[r_A^2(W_A(r)\rho_{HF}(r) - \sum_a \eta_a^A|\psi_a^A(r)|^2)\psi_b^B(r)]\mathrm{d}r \\ -\eta_b^B\lambda'\int r_B^2\delta\psi_b^{B*}(r)\psi_b^B\mathrm{d}r \\ +2\eta_b^B\delta\psi_b^B(r)\int[r_A^2(W_A(r)\rho_{HF}(r) - \sum_a \eta_a^A|\psi_a^A(r)|^2)\psi_b^{B*}(r)\mathrm{d}r] \\ -\eta_b^B\lambda'\int r_B^2\delta\psi_b^B(r)\psi_b^{B*}\mathrm{d}r = 0\end{aligned} \tag{3.1.7}$$

Rearranging Equation 3.1.7 becomes:

$$2\eta_b^B\delta\psi_b^{B*}(r)\int[r_A^2(W_A(r)\rho_{HF}(r) - \sum_a \eta_a^A|\psi_a^A(r)|^2) - r_B^2\lambda']\psi_b^B(r)\mathrm{d}r + cc = 0 \tag{3.1.8}$$

It should be noted that in Equation 3.1.8 the derivative is for some orbital b on some atom B, and that λ' is $\lambda/2$. Ignoring the complex conjugate the equation can be rearranged to give the eigenvalue equation:

$$r_A^2(W_A(r)\rho_{HF}(r) - \sum_a \eta_a^A|\psi_a^A(r)|^2)\psi_b^B(r) = \lambda' r_B^2\psi_b^B(r) \tag{3.1.9}$$

Next we define an operator $\hat{f}$ as:

$$\hat{f} = r_A^2(W_A(r)\rho_{HF}(r) - \sum_a \eta_a^A|\psi_a^A(r)|^2) \tag{3.1.10}$$

Equation 3.1.9 can now be written as:

$$\hat{f}\psi_b^B(r) = \lambda' r_B^2 \psi_b^B(r) \tag{3.1.11}$$

Introducing the basis set:

$$\psi_b^B(r) = \sum_\lambda C_{\lambda b}^B \chi_\lambda^B(r) \tag{3.1.12}$$

and then integrating and multiplying though by χ_λ^{B*}, Equation 3.1.11 becomes:

$$\sum_\lambda C_{\lambda b}^B \int \chi_\sigma^{B*}(r)\hat{f}\chi_\lambda^B(r)\mathrm{d}r = \sum_\lambda C_{\lambda b}^B \lambda' \int \chi_\sigma^{B*}(r)\chi_\lambda^B(r)\mathrm{d}r \tag{3.1.13}$$

Expanding $\hat{f}$ we have:

$$\sum_\lambda C_{\lambda b}^B \int [\chi_\sigma^{B*}(r) r_A^4 (W_A(r)\rho_{HF}(r) - \sum_a \eta_a^A \psi_a^A(r)\psi_a^{A*}(r))\chi_\lambda^B(r)]\mathrm{d}r \tag{3.1.14}$$

This equation can be broken into a part that contains the Hartree-Fock density and individual atom densities. Expressing the Hartree-Fock density as molecular orbitals and introducing the basis set, the first integral becomes:

$$r_A^2 W_A(r) \int \chi_\sigma^{B*}(r)(\sum_\alpha \sum_\beta C_{\alpha c}^C C_{\beta d}^D \chi_\alpha^C(r)\chi_\beta^{D*}(r))\chi_\lambda^B(r)\mathrm{d}r \tag{3.1.15}$$

The integral in Equation 3.1.15 is similar to an overlap integral with four centers instead of the usual two, so it is written in a similar notation to the overlap integral. We now have the Hartree-Fock contribution to the left hand side of Equation 3.1.13

as:

$$F_{HF} = r_A^2 W_A P_{\alpha\beta}^{CD} S_{\alpha\beta\lambda\sigma}^{CDBB} \tag{3.1.16}$$

The second integral obtained after expanding $\hat{f}$ of Equation 3.1.13, involves the individual atom densities of the atoms in molecules, unlike the first integral of Equation 3.1.14 that involed the Hartree-Fock molecular density. The molecular orbitals can be expressed in terms of the basis set as:

$$r_A^2 \int \chi_\sigma^{B*}(r)(\sum_a \eta_a^A \sum_\mu \sum_\nu C_{\mu a}^A C_{\nu a}^A \chi_\mu^A(r) \chi_\nu^{A*}(r)) \chi_\lambda^B(r) \mathrm{d}r \tag{3.1.17}$$

Now Equation 3.1.17 also has an overlap like integral with four centers instead of two and thus can be written as:

$$F_{Fit} = r_A^2 P_{\mu\nu}^{AA} S_{\mu\nu\lambda\sigma}^{AABB} \tag{3.1.18}$$

From Equations 3.1.16 and 3.1.18 the integral part of Equation 3.1.14 can be written as:

$$F = F_{HF} - F_{Fit} \tag{3.1.19}$$

The integral on the right side of Equation 3.1.13 is the overlap. Equation 3.1.13 in matrix notation is:

$$FC = \lambda' CS \tag{3.1.20}$$

which is similar to Roothaan's equation with four center integrals.

3.2 Gaussian Product and Multi-Centered Integrals

One of the useful properties of Gaussian functions is that the product of two Gaussian functions is a Gaussian function. If we have a general Gaussian function centered at A:

$$G_A = N_A x_A^l y_A^m z_A^n \mathrm{e}^{-\alpha r_A^2} \tag{3.2.1}$$

where N_A is a constant, α is an exponent. x_A^l, y_A^m, z_A^n are the angular components that depend on the quantum numbers l,m,n. The sum of the quantum numbers l,m,n determines the type of function. When the quantum numbers sum to zero it is an s-type, when the sum is one it's a p-type, and so on. In Equation 3.2.1 r_A is:

$$r_A = r - A \tag{3.2.2}$$

where A represents the coordinates of center A. To simplify how the new center is found we initially choose an s-type, making Equation 3.2.1:

$$G_A = N_A \mathrm{e}^{-\alpha r_A^2} \tag{3.2.3}$$

If we multiply this function by a similar Gaussian function centered at B then the resulting function is another Gaussian function centered at P. Our new Gaussian function is:

$$G_P = N_A N_B \mathrm{e}^{-\alpha r_A^2} \mathrm{e}^{-\beta r_B^2} \tag{3.2.4}$$

which can be rewritten as:

$$G_P = N_A N_B K \mathrm{e}^{-\gamma r_P^2} \tag{3.2.5}$$

where K and γ are found when multiplying out the exponent portion of Equation 3.2.4 and can be shown to be:

$$K = exp\left[\frac{-\alpha\beta}{\alpha+\beta}(A-B)^2\right] \tag{3.2.6}$$

$$\gamma = \alpha + \beta \tag{3.2.7}$$

r_P from Equation 3.2.5 can be expressed the same way as r_A in Equation 3.2.2 by replacing the A with P, which is defined as:

$$P = \frac{\alpha A + \beta B}{\alpha + \beta} \tag{3.2.8}$$

Now to show how the angular part of the Gaussian functions behaves, consider a Gaussian function with m and n quantum numbers set to zero. With any non-zero number for l, equation 3.2.1 becomes:

$$G_A = N_A x_A^l \mathrm{e}^{-\alpha r_A^2} \tag{3.2.9}$$

where x_A is:

$$x_A = x - A_x = (x - P_x) + (P_x - A_x) = x_A + \overline{PA_x} \tag{3.2.10}$$

where P_x and A_x are the x coordinates of centers P and A respectively. Now that x_A is known x_A^l can be shown to be:

$$x_A^l = \sum_{i=0}^{l} \overline{PA_x}^{l-i} \binom{l}{i} x_P^i \tag{3.2.11}$$

where $\binom{l}{i}$ is the binomial coefficient which can be expressed mathematically as:

$$\binom{l}{i} = \frac{l!}{i!(l-i)!} \tag{3.2.12}$$

In the case of the product of two Gaussian functions, we would obtain a G_P similar to the one in Equation 3.2.5 with the added product of x_A^l and x_B^o which is expressed as:

$$x_A^l x_B^o = \sum_{i=0}^{l} \sum_{j=0}^{o} \overline{PA_x}^{l-i} \overline{PB_x}^{o-j} \binom{l}{i} \binom{o}{j} x_P^{i+j} \tag{3.2.13}$$

Now lets introduce the integer k such that:

$$k = i + j \tag{3.2.14}$$

Now Equation 3.2.13 becomes:

$$x_A^l x_B^o = \sum_{k=0}^{l+o} \left[\sum_{i=0}^{l} \sum_{j=0}^{o} \overline{PA_x}^{l-i} \overline{PB_x}^{o-j} \binom{l}{i} \binom{o}{j} \right] x_P^k \tag{3.2.15}$$

The portion of Equation 3.2.15 in square brackets is a function of l,o,PA_x,PB_x for each value of k. Equation 3.2.15 can now be written as:

$$x_A^l x_B^o = \sum_{k=0}^{l+o} f_{k^x}(l, o, PA_x, PB_x) x_P^{k^x} \tag{3.2.16}$$

Now the product of two general Gaussian functions at two different centers is:

$$G_A G_B = G_P = N_A N_B K \sum_{k^x=0}^{l+o} \sum_{k^y=0}^{m+p} \sum_{k^z=0}^{n+q} f_{k^x} f_{k^y} f_{k^z} x_P^{k^x} x_P^{k^y} x_P^{k^z} \mathrm{e}^{-\gamma r_P^2} \tag{3.2.17}$$

The overlap of two Gaussian functions centered at A and B is the integral of Equation 3.2.17, which can be broken down into the product of three equations of the form:

$$F_k = \int_{-\infty}^{\infty} t^k \mathrm{e}^{-\gamma t^2} \mathrm{d}t \tag{3.2.18}$$

The only nonzero solution to Equation 3.2.18 is when k is an even integer. Equation 3.2.18 can be rewritten as:

$$F_k = 2 \int_{0}^{\infty} t^k \mathrm{e}^{-\gamma t^2} \mathrm{d}t \tag{3.2.19}$$

which when solved for even k gives:

$$F_k = \gamma^{-(k+1)/2} \frac{(k-1)!!}{2^{k/2}} \sqrt{\pi} \tag{3.2.20}$$

This gives the overlap integral of two Gaussian functions centered at A and B (S_{AB}):

$$\begin{aligned} S_{AB} = N_A N_B K \sum_{k^x=0}^{(l+o)/2} \sum_{k^y=0}^{(m+p)/2} \sum_{k^z=0}^{(n+q)/2} & f_{2k^x} f_{2k^y} f_{2k^z} \gamma^{(2k^x+2k^y+2k^z+3)/2} \\ & * \frac{(2k^x-1)!!}{2^{k^x/2}} \frac{(2k^y-1)!!}{2^{k^y/2}} \frac{(2k^z-1)!!}{2^{k^z/2}} \pi^{3/2} \end{aligned} \tag{3.2.21}$$

To solve a four centered integral of the centers A,B,C,D the procedure for the Gaussian product can be followed. First create a new center at P using centers A and B.

$$G_A G_B = G_P = N_A N_B K_P \sum_{k_P^x=0}^{P_x} \sum_{k_P^y=0}^{P_y} \sum_{k_P^z=0}^{P_z} f^P_{k_P^x} f^P_{k_P^y} f^P_{k_P^z} x_P^{k_P^x} x_P^{k_P^y} x_P^{k_P^z} \mathrm{e}^{-\gamma^P r_P^2} \tag{3.2.22}$$

Then create second new centers Q using center C and D.

$$G_C G_D = G_Q = N_C N_D K_Q \sum_{k_Q^x=0}^{Q_x} \sum_{k_Q^y=0}^{Q_y} \sum_{k_Q^z=0}^{Q_z} f^Q_{k_Q^x} f^Q_{k_Q^y} f^Q_{k_Q^z} x_Q^{k_Q^x} x_Q^{k_Q^y} x_Q^{k_Q^z} \mathrm{e}^{-\gamma^Q r_Q^2} \tag{3.2.23}$$

Next create another center S from centers P and Q, the same way that P and Q were created. The new expoent γ_S is defined as:

$$\gamma^S = \gamma^P + \gamma^Q \tag{3.2.24}$$

The new center can be defined as:

$$S = \frac{\gamma^P P + \gamma^Q Q}{\gamma^P + \gamma^Q} \tag{3.2.25}$$

the expanding the exponent portion of the equation gives:

$$K^S = exp\left[\frac{-\gamma^P \gamma^Q}{\gamma^P + \gamma^Q}(P-Q)^2\right] \tag{3.2.26}$$

The angular part of G_S is:

$$x_P^{k_P^x} x_Q^{k_Q^x} = \sum_{k=0}^{k_P^x + k_Q^x} \left[\sum_{i=0}^{k_P^x} \sum_{j=0}^{k_Q^x} \overline{PP_x}^{k_Q^x - i} \overline{PQ_x}^{k_Q^x - j} \binom{k_P^x}{i} \binom{k_Q^x}{j}\right] x_S^k \tag{3.2.27}$$

As in Equation 3.2.16, the portion in the square brackets can be replaced with the function f making the Gaussian at center S:

$$\begin{aligned} G_S = N_A N_B N_C N_D K_P K_Q K_S & \sum_{k_P^x=0}^{P_x} \sum_{k_P^y=0}^{P_y} \sum_{k_P^z=0}^{P_z} \sum_{k_Q^x=0}^{Q_x} \sum_{k_Q^y=0}^{Q_y} \sum_{k_Q^z=0}^{Q_z} f_{k_P^x}^P f_{k_P^y}^P f_{k_P^z}^P f_{k_Q^x}^Q f_{k_Q^y}^Q \\ & \sum_{k_S^x=0}^{P_x+Q_x} \sum_{k_S^y=0}^{P_y+Q_y} \sum_{k_S^z=0}^{P_z+Q_z} f_{k_S^x}^S f_{k_S^y}^S f_{k_S^z}^S x_S^{k_S^x} x_S^{k_S^y} x_S^{k_S^z} \mathrm{e}^{-\gamma^S r_S^2} \end{aligned} \tag{3.2.28}$$

The new Gaussian G_S can be integrated using the same method as the 2 center integrals to give four center overlap of A,B,C,D.

The code developed from this section can be found in Appendix A.

Chapter 4

Results

4.1 Fitting Radial Electron Densities of Atoms

The proof of concept is to show that molecular radial electron densities can be partitioned into atomic contributions. These contributions were then fitted and the fits were then combined into the molecular density. Before attempting the atoms in molecules fits it was decided to fit the individual atomic radial electron density for atoms of the first three periods of the periodic table (H-Ar). The individual densities were fit using a linear combination of nuclear centered s-type Gaussians given by:

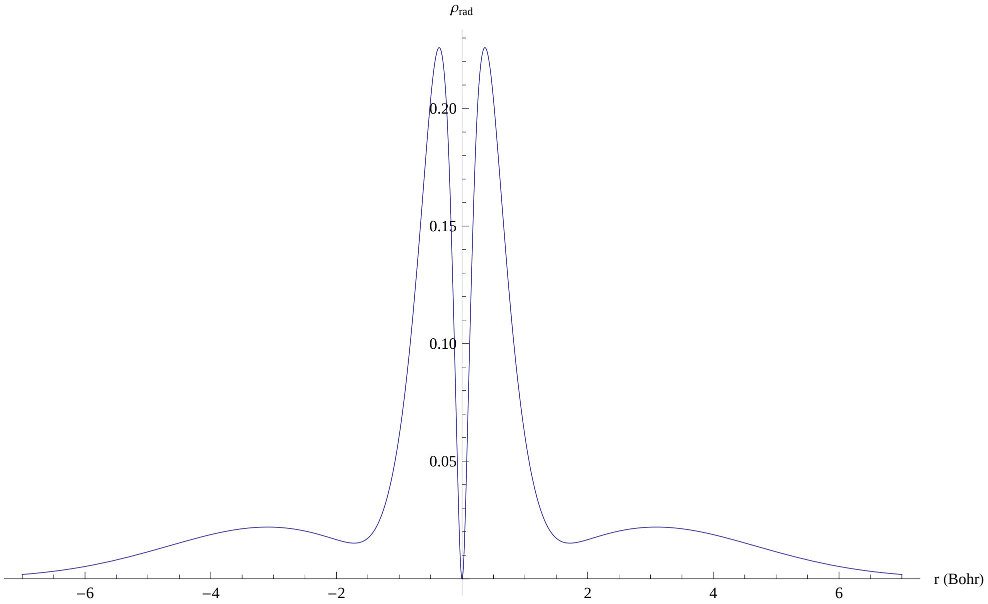
$$\rho^{rad} = \sum_{i=1}^{n} (c_i \mathrm{e}^{-\alpha_i r^2})^2 r^2 \tag{4.1.1}$$

In Equation 4.1.1 c_i and α_i are the coefficients and exponents of the Gaussian function found using Mathematica's built-in function *Findfit*. In all cases, any negative coefficients or exponents were thrown out and the case reran with constraints to ensure all positive coefficients and exponents. Although negative coefficients make sense mathematically they do not physically. The density is calculated from the coefficients, therefore negative coefficients implies negative density. The number of Gaussians used (n in Equation 4.1.1) for the first three periods was six. In the case of atoms, the electron density in all three dimensions is spherically symmetric about the nucleus so Equation 4.1.1 is general for one to three dimensions and r is:

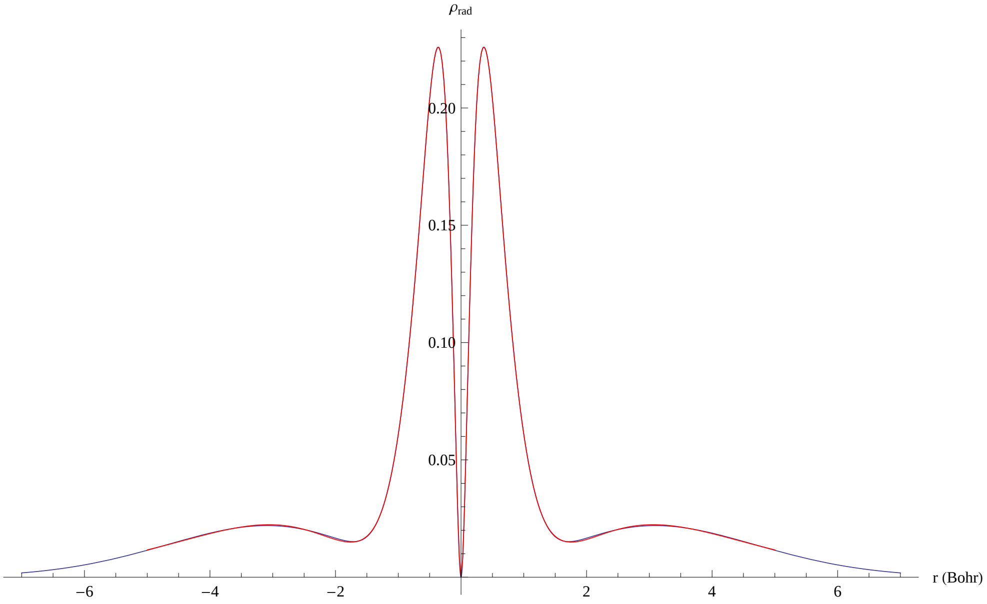
$$r = \sqrt{x^2 + y^2 + z^2} \tag{4.1.2}$$

Looking at the atom in the yz-plane the equation would be the same as 4.1.1 while replacing r^2 with $y^2 + z^2$.

Figures 4.1.1a and 4.1.1b show plots of the fit for the radial electron density of the lithium atom and a comparison of the fit with the actual radial density obtained via Hartree-Fock theory. As was established in Section 1.2, some shell structure is visible, a sharper peak close to the nucleus representing the core and more diffuse valence peak further out from the nucleus. Another feature of the individual atoms is that they are spherically symmetric about the nucleus, which in the case of the individual atom fits is at the origin. Visually the graphical comparison of the actual radial density to the fit is excellent, they appear to overlap perfectly, the fitted graph in 4.1.1b is only visible above and below 5 and -5 bohr respectively where the actual density was cut off, to show that the fitted graph was also present.



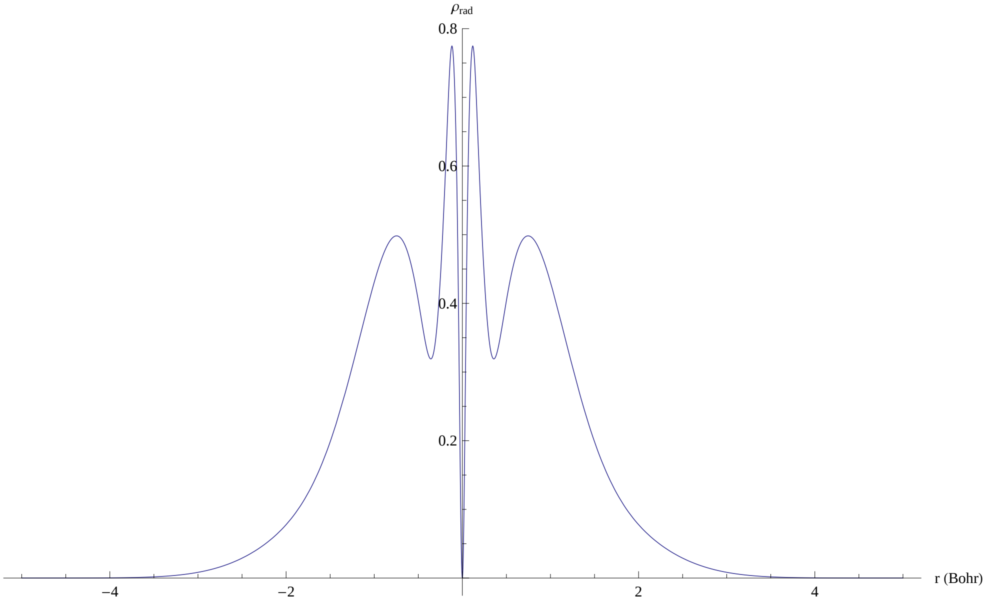
(a) Fitted radial electron density of a Li atom.



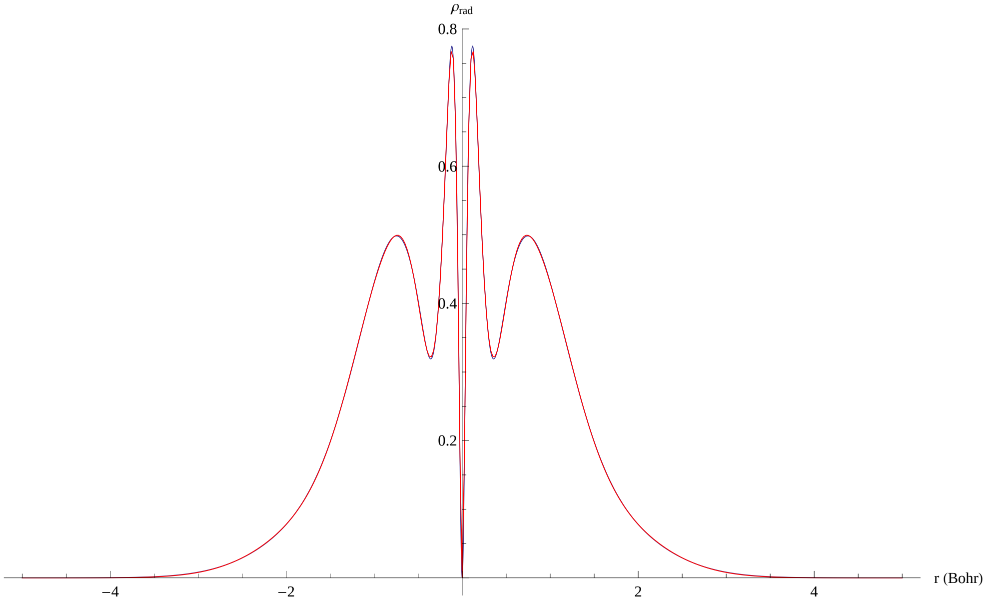
(b) Fitted (blue) vs actual (red) radial electron density of a Li atom.

Figure 4.1.1: Fit of the radial electron density of a Li atom (a) and a comparison of the fit and the actual radial electron density (b) vs r in bohr.

Figures 4.1.2a and 4.1.2b show plots of the fit for the radial electron density of the fluorine atom and a comparison of the fit with the actual radial density. As expected, the core shell of fluorine is much sharper than the core shell of the lithium atom. The valence shell is less diffuse and closer to the nucleus than the valence shell of the lithium atom. Looking at the comparison Figure 4.1.2b, visually it looks good but not perfect. Looking closely at the core shell it can be seen that the fit slightly exaggerates the maximum and minimum of the core shell.



(a) Fitted radial electron density of a F atom.



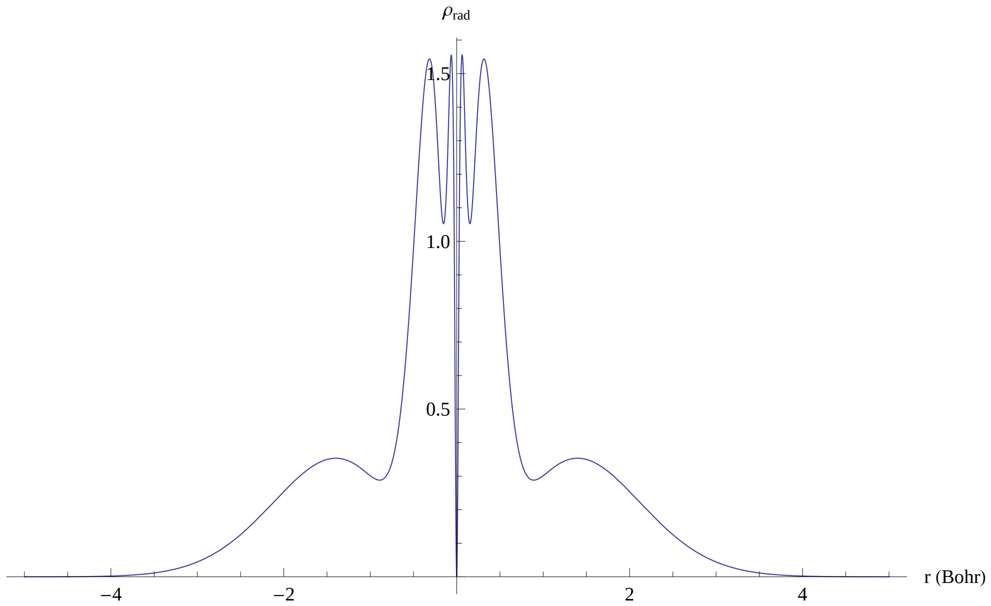
(b) Fitted (blue) vs actual (red) radial electron density of a F atom.

Figure 4.1.2: Fit of the radial electron density of a F atom (a) and a comparison of the fit and the actual radial electron density (b) vs r in bohr.

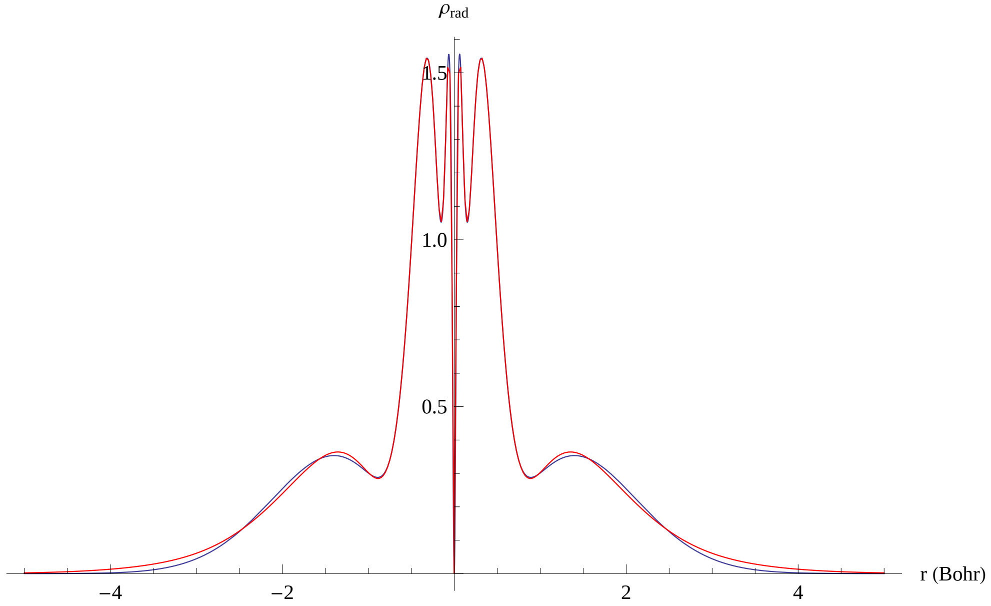
The fit of the Cl atom can be seen in Figure 4.1.3a. There are now two sharp core shells close to the nucleus as expected. As you move down the rows of the periodic table the number of cores increases, and it has a diffuse valence shell. Looking at the valence shell in Figure 4.1.3b we see that the fit does not do as well on the valence shell, as compared to the fits of the valence shells of atoms from the second period. This suggests that the 6 s-type Gaussians are not enough to fit both core and valence shells of a second row atom. From Table 4.1, it can be seen that as you move across the periodic table from lithium to neon, six s-type Gaussians are enough to fit the density, since the root mean square deviation(RMSD) in the electron density is no more than 0.0015. As you move down the period to sodium and towards argon the six Gaussians are not enough to fit the two cores and the valence shells. Here the RMSD becomes as large as 0.012. Since the core shells compose most of the electron density the valence shell is fitted poorly.

Table 4.1: Root mean squared deviation, the number of electrons (N_e) and the error in the expected number of electrons in the fits for the atomic radial electron densities.

Atom	RMSD	N_e	$\|\Delta N_e\|$	Atom	RMSD	N_e	$\|\Delta N_e\|$
H	0.00006	1.001	0.001	Ne	0.00146	10.005	0.005
He	0.000007	2.000	0.000	Na	0.00251	10.861	0.139
Li	0.00024	2.962	0.038	Mg	0.00185	11.893	0.107
Be	0.00138	3.960	0.040	Al	0.00305	12.820	0.180
B	0.00067	5.005	0.005	Si	0.00456	13.815	0.185
C	0.00049	5.999	0.001	P	0.00546	15.005	0.005
N	0.00089	6.991	0.009	S	0.00539	16.004	0.004
O	0.00118	8.001	0.001	Cl	0.01013	16.698	0.302
F	0.00131	9.003	0.003	Ar	0.01198	17.665	0.335



(a) Fitted radial electron density of a Cl atom.



(b) Fitted (blue) vs actual (red) radial electron density of a Cl atom.

Figure 4.1.3: Fit of the radial electron density of a Cl atom (a) and a comparison of the fit and the actual radial electron density (b) vs r in bohr.

The need for more than six Gaussian functions can also be seen in the number of electrons, which can be obtained by integrating the fit such that:

$$N_e = 4\pi \int_0^\infty \sum_{i=1}^{n} (c_i \mathrm{e}^{-\alpha_i r^2})^2 r^2 \mathrm{d}r \tag{4.1.3}$$

The number of electrons is in good agreement with the expected number of electrons for the first two periods but gets worse as you move down to the third period and across it. As can be seen in Table 4.1 the root mean square deviation in the number of electrons goes from 0.139 to 0.335 in going from sodium to argon. The root mean square deviation can be defined as:

$$d = \sqrt{\frac{\sum_r (\rho_{HF}(r) - \rho_{fit}(r))^2}{n}} \tag{4.1.4}$$

where $\rho_{HF}(r)$ is the actual radial density, $\rho_{fit}(r)$ is the radial density of the fit at some point r, and n is the number of points evaluated.

4.2 Fitting Radial Electron Densities of Atoms in Molecules

As is expected, for an atom in a molecule when the atoms interact their properties, such as density, are affected. In the case of electron densities, the valence electrons can be redistributed, lost or gained, while leaving the core electrons mostly unaffected [1, 19]. The redistribution of valence electron densities means that s-type Gaussian functions are no longer suitable to model the valence shells of the electron density.

S-type Gaussian functions are spherically symmetric about their center. While this is fine for the core electrons, the valence electron shells will not be spherically symmetric about the nucleus. The spherical symmetry can be broken by adding non-nuclear centered Gaussian functions which will have the form:

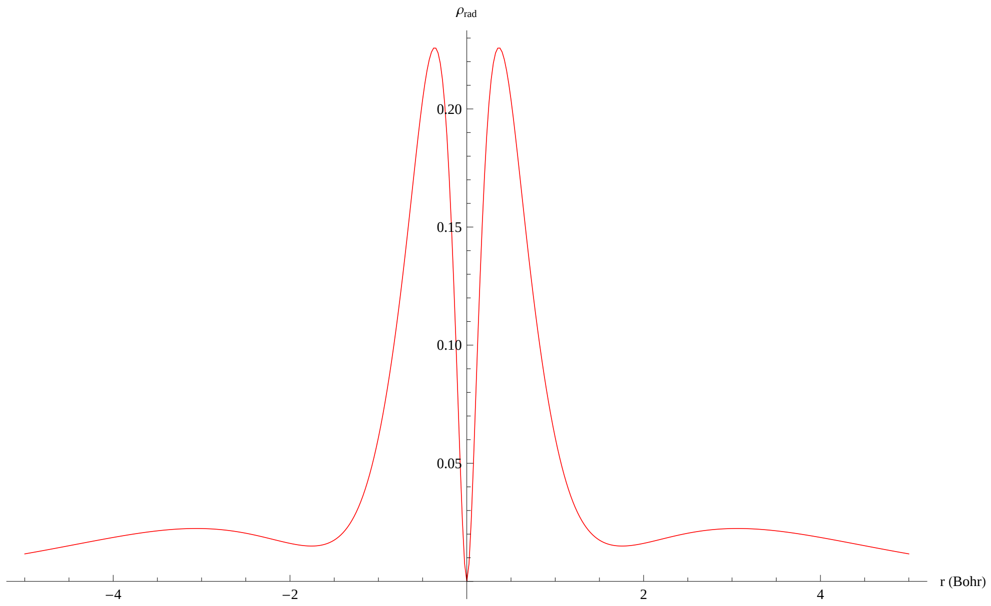
$$c\mathrm{e}^{-\alpha(r+r_s)^2} \tag{4.2.1}$$

where c is a constant and α is an exponent and r_s is some shift in the direction of r. In the case of one dimensional fits, r is along the bond axis, which in MUNgauss is always in the z direction for diatomics. The general model, Equation 4.1.1, as seen in Section 4.1, becomes:

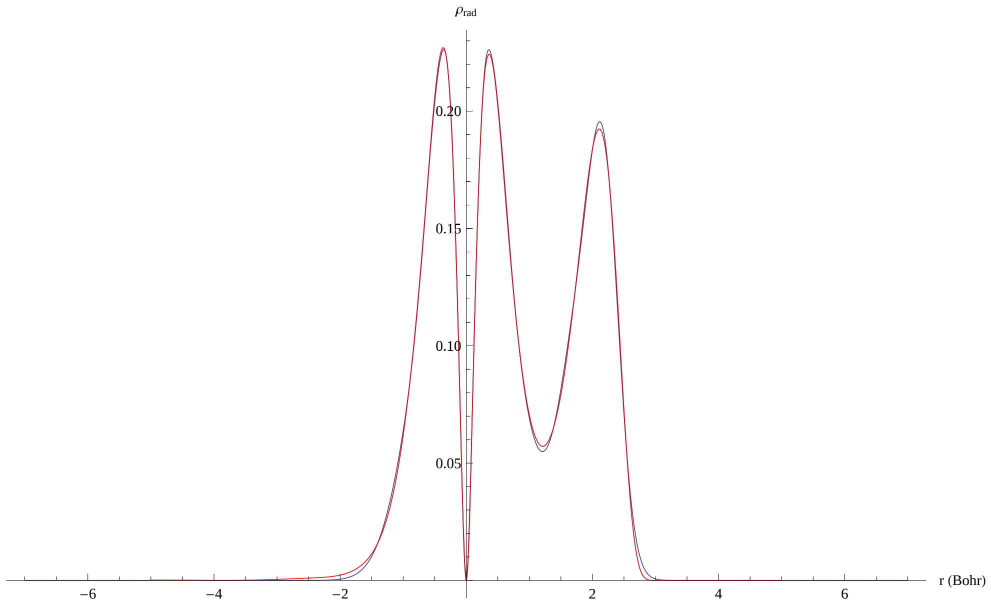
$$\rho^{rad} = \sum_{i=1}^{n}(c_i\mathrm{e}^{-\alpha_i r^2})^2 r^2 + \sum_{j=1}^{m}(c_j\mathrm{e}^{-\alpha_j(r+s_j)^2})^2 r^2 \tag{4.2.2}$$

where n and m are the number of nuclear centered and floating Gaussian functions (Gaussians that have been shifted to a new center) used in the fit respectively and s_j is the shift in the floating Gaussian. Unlike the individual atom fits, where n was set to six, for AIM the number of nuclear centered Gaussian functions is set to four or five depending on where the atom is on the periodic table of the elements. The number of floating Gaussian functions used was one to four. The nuclear centered Gaussian functions were used to fit the core shells of the radial electron density. The floating Gaussian functions were used to fit the valence shell radial electron density. The initial guess for the nuclear centered Gaussian functions was taken from the individual atom fits.

Comparing Figures 4.2.1a and 4.2.1b, it is easy to see the change in the valence shell of the radial electron density for lithium in lithium hydride. For atomic lithium (Figure 4.2.1a) the valence shell is spherically symmetric, while lithium in lithium hydride has no valence shell on the non-bonding side of the nucleus in the radial electron density. Also, where the valence shell of the radial election density in atomic lithium is diffuse with a low minimum, the valence shell in the bonding area of lithium in lithium hydride (right of the atom at the origin) is very sharp with a maximum close to the maximum of the core shell radial electron density. Figure 4.2.1b, the comparison of the fit to the actual radial electron density, it can be seen that though it may not appear to match the actual radial electron density as well as the individual lithium fit it does have the correct shape.



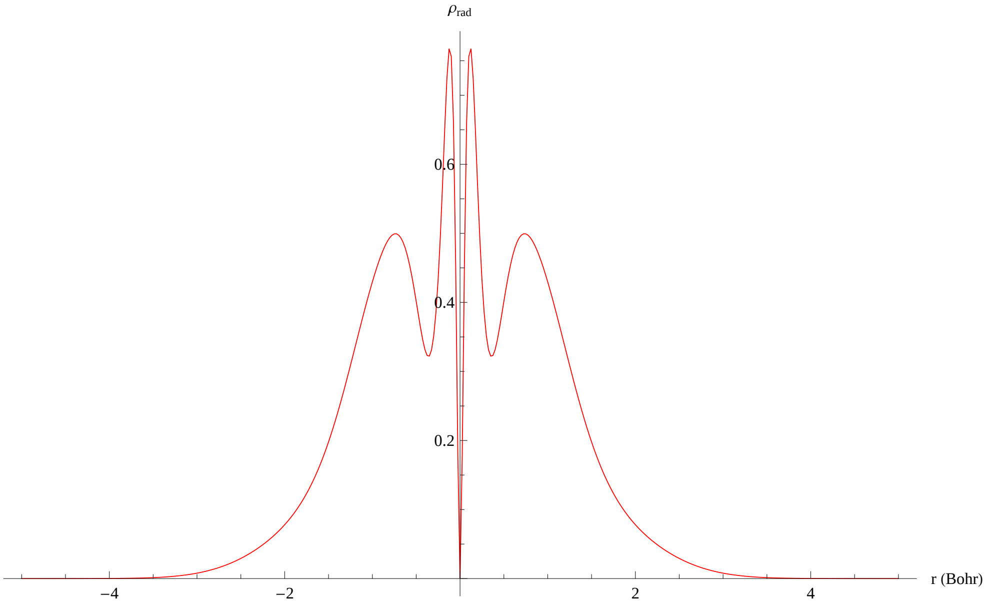
(a) Actual individual Li atom radial electron density.



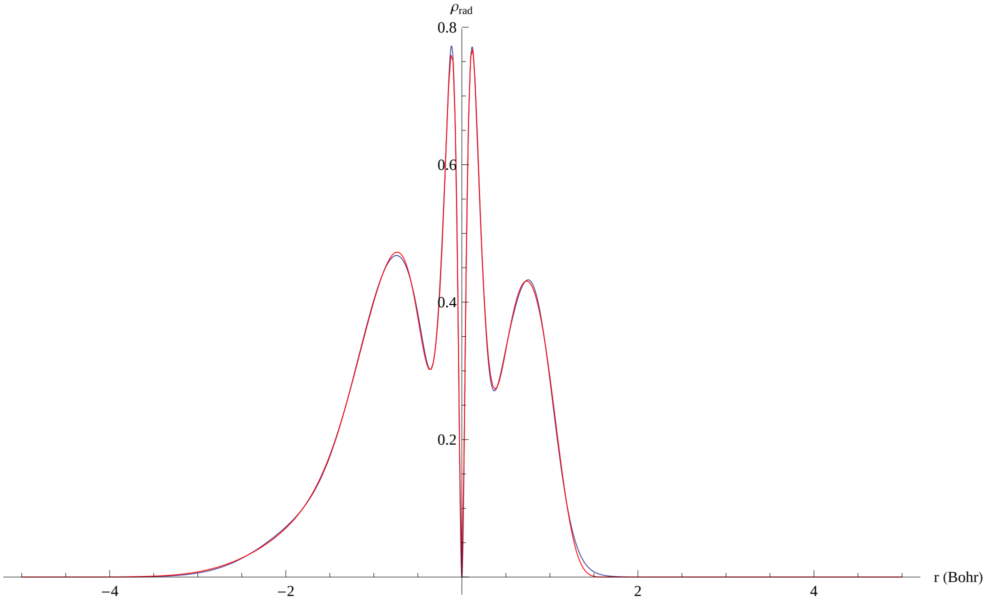
(b) Fitted (blue) vs actual (red) radial electron density of Li in LiH.

Figure 4.2.1: Radial electron density of the Li atom (a) and a comparison of the fit and the actual radial electron density of Li in LiH (b) vs r in bohr.

Figure 4.2.2b shows the fit of fluorine in hydrogen fluoride. Unlike lithium in Figure 4.2.1b, fluorine still has a valence shell of the radial electron density in the non-bonding region (left of the atom at the origin) which is a wider and taller peak compared to the valence shell of the radial electron density in the bonding region. Visually the comparison of the fit and actual radial electron density, Figure 4.2.2b, has a few flaws, but overall has the correct shape. Figures 4.2.2a and 4.2.2b of fluorine also shows, like the lithium Figures 4.2.1a and 4.2.1b, how the core shells of the radial electron density are unchanged, while the valence shell in the bonding and non-bonding region are no longer spherically symmetric.



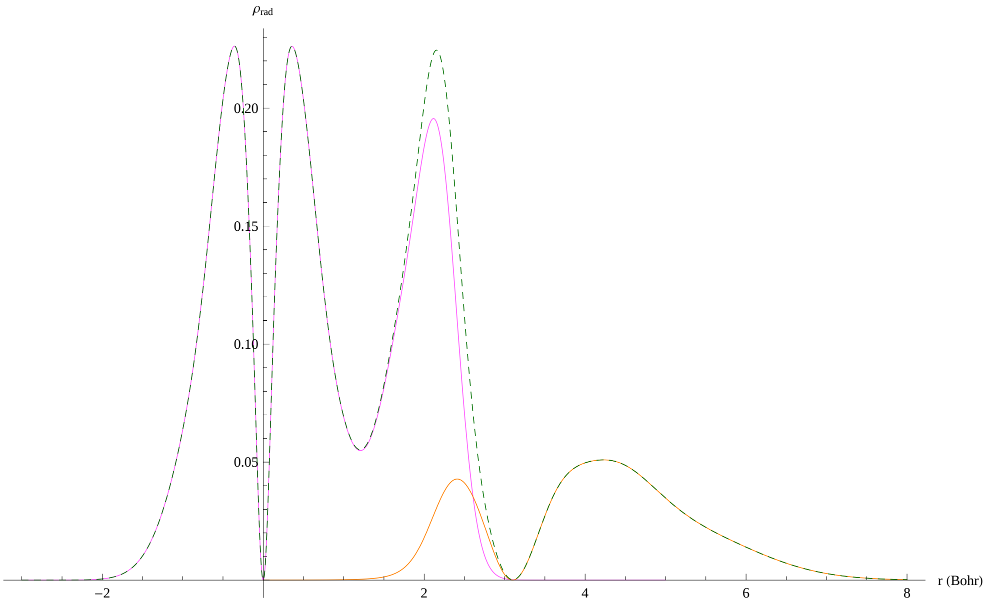
(a) Actual individual F atom radial electron density.



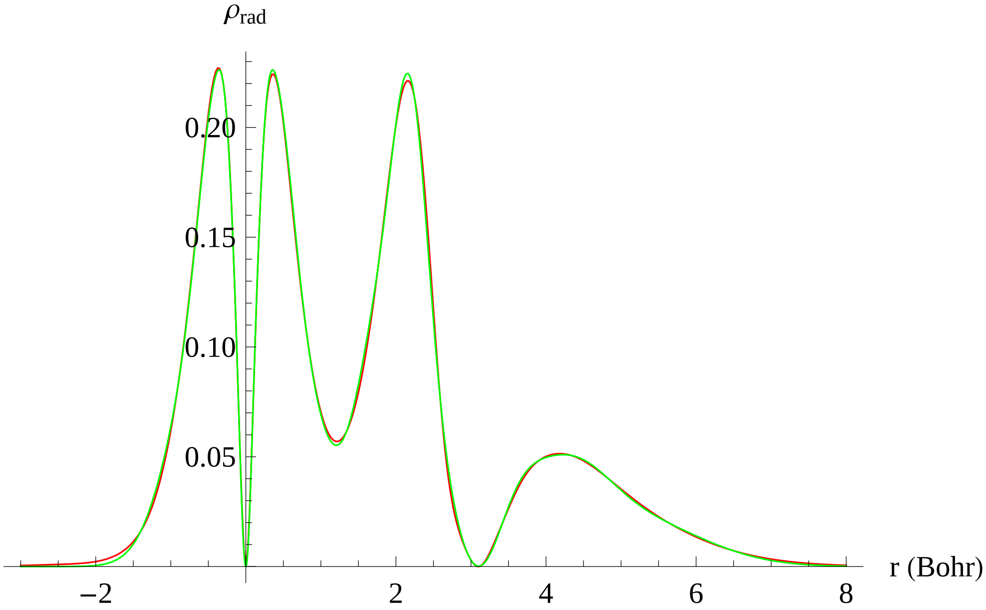
(b) Fitted (blue) vs Actual (red)radial electron density of F in HF.

Figure 4.2.2: Radial electron density of the F atom (a) and a comparison of the fit and the actual radial electron density of F in HF (b) vs r in bohr.

One of our goals was to show that once a molecule is partitioned into its atomic contributions, and those contributions are fitted, the fits could be used to reconstruct the molecular radial electron density. Figure 4.2.3a is an example of the reconstruction of the molecular radial electron density of lithium hydride (LiH). The fit of lithium, the pink graph, was added to the hydrogen fit, the orange graph, to give the molecular radial electron density fit of LiH, the green graph. When visually compared to the actual molecular radial electron density, as seen in Figure 4.2.3b, it has the correct shape, with few errors. The same can be said of Figure 4.2.4b where the fits of hydrogen and fluorine in hydrogen fluoride were used to construct the molecular radial density fit of hydrogen fluoride, which when compared visually to the actual molecular radial electron density appears to match almost perfectly.

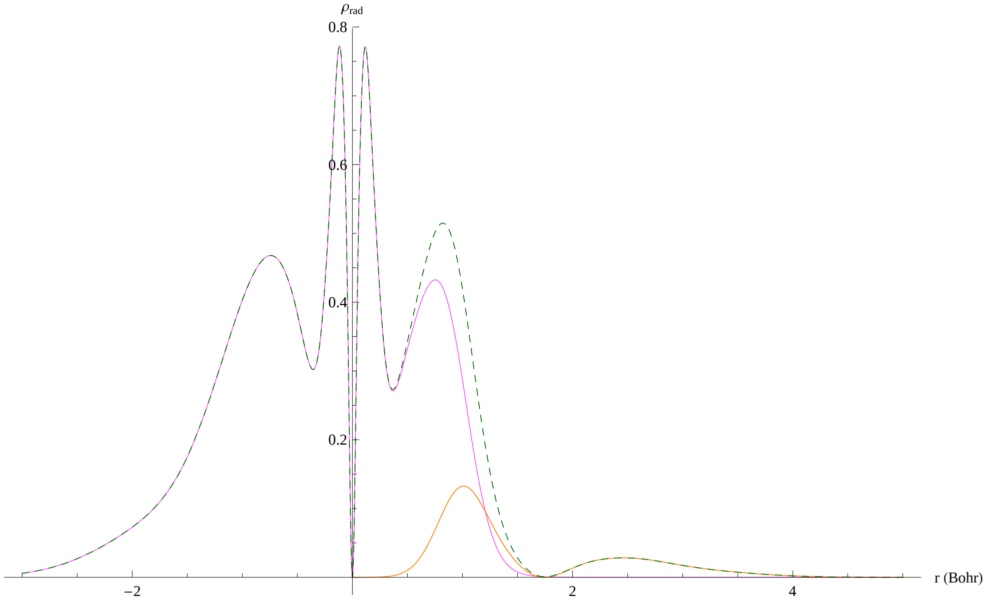


(a) Fitted radial electron densities of Li (pink) and H (orange) in LiH (green).

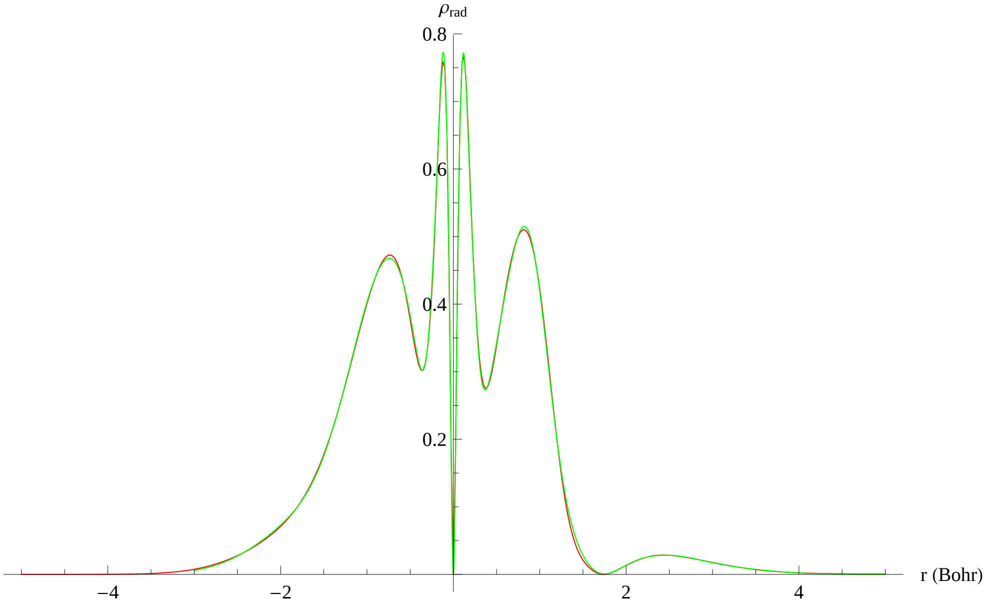


(b) Fitted(green) vs actual(red) molecular radial electron density of LiH.

Figure 4.2.3: A comparison of radial electron density of the fitted AIM and the total molecular radial electron density (a) and a comparison of the fitted and the actual molecular radial electron density (b) of LiH vs r in bohr.



(a) Fitted radial electron densities of F (pink) and H (orange) in HF (green).



(b) Fitted (green) vs actual (red) molecular radial electron density of HF.

Figure 4.2.4: A comparison of radial electron density of the fitted AIM and the total molecular radial electron density (a) and a comparison of the fitted and the actual molecular radial electron density (b) of HF vs r in bohr.

Unlike the root mean squared deviation in radial electron density for the individual atoms, whose error increases from left to right in a period, the deviation in the atom in molecules, as seen in Table 4.2, tends to vary more. For instance oxygen in diatomic oxygen and beryllium in beryllium hydride had a RMSD of 0.0058 and 0.0056 respectively. Which is larger than the RMSD of fluorine in difluoride and hydrogen fluoride which were 0.0034 and 0.0028 respectively. The fit of oxygen in carbon monoxide has a RMSD of 0.0021 which is again lower than both the fluorine fits, which suggests that the error depends on what the atoms are bonded to.

Table 4.2: Root mean squared deviation of the fits for atomic radial electron densities of AIM and root mean squared deviation of the fits for the total molecular radial electron density.

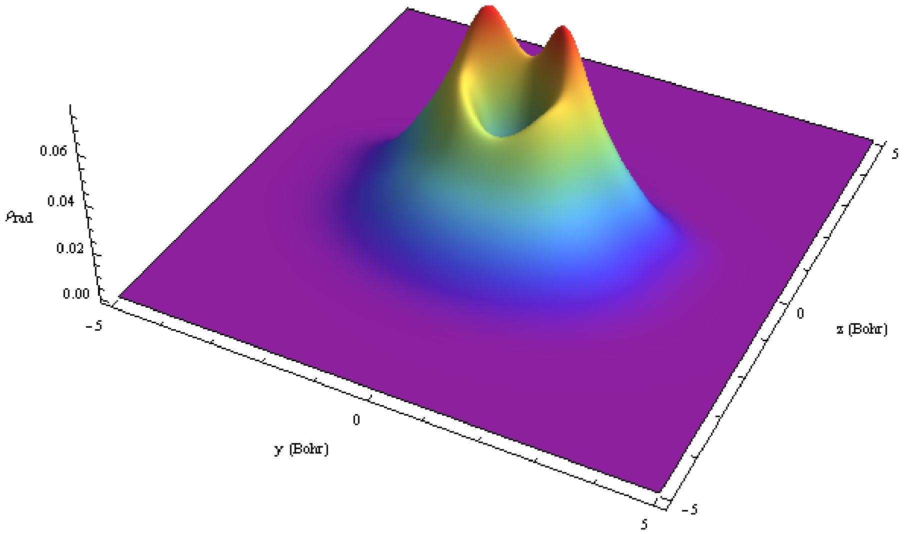
Molecule	Atom	RMSD	
		AIM	Molecular
H_2	H	0.00067	0.00094
Li_2	Li	0.00095	0.0013
O_2	O	0.00582	0.0082
F_2	F	0.00342	0.0048
LiH	Li	0.00150	0.0016
	H	0.00042	
HF	F	0.00284	0.0029
	H	0.00038	
CO	C	0.00429	0.0048
	O	0.00209	
BeH_2	Be	0.00563	0.0057
	H1	0.00044	
	H2	0.00044	

4.3 Fitting Molecular Orbital Radial Electron Densities of Atoms in Molecules

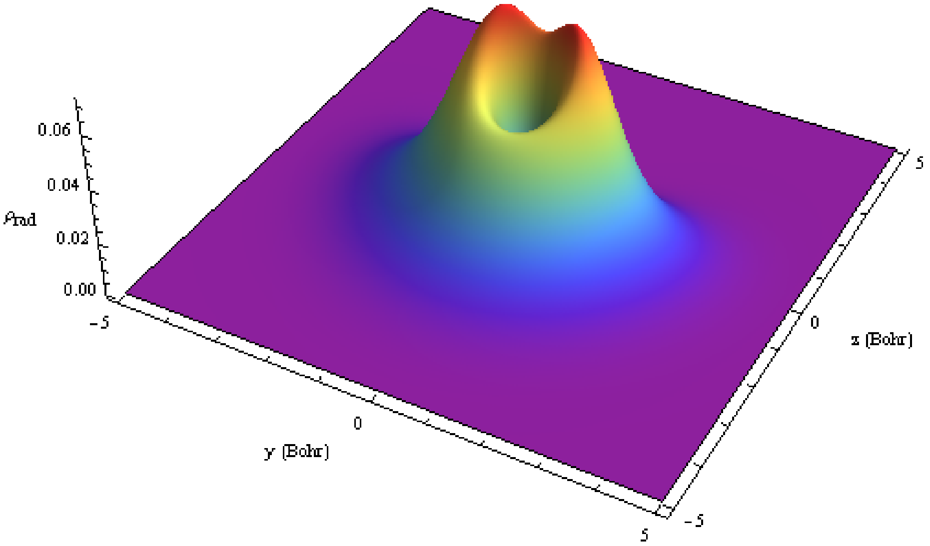
It has been shown that it is possible to fit atoms in molecules radial electron densities and then use those fitted radial electron densities to reconstruct the molecular radial electron density, in one dimension. Atoms exist in three dimensions so we need to expand from one dimension to three dimensions to see the whole picture. Instead of jumping from fitting atoms in one dimension to three dimensions, it can be beneficial to fit atoms in two dimensions first. The fewer electrons that an atom or atom in a molecule has, the less complex the radial electron density, hence the easier it is to fit. For instance, hydrogen in dihydrogen has no core electrons, in one dimension it was fitted using four floating Gaussian functions. So in two dimensions the fit should be four floating Gaussian functions dependent on both y and z. The equation used to fit the two dimensional hydrogen in dihydrogen is:

$$\rho^{rad} = \sum_{j=1}^{4} (c_j^{yz} \mathrm{e}^{-\alpha_j^y (y+y_j^s)^2 - \alpha_j^z (z+z_j^s)^2})^2 (z^2 + y^2) \tag{4.3.1}$$

In Equation 4.3.1 there are separate shifts and exponents for the y and z directions for every floating Gaussian used, this formulation seems to give the best results which can be seen in Figures 4.3.1a and 4.3.1b. In Figure 4.3.1a, the fit of hydrogen in dihydrogen, even though it has the correct general shape the minimums at z of +1 and -1 bohr are too low.



(a) Fitted radial electron density.



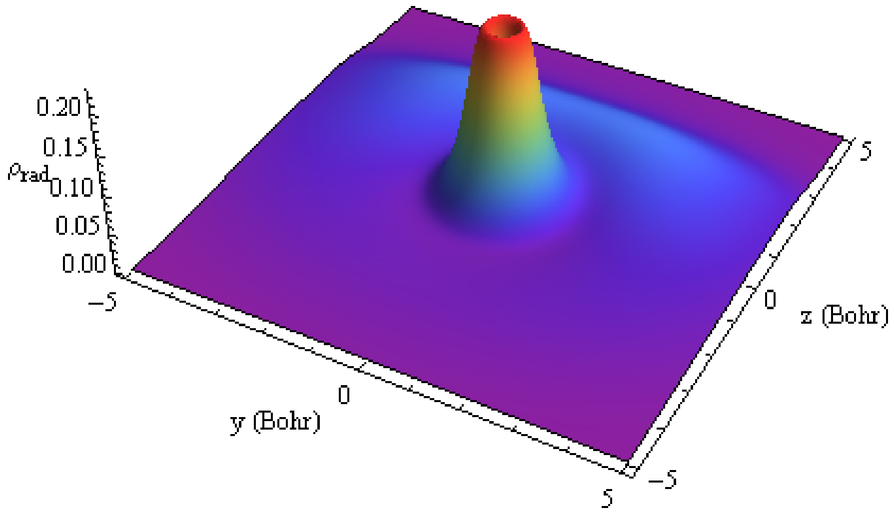
(b) Actual radial electron density.

Figure 4.3.1: The fitted (a) and actual (b) radial electron density of H in H_2 in the yz-plane.

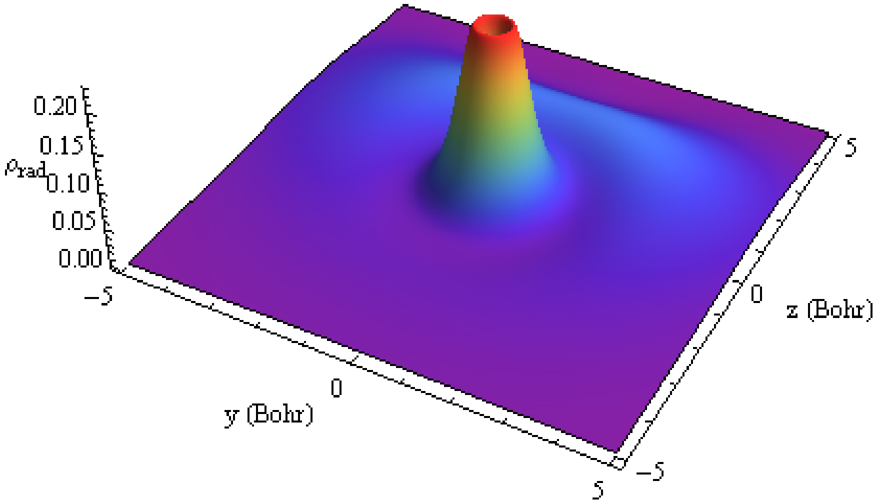
From this we can generalize Equation 4.3.1 to be:

$$\rho^{rad} = \sum_{i=1}^{n}(c_i^{yz}\mathrm{e}^{-\alpha_j^y(y)^2-\alpha_j^z(z)^2})^2(z^2+y^2)+\sum_{j=1}^{m}(c_j^{yz}\mathrm{e}^{-\alpha_j^y(y+s_j^y)^2-\alpha_j^z(z+s_j^z)^2})^2(z^2+y^2) \quad (4.3.2)$$

which is the general equation used for fitting lithium in dilithium. In one dimension lithium required four nuclear centered Gaussian functions for the core and two floating Gaussian functions for the valence shell, one in the non-bonding region (left of the nucleus) and one in the bonding region (right of the nucleus). A two dimensional fit was done using the same number of nuclear centered and floating Gaussian functions. The results can be seen in Figure 4.3.2a which when compared to the actual radial electron density, Figure 4.3.2b, that the valence shells have slightly different shapes. For instance, the valence shell of the fit seems to be rounder moving from -5 to +5 bohr on the y axis at about a z of +4 bohr. When trying to improve the fitted radial electron density of lithium the core tends to cause problems, for instance when trying to add another function, *Findfit* sets the floating Gaussians shift to zero to fit the core shell better or moves the floating Gaussian away from any shell (valence or core) and, forces the coefficient of the new Gaussian to be near zero, leaving the fit unimproved.



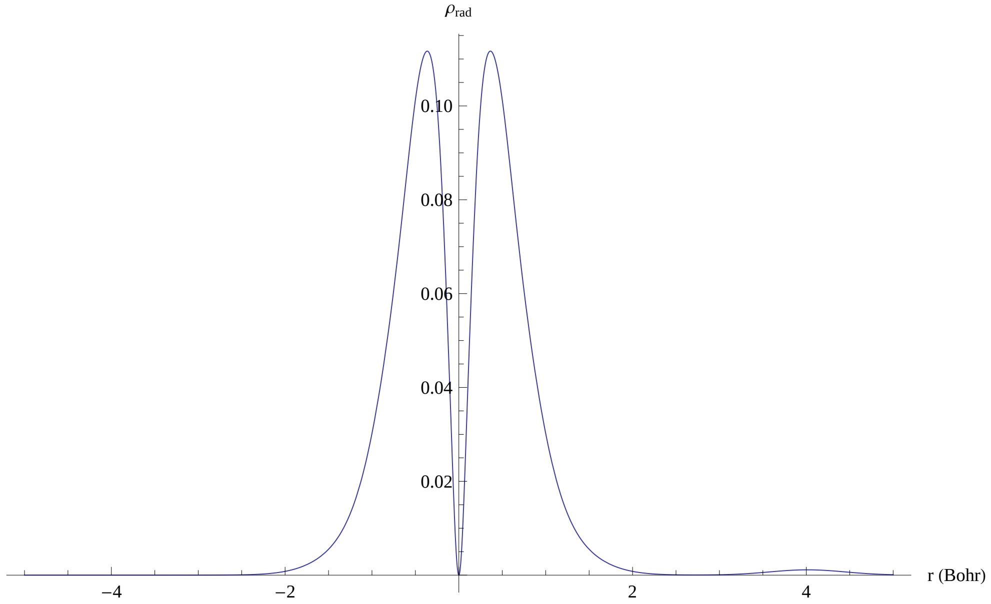
(a) Fitted radial electron density.



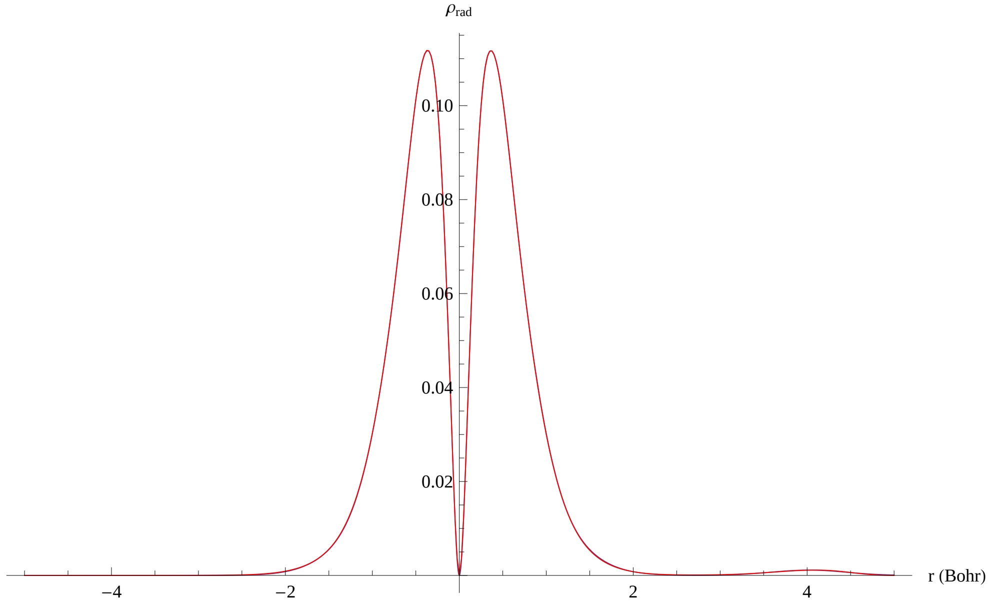
(b) Actual radial electron density.

Figure 4.3.2: The fitted (a) and actual (b) radial electron density of Li in Li_2 in the yz-plane.

The problem with trying to improve the fits of the two dimensional radial electron densities lead to trying to fit the radial densities of the molecular orbitals of each atom in the molecule. Using molecular orbitals instead of the whole atom density would allow us to fit the atom densities in parts. Core electrons occupy the lower energy molecular orbitals while the valence electrons occupy the higher energy molecular orbitals. Considering lithium in dilithium, the first two molecular orbitals as seen in Figures 4.3.3a and 4.3.3b are identical and are each one half of the core electron density of the total core electron density of lithium with a small valence shell at around 4.0 bohr. The fit with the lowest RMSD was obtained using five nuclear centered Gaussian functions and one floating Gaussian function.



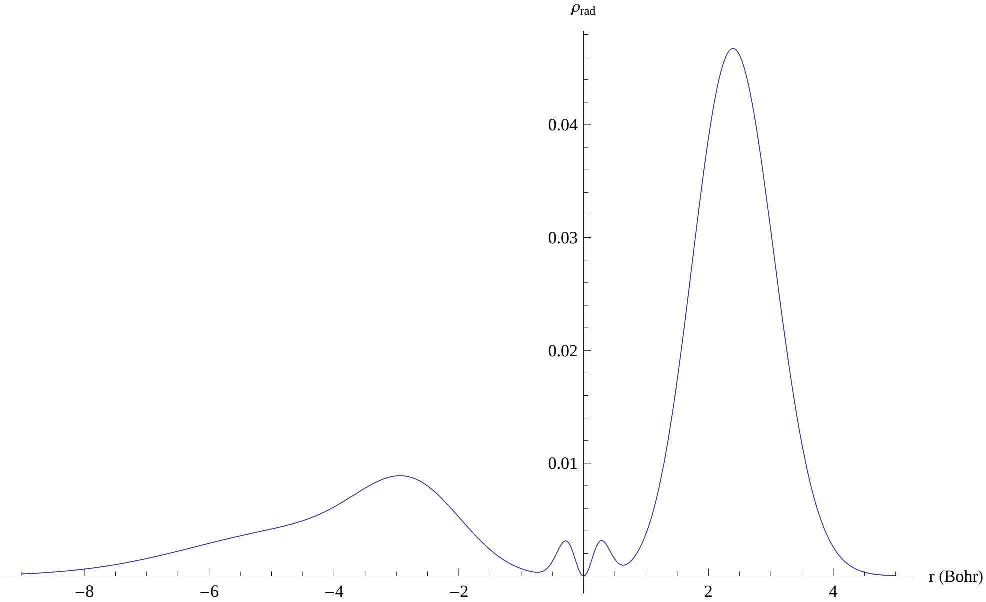
(a) Fitted radial electron density.



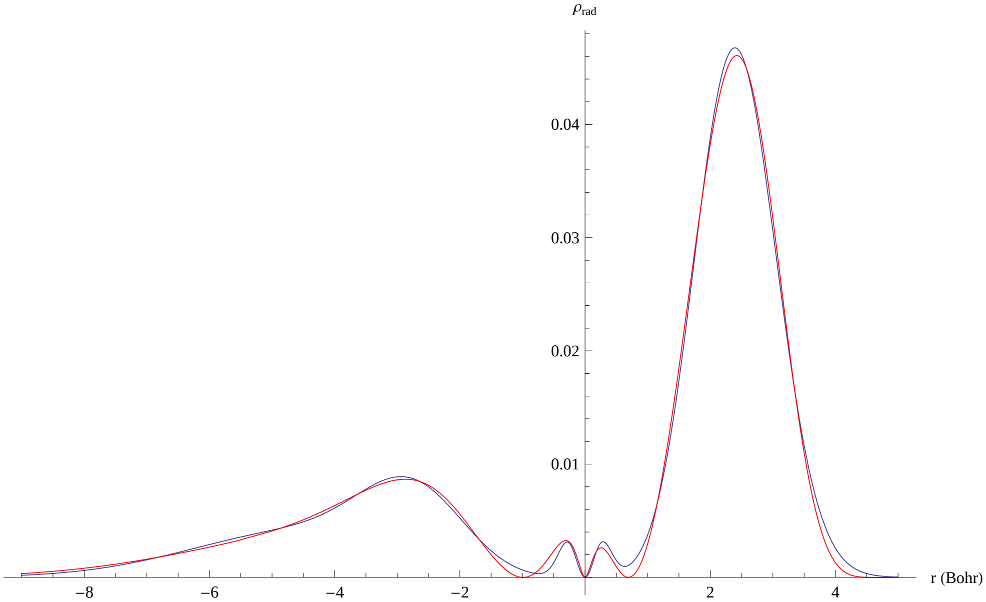
(b) Fitted (blue) vs actual (red) radial electron density.

Figure 4.3.3: Fitted radial electron density (a) and a comparison of the fitted and the actual radial electron density (b) of molecular orbitals one and two of Li in Li_2 vs r in bohr.

The third molecular orbital, as can be seen in Figures 4.3.4a and 4.3.4b, consists of the valence shells in the bonding and non-bonding region, and small core shells on either side of the nucleus. To fit this molecular orbital the individual shells were isolated by ignoring any density before and after their minimums. For example, to fit the large valence shell in the bonding region, only the density from 1.12 to 4.5 bohr was considered. The large valence shell in the bonding region was fitted using one floating Gaussian function, the core shells at the nucleus were fitted using one nuclear centered Gaussian function, and the diffuse valence shell was fitted using two floating Gaussian functions. The four functions were then added together and their parameters optimized to give the best result. In Figure 4.3.4b the fit has the correct general shape but is not perfect, for instance, going from the valence shell to the core shell in the non-bonding region, the radial density of the fit does not reach zero, unlike the actual radial density.



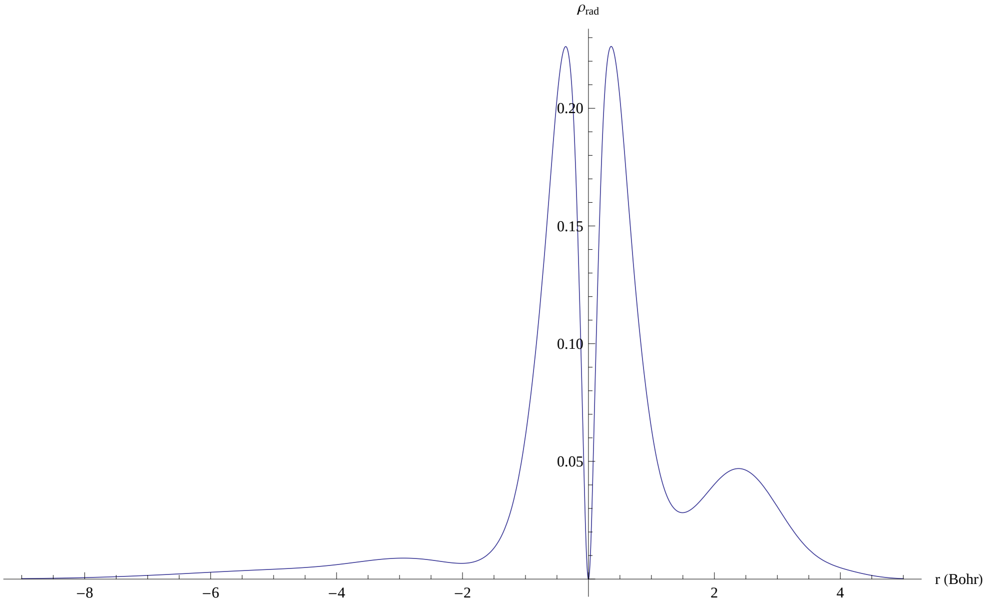
(a) Fitted radial electron density.



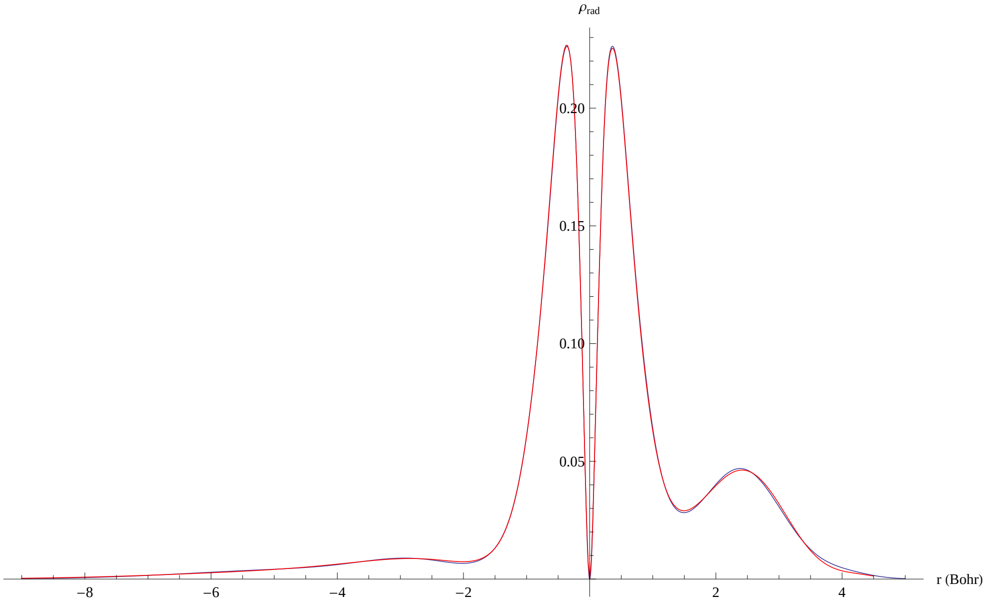
(b) Fitted (blue) vs actual (red) radial electron density.

Figure 4.3.4: Fitted radial electron density (a) and a comparison of the fitted and the actual radial electron density (b) of molecular orbital three of Li in Li_2 vs r in bohr.

Once the molecular orbitals were fitted they were then added together to give the total atom in molecule radial electron density to verify the process. This can be seen in Figures 4.3.5a and 4.3.5b.



(a) Fitted total radial electron density.



(b) Fitted (blue) vs actual (red) total radial electron density.

Figure 4.3.5: Total fitted radial electron density (a) and a comparison of the fitted and the actual total radial electron density (b) of Li in Li_2 vs r in bohr.

In Table 4.3, where the RMSD of the atom in molecule fit and the atom in molecule fit by molecular orbitals is compared. It can see that the method lowered the error in case of lithium and carbon in dilithium and carbon monoxide respectively, while the error was higher in the case of diatomic oxygen. Not only is in the deviation of the fit from the actual electron density show little overall improvement for the fits of atoms in molecules, it also more time consuming to fit each atom. There are multiple fits per atom, the number of fits is one half the number of electrons for that atom in a molecule, which for the diatomic oxygen is eight. Although it is good to note that in some cases not all the molecular orbitals will have radial electron densities for certain atoms, in the case of diatomic oxygen only molecular orbitals one, two, three, four and six were required to be fitted for the first atom.

Table 4.3: Root mean square deviation of the fits for atomic and molecular orbital radial electron densities of AIM.

Molecule	Atom	AIM RMSD	MO RMSD
Li_2	Li	0.00095	0.00047
LiH	Li	0.00015	0.00016
O_2	O	0.0058	0.0083
CO	C	0.0043	0.0031

Chapter 5

Conclusions

The code utilizing the least squares fitting algorithm could not be completed within the time frame of this thesis. The code containing the four center integral algorithm though ready, requires fine tuning and some minor updates to MUNgauss, such as updating overlap integrals to handle any shells s-h integrated with shells from s-g instead of only shells s-f. Shells are like the Gaussian types talked about in section 3.2, an s shell has no special dependence in any direction so the sum of its angular quantum numbers is zero, while a p can be spatially dependent in either x,y or z, the sum of the p shells angular quantum numbers would be one. The sum of quantum numbers for d, f, g and h would be 2, 3, 4 and 5 respectively. The least squares fitting code is still a viable fitting option once the four center integral code is ready.

It has been shown that it is possible to fit the first two periods of atoms of the periodic table (H-Ne) with just six s-type nuclear centered Gaussian functions. It is also possible to fit the third period (Na-Ar) with six s-type nuclear centered

Gaussian functions but not as well, and it can be seen that the error of the six Gaussian function fit increases moving from left to right within both the second and third periods. This is because a core shell requires about four to six s-type nuclear centered Gaussian functions to fit it well, while a valence shell requires around one to four s-type Gaussian functions. Meaning that to improve the atomic fits one needs to add more s-type nuclear centered Gaussian functions for atoms of the third period. The second period may show improvement with one additional Gaussian function to the heavier elements in that row. The individual atom fits may also be improved by fitting atoms by atomic orbital the way the atoms in molecules were fitted in Section 4.3. The cores shells could be fitted with six s-type nuclear centered Gaussian functions separately, without hindering the fitting of the valence shell. The core shells in this case are more important, as they are utilized as the initial guess for the cores shells of atoms in molecules. It has also been shown that the radial densities of atoms in molecules can be fitted using a combination of s-type nuclear and non-nuclear centered Gaussian functions in one dimension. It is also possible to assemble molecules from fitted electron densities in one dimension. Though some work was done in two dimensions, the fits still have to be improved before moving to fitting in three dimensions. The best way to do this is by fitting radial densities of molecular orbitals. Though it is very time consuming and tends to give results that vary from better to worse than the atoms in molecule fits, it would help manage the core shells that become more of a problem when fitting in two dimensions. It would also benefit from automation. As it would no longer take hours to fit the atoms and errors may also be reduced.

Bibliography

[1] Jessica Elizabeth Besaw. ATOMS AND BONDS IN MOLECULES : A Radial Density Analogue to the Quantum Theory of Atoms in Molecules, 2013.

[2] Timothy C. Lillestolen and Richard J. Wheatley. Redefining the atom: atomic charge densities produced by an iterative stockholder approach. *Chemical communications (Cambridge, England)*, 7345(45):5909–11, December 2008.

[3] A. D. Becke. A multicenter numerical integration scheme for polyatomic molecules. *The Journal of Chemical Physics*, 88(4):2547–54, 1988.

[4] R. F. W. Bader. Atoms in molecules. *Accounts of Chemical Research*, 18(1):9–15, January 1985.

[5] Eloy Ramos-Cordoba, Pedro Salvador, and István Mayer. The atomic orbitals of the topological atom. *The Journal of chemical physics*, 138(21):214107–16, 2013.

[6] R. A. Poirier and J. W. Hollett. *MUNgauss, Memorial University, Chemistry Department, St. John's, NL A1B 3X7. With contributions from A. Alrawashdeh, J.-P. Becker, J. Besaw, S.D. Bungay, F. Colonna, A. El-Sherbiny, T. Gosse, D.*

Keefe, A. Kelly, D. Nippard, C.C. Pye, D. Reid, K. Saputantri, M. Shaw, M. Staveley, O. Stueker, Y. Wang, P. L. Warburton and J. Xidos, 2013.

[7] Patrick Bultinck, David L. Cooper, and Dimitri Van Neck. Comparison of the Hirshfeld-I and iterated stockholder atoms in molecules schemes. *Physical chemistry chemical physics : PCCP*, 11(18):3424–9, May 2009.

[8] Pere Constans. Atomic Shell Approximation: Electron Density Fitting Algorithm Restricting Coefficients to Positive Values. *Journal of Chemical Information and Computer Science*, 35(9):1046–1053, 1995.

[9] A.T.B. Gilbert, A.M. Lee, and P.M.W. Gill. Methods for constructing Stewart atoms. *Journal of Molecular Structure: THEOCHEM*, 500(1-3):363–374, April 2000.

[10] Andrew T. B. Gilbert, Peter M. W. Gill, and Stephen W. Taylor. Extracting atoms from molecular electron densities via integral equations. *The Journal of chemical physics*, 120(17):7887–93, May 2004.

[11] Peter M. W. Gill. Extraction of Stewart Atoms from Electron Densities. *Journal of Physical Chemistry*, 3654(96):15421–15427, 1996.

[12] Tibor Koritsanszky and Anatoliy Volkov. Atomic density radial functions from molecular densities. *Chemical Physics Letters*, 385(5-6):431–434, February 2004.

[13] I. Mayer. Orthogonal effective atomic orbitals in the topological theory of atoms. *Journal of Chemistry*, 74(2):939–942, 2011.

[14] I. Mayer. Relation between the Hilbert space and fuzzy atoms analyses. *Chemical Physics Letters*, 585:198–200, 2013.

[15] R. S. Mulliken. Electronic Population Analysis on LCAO[Single Bond]MO Molecular Wave Functions. I. *The Journal of Chemical Physics*, 23(10):1833, 1955.

[16] R. S. Mulliken. Criteria for the Construction of Good Self-Consistent-Field Molecular Orbital Wave Functions, and the Significance of LCAO-MO Population Analysis. *The Journal of Chemical Physics*, 36(12):3428, 1962.

[17] Soumen Saha, R. A. M. Kinkar Roy, and Paul W. Ayers. Population Analysis Schemes Consistent With Chemical Intuition. *Wiley InterScience*, 109(2009):1790–1806, 2008.

[18] Anatoliy Volkov, Tibor Koritsanszky, Michal Chodkiewicz, and Harry F King. On the basis-set dependence of local and integrated electron density properties: Application of a new computer program for quantum-chemical density analysis. *Journal of computational chemistry*, 30(9):1379–91, July 2009.

[19] Peter L. Warburton, Raymond A. Poirier, and Devin Nippard. Atoms and bonds in molecules from radial densities. *The journal of physical chemistry. A*, 115(5):852–67, February 2011.

[20] Raymond Daudel, Georges Leroy, Daniel Peter, and Muchel Sana, editors. *Quantum Chemistry.* Wiley-interscience Publication, Hoboken, NJ, USA, 1983.

Appendices

Appendix A

Code

A.1 Four Center Integrals

```
      FUNCTION TRACEMS(A,B)
      USE program_constants
      USE type_molecule
      USE type_basis_set
      USE QM_objects
      implicit none
      double precision :: TRACEMS
      double precision,dimension(Basis%Nbasis,Basis%Nbasis) :: C,A,B
      integer :: Ibasis

      C=MATMUL(A,B)

      TRACEMS=ZERO
      do Ibasis=1,Basis%Nbasis
      TRACEMS = TRACEMS + C(Ibasis,Ibasis)
      end do
      return
      END FUNCTION TRACEMS
      subroutine GG_prod_int
!***********************************************************************
!     Date last modified: MAY 08, 2012                      Version 1.0 *
!     Author: R.A. Poirier & Devin G. S. Nippard                        *
```

```
!     Description:Integrate the product of 4 Gaussians               *
!*********************************************************************
! Modules:
      USE program_constants
      USE type_molecule
      USE type_basis_set
      USE QM_objects
      USE matrix_print

      implicit none
!
! Local scalars:
      integer :: Ishell,Jshell,Ifrst,Jfrst,Ilast,Jlast
      integer :: Lshell,Kshell,Lfrst,Kfrst,Llast,Klast
      integer :: Istart,Jstart,Pstart,Iend,Jend,Pend
      integer :: Lstart,Kstart,Qstart,Lend,Kend,Qend
      integer :: Iatmshl,Jatmshl
      integer :: Latmshl,Katmshl
      integer :: LAMAX,LBMAX
      integer :: LCMAX,LDMAX
      integer :: Irange,Jrange,Lrange,Krange
      integer :: Igauss,Jgauss,Lgauss,Kgauss
      integer :: Ibasis,Jbasis,Lbasis,Kbasis
```

```
      integer :: Iaos,JaoS
      integer :: Laos,KaoS
      integer :: IGBGN,JGBGN,IGend,JGend,JF
      integer :: LGBGN,KGBGN,LGend,KGend,KF,LF
      integer :: LPMAX,LIM1DSP,LIM1DSQ,LIM1DSPQ
      integer :: I,IJX,IJY,IJZ,J,IX,IY,IZ,JX,JY,JZ
      integer :: K,KLX,KLY,KLZ,L,KX,KY,KZ,LX,LY,LZ
      integer :: IJKLX,IJKLY,IJKLZ
      integer :: INTC,INTC2,ICOUNT,TOTint,INTCa,INTCb,INTCc
      integer :: LENTQP,LENTQQ,LENTQPQ,LPQMAX,LQMAX
      integer :: MATlen,MAXlen
      double precision :: ABX,ABY,ABZ,ARABSQ,CRCDSQ,KAB,KCD,KPQ, &
                          AS,BS,CS,DS,CUT1,COEFPQ,COEFQ,COEFP, &
                          EAB,EIAB,EXPAB,ECD,EICD,EXPCD,EPQ, &
                          EIPQ,EIO2PQ,EXPPQ,PX,PY,PZ,QX,QY,QZ, &
                          PQX,PQY,PQZ,RX,RY,RZ,STERMPQ,ZPR,ZQR, &
                          YPR,YQR,XPR,XQR,CDX,CDY,CDZ
      double precision :: XA,XB,YA,YB,ZA,ZB,XC,XD,YC,YD,ZC,ZD, &
                          ABSQ,PQSQ,CDSQ,XAP,XBP,YBP,YAP,ZAP, &
                          ZBP,COEFAB,STERMAB,EIO2AB,XCQ,XDQ,  &
                          YCQ,YDQ,ZCQ,ZDQ,COEFCD,STERMCD,EIO2CD,COEFR
!
! Local arrays:
```

```
      integer :: INDIX(32),INDIY(32),INDIZ(32),INDJX(32),INDJY(32), &
                 INDJZ(32),INDeIX(32),INDeIY(32),INDeIZ(32), &
                 INDeJX(32),INDeJY(32),INDeJZ(32),INDeLX(32), &
                 INDeLY(32),INDeLZ(32),INDeKX(32),INDeKY(32),INDeKZ(32)
      double precision :: CA(20),CB(20),CC(20),CD(20),CP(40),CQ(40), &
                          SSPQ(1300),SXPQ(32),SYPQ(32),SZPQ(32), &
                          S1CPQ(6),CCXPQ(192),CCYPQ(192),CCZPQ(192), &
                          SSAB(100),SXAB(32),SYAB(32),SZAB(32), &
                          S1CAB(6),CCXAB(192),CCYAB(192),CCZAB(192), &
                          SSCD(100),SXCD(32),SYCD(32),SZCD(32),S1CCD(6),CCXCD(192)
      double precision, dimension(:), allocatable :: OVRLAPP,OVRLAPQ
      double precision, dimension(:,:), allocatable :: Nij
!
      parameter (CUT1=-75.0D0)
!      s|  p  |    d      |         f         |           g           |
 DATA
 INDJX/1,2,1,1,3,1,1,2,2,1,4,1,1,2,3,3,2,1,1,2,5,1,1,2,4,4,2,1,1,3,3,1/,&
 INDJY/1,1,2,1,1,3,1,2,1,2,1,4,1,3,2,1,1,2,3,2,1,5,1,4,2,1,1,2,4,3,1,3/,&
 INDJZ/1,1,1,2,1,1,3,1,2,2,1,1,4,1,1,2,3,3,2,2,1,1,5,1,1,2,4,4,2,1,3,3/,&
 INDeIX/0,4,0,0,8,0,0,4,4,0,12,0,0,4,8,8,4,0,0,4,16,0,0,4,12,12,4,0,0,8,8,0/,&
 INDeIY/0,0,4,0,0,8,0,4,0,4,0,12,0,8,4,0,0,4,8,4,0,16,0,12,4,0,0,4,12,8,0,8/,&
 INDeIZ/0,0,0,4,0,0,8,0,4,4,0,0,12,0,0,4,8,8,4,4,0,0,16,0,0,4,12,12,4,0,8,8/,&
 INDeJX/0,4,0,0,8,0,0,4,4,0,12,0,0,4,8,8,4,0,0,4,16,0,0,4,12,12,4,0,0,8,8,0/,&
```

```
 INDeJY/0,0,4,0,0,8,0,4,0,4,0,12,0,8,4,0,0,4,8,4,0,16,0,12,4,0,0,4,12,8,0,8/,&
 INDeJZ/0,0,0,4,0,0,8,0,4,4,0,0,12,0,0,4,8,8,4,4,0,0,16,0,0,4,12,12,4,0,8,8/,&
 INDeKX/0,1,0,0,2,0,0,1,1,0,3,0,3,1,2,2,1,0,0,1,4,0,0,1,3,3,1,0,0,2,2,0/,&
 INDeKY/0,0,1,0,0,2,0,1,0,1,0,3,0,2,1,0,0,1,2,1,0,4,0,3,1,0,0,1,3,2,0,2/,&
 INDeKZ/0,0,0,1,0,0,2,0,1,1,0,0,3,0,0,1,2,2,1,1,0,0,4,1,0,1,3,3,2,0,2,2/,&
 INDeLX/1,2,1,1,3,1,1,2,2,1,4,1,1,2,3,3,2,1,1,2,5,1,1,2,4,4,2,1,1,3,3,1/,&
 INDeLY/1,1,2,1,1,3,1,2,1,2,1,4,1,3,2,1,1,2,3,2,1,5,1,4,2,1,1,2,4,3,1,3/,&
 INDeLZ/1,1,1,2,1,1,3,1,2,2,1,1,4,1,1,2,3,3,2,2,1,1,5,1,1,2,4,4,2,1,3,3/
!
      type Gauss_prod_int
        integer :: Ibasis
        integer :: Jbasis
        integer :: Kbasis
        integer :: Lbasis
        double precision :: value
      end type
      type(Gauss_prod_int), dimension(:), allocatable :: GGPQ
! Begin:
      call PRG_manager ('enter', 'GG_PROD_INT', '1EINT%GG')
!
      call GET_object ('QM','CMO','RHF')
      write(6,*)'Byond get object '
      call flush(6)
```

```
      MATlen=Basis%Nbasis*(Basis%Nbasis+1)/2
      MAXlen=MATlen*MATlen
! Object:
      if(.not.allocated(GGPQ))then
        allocate (GGPQ(MAXlen),OVRLAPP(MATlen),OVRLAPQ(MATlen))
      else
        if(Basis%Nbasis.ne.size(GGPQ,1))then
          deallocate (GGPQ,OVRLAPP,OVRLAPQ)
          allocate (GGPQ(MAXlen),OVRLAPP(MATlen),OVRLAPQ(MATlen))
        end if
      end if
      if(.not.allocated(Nij))then
        allocate (Nij(Basis%Nbasis,Basis%Nbasis))
      else
        if(Basis%Nbasis.ne.size(Nij,1))then
          deallocate (Nij)
          allocate (Nij(Basis%Nbasis,Basis%Nbasis))
        end if
      end if
      Nij(1:Basis%Nbasis,1:basis%Nbasis)=0.0D0
!
      do I=1,32
      INDIX(I)= 4*(INDJX(I)-1)
```

```
      INDIY(I)= 4*(INDJY(I)-1)
      INDIZ(I)= 4*(INDJZ(I)-1)
      end do

      ICOUNT=0
!
! Loop over shells.
! Loop over Ishell.
      do Ishell=1,Basis%Nshells
      LAMAX=Basis%shell(Ishell)%Xtype+1
      Istart=Basis%shell(Ishell)%Xstart
      Iend=Basis%shell(Ishell)%XEND
      Irange=Iend-Istart+1
      IGBGN=Basis%shell(Ishell)%EXPBGN
      IGEND=Basis%shell(Ishell)%EXPEND
      Ifrst=Basis%shell(Ishell)%frstSHL
!
! Loop over Jshell.
      do Jshell=1,Ishell
      LBMAX=Basis%shell(Jshell)%Xtype+1
      Jstart=Basis%shell(Jshell)%Xstart
      Jend=Basis%shell(Jshell)%XEND
      Jrange=Jend-Jstart+1
```

```
      JGBGN=Basis%shell(Jshell)%EXPBGN
      JGEND=Basis%shell(Jshell)%EXPEND
      Jfrst=Basis%shell(Jshell)%frstSHL
!
      Ilast= Basis%shell(Ishell)%lastSHL

!
! Loop over Kshell.
      do Kshell=1,Jshell
      LCMAX=Basis%shell(Kshell)%Xtype+1
      Kstart=Basis%shell(Kshell)%Xstart
      Kend=Basis%shell(Kshell)%XEND
      Krange=Kend-Kstart+1
      KGBGN=Basis%shell(Kshell)%EXPBGN
      KGEND=Basis%shell(Kshell)%EXPEND
      Kfrst=Basis%shell(Kshell)%frstSHL
!
!  Loop over Lshell
!
      do Lshell=1,Kshell
      LDMAX=Basis%shell(Lshell)%Xtype+1
      Lstart=Basis%shell(Lshell)%Xstart
      Lend=Basis%shell(Lshell)%XEND
```

```
      Lrange=Lend-Lstart+1

      LGBGN=Basis%shell(Lshell)%EXPBGN

      LGEND=Basis%shell(Lshell)%EXPEND

      Lfrst=Basis%shell(Lshell)%frstSHL


! Loop over Iatmshl

      do Iatmshl=Ifrst,Ilast

      XA=CARTESIAN(Basis%atmshl(Iatmshl)%ATMLST)%X

      YA=CARTESIAN(Basis%atmshl(Iatmshl)%ATMLST)%Y

      ZA=CARTESIAN(Basis%atmshl(Iatmshl)%ATMLST)%Z

      Iaos=Basis%atmshl(Iatmshl)%frstAO-1

!

! Loop over Jatmshl

      Jlast= Basis%shell(Jshell)%lastSHL

      IF(Ishell.EQ.Jshell)Jlast=Iatmshl

      do Jatmshl=Jfrst,Jlast

      XB=CARTESIAN(Basis%atmshl(Jatmshl)%ATMLST)%X

      YB=CARTESIAN(Basis%atmshl(Jatmshl)%ATMLST)%Y

      ZB=CARTESIAN(Basis%atmshl(Jatmshl)%ATMLST)%Z

      Jaos=Basis%atmshl(Jatmshl)%frstAO-1

!

      LPMAX=LAMAX+LBMAX-1

      LIM1DSP=(LPMAX+3)/2
```

```
      LENTQP=Irange*Jrange

      ABX=XB-XA

      ABY=YB-YA

      ABZ=ZB-ZA

      ABSQ=ABX*ABX+ABY*ABY+ABZ*ABZ


!

! Loop over Katmshl

      Klast= Basis%shell(Kshell)%lastSHL

!      IF(Ishell.EQ.Kshell)Klast=Iatmshl

!      IF(Jshell.EQ.Kshell)Klast=Jatmshl

      do Katmshl=Kfrst,Klast

      XC=CARTESIAN(Basis%atmshl(Katmshl)%ATMLST)%X

      YC=CARTESIAN(Basis%atmshl(Katmshl)%ATMLST)%Y

      ZC=CARTESIAN(Basis%atmshl(Katmshl)%ATMLST)%Z

      Kaos=Basis%atmshl(Katmshl)%frstAO-1


!

! Loop over Latmshl

      Llast= Basis%shell(Lshell)%lastSHL

!      IF(Ishell.EQ.Lshell)Llast=Iatmshl

!      IF(Jshell.EQ.Lshell)Llast=Jatmshl

      IF(Kshell.EQ.Lshell)Llast=Katmshl
```

```
      do Latmshl=Lfrst,Llast
      XD=CARTESIAN(Basis%atmshl(Latmshl)%ATMLST)%X
      YD=CARTESIAN(Basis%atmshl(Latmshl)%ATMLST)%Y
      ZD=CARTESIAN(Basis%atmshl(Latmshl)%ATMLST)%Z
      Laos=Basis%atmshl(Latmshl)%frstAO-1
!
      LQMAX=LCMAX+LDMAX-1
      LIM1DSQ=(LQMAX+3)/2
      LENTQQ=Lrange*Krange
      CDX=XD-XC
      CDY=YD-YC
      CDZ=ZD-ZC
      CDSQ=CDX*CDX+CDY*CDY+CDZ*CDZ

      LPQMAX=LPMAX+LQMAX-1
      LENTQPQ=Irange*Jrange*Lrange*Krange
      LIM1DSPQ=(LPQMAX+3)/2

      do I=1,LENTQPQ
        SSPQ(I)=ZERO
      end do ! I
!      write(UNIout,'(A,10I2)')'LENTQPQ ', LENTQPQ
!      write(UNIout,'(A,10I2)')'LENTQP ', LENTQP
```

```
!      write(UNIout,'(A,10I2)')'LENTQQ ', LENTQQ

      do I=1,LENTQP
      SSAB(I)=ZERO
      end do ! I

!
! Loop over primitive Gaussians.
      do Igauss=IGBGN,IGEND
      AS=Basis%gaussian(Igauss)%exp
      call FILLCC (LAMAX, Basis%gaussian(Igauss)%CONTRC, CA)

      do Jgauss=JGBGN,JGEND
      BS=Basis%gaussian(Jgauss)%exp
      call FILLCC (LBMAX, Basis%gaussian(Jgauss)%CONTRC, CB)
      EAB=AS+BS
      EIAB=ONE/EAB
      EIO2AB=PT5*EIAB
      KAB=-BS*AS*ABSQ/EAB
      EXPAB=ZERO
      IF(KAB.GT.CUT1)EXPAB=DEXP(KAB)
      PX=(AS*XA+BS*XB)/EAB
      PY=(AS*YA+BS*YB)/EAB
```

```
      PZ=(AS*ZA+BS*ZB)/EAB
      XAP=PX-XA
      XBP=PX-XB
      YAP=PY-YA
      YBP=PY-YB
      ZAP=PZ-ZA
      ZBP=PZ-ZB
      call GETCC1 (CCXAB,CCYAB,CCZAB,XAP,XBP,YAP,YBP,ZAP,ZBP, &
                                                 LAMAX,LBMAX)
!
      STERMAB=DSQRT(EIAB*PI_VAL)
      call GET1CS (S1CAB, STERMAB, EIO2AB, LPMAX)
      call GET2CS (SXAB, S1CAB, CCXAB, LAMAX, LBMAX)
      call GET2CS (SYAB, S1CAB, CCYAB, LAMAX, LBMAX)
      do I=1,LIM1DSP
      S1CAB(I)=S1CAB(I)*EXPAB
      end do ! I
      call GET2CS (SZAB, S1CAB, CCZAB, LAMAX, LBMAX)
      INTCa=0
!
      do I=Istart,Iend
      IX=INDIX(I)
      IY=INDIY(I)
```

```
      IZ=INDIZ(I)
      do J=Jstart,Jend
      JX=INDJX(J)
      JY=INDJY(J)
      JZ=INDJZ(J)
      IJX=IX+JX
      IJY=IY+JY
      IJZ=IZ+JZ
      INTCa=INTCa+1
      COEFAB=CA(I)*CB(J)
      SSAB(INTCa)=SSAB(INTCa)+SYAB(IJY)*SZAB(IJZ)*SXAB(IJX)*COEFAB
      end do ! J
      end do ! I
!
      call FILMAT (SSAB,OVRLAPP,MATlen,Iend,Jend, &
                    Iatmshl,Jatmshl,Irange,Jrange,Iaos,JaoS)
      do I=1,LENTQQ
      SSCD(I)=ZERO
      end do ! I

      do Kgauss=KGBGN,KGEND
      CS=Basis%gaussian(Kgauss)%exp
      call FILLCC (LCMAX, Basis%gaussian(Kgauss)%CONTRC, CC)
```

```
do Lgauss=LGBGN,LGEND

DS=Basis%gaussian(Lgauss)%exp

call FILLCC (LDMAX, Basis%gaussian(Lgauss)%CONTRC, CD)

ECD=CS+DS

EICD=ONE/ECD

EIO2CD=PT5*EICD

KCD=-DS*CS*CDSQ/ECD

EXPCD=ZERO

IF(KCD.GT.CUT1)EXPCD=DEXP(KCD)

QX=(CS*XC+DS*XD)/ECD

QY=(CS*YC+DS*YD)/ECD

QZ=(CS*ZC+DS*ZD)/ECD

XCQ=QX-XC

XDQ=QX-XD

YCQ=QY-YC

YDQ=QY-YD

ZCQ=QZ-ZC

ZDQ=QZ-ZD

call GETCC1 (CCXCD,CCYCD,CCZCD,XCQ,XDQ,YCQ,YDQ,ZCQ,ZDQ, &

                                                LCMAX,LDMAX)


STERMCD=DSQRT(EICD*PI_VAL)
```

```
      call GET1CS (S1CCD, STERMCD, EIO2CD, LQMAX)
      call GET2CS (SXCD, S1CCD, CCXCD, LCMAX, LDMAX)
      call GET2CS (SYCD, S1CCD, CCYCD, LCMAX, LDMAX)
      do I=1,LIM1DSQ
      S1CCD(I)=S1CCD(I)*EXPCD
      end do ! I
      call GET2CS (SZCD, S1CCD, CCZCD, LCMAX, LDMAX)
!
      INTCb=0
      do K=Kstart,Kend
      KX=INDIX(K)
      KY=INDIY(K)
      KZ=INDIZ(K)
      do L=Lstart,Lend
      LX=INDJX(L)
      LY=INDJY(L)
      LZ=INDJZ(L)
      KLX=KX+LX
      KLY=KY+LY
      KLZ=KZ+LZ
      INTCb=INTCb+1
      COEFCD=CC(K)*CD(L)
      SSCD(INTCb)=SSCD(INTCb)+SYCD(KLY)*SZCD(KLZ)*SXCD(KLX)*COEFCD
```

```
      end do ! J

      end do ! I

!

      call FILMAT (SSCD,OVRLAPQ,MATlen,Kend,Lend, &

                          Katmshl,Latmshl,Krange,Lrange,Kaos,LaoS)


!

      PQX=QX-PX

      PQY=QY-PY

      PQZ=QZ-PZ

      PQSQ=PQX*PQX+PQY*PQY+PQZ*PQZ


      EPQ=EAB+ECD

      EIPQ=ONE/EPQ

      EIO2PQ=PT5*EIPQ


      KPQ=-EAB*ECD*PQSQ/EPQ

      EXPPQ=ZERO

      IF(KPQ.GT.CUT1)EXPPQ=DEXP(KPQ)

      RX=(ECD*QX+EAB*PX)/EPQ

      RY=(ECD*QY+EAB*PY)/EPQ

      RZ=(ECD*QZ+EAB*PZ)/EPQ

      XPR=RX-PX
```

```
      XQR=RX-QX

      YPR=RY-PY

      YQR=RY-QY

      ZPR=RZ-PZ

      ZQR=RZ-QZ

      call GETCC1 (CCXPQ,CCYPQ,CCZPQ,XPR,XQR,YPR,YQR,ZPR,ZQR, &

                                                   LPMAX,LQMAX)


!

! Calculate the gaussian products

      STERMPQ=DSQRT(EIPQ*PI_VAL)

      call GET1CS (S1CPQ, STERMPQ, EIO2PQ, LPQMAX)

      call GET2CS (SXPQ, S1CPQ, CCXPQ, LPMAX, LQMAX)

      call GET2CS (SYPQ, S1CPQ, CCYPQ, LPMAX, LQMAX)

      do I=1,LIM1DSPQ

      S1CPQ(I)=S1CPQ(I)*EXPPQ

      end do ! I

      call GET2CS (SZPQ, S1CPQ, CCZPQ, LPMAX, LQMAX)


!


      INTC=0

      do I=Istart,Iend
```

```
      IX=INDeIX(I)
      IY=INDeIY(I)
      IZ=INDeIZ(I)
!      write(6,*)'IX,IY,IZ: ',IX,IY,IZ
!      call flush(6)
      do J=Jstart,Jend
      JX=INDeJX(J)
      JY=INDeJY(J)
      JZ=INDeJZ(J)
!      write(6,*)'JX,JY,JZ: ',JX,JY,JZ
!      call flush(6)
      do K=Kstart,Kend
      KX=INDeKX(K)
      KY=INDeKY(K)
      KZ=INDeKZ(K)
!      write(6,*)'KX,KY,KZ: ',KX,KY,KZ
!      call flush(6)
      do L=Lstart,Lend
      LX=INDeLX(L)
      LY=INDeLY(L)
      LZ=INDeLZ(L)
!      write(6,*)'LX,LY,LZ: ',LX,LY,LZ
!      call flush(6)
```

```
      IJKLX=IX+JX+KX+LX
      IJKLY=IY+JY+KY+LY
      IJKLZ=IZ+JZ+KZ+LZ
      COEFR=CA(I)*CB(J)*CC(K)*CD(L)
      INTC=INTC+1
      SSPQ(INTC)=SSPQ(INTC)+SXPQ(IJKLX)*SYPQ(IJKLY)*SZPQ(IJKLZ)*COEFR
      end do ! L
      end do ! K
      end do ! J
      end do ! I

      end do ! Jgauss
      end do ! Igauss
      end do ! Lgauss
      end do ! Kgauss
!     write(6,*)'INTC after P,Q loop: ',INTC

!
! End of loop over Gaussians.
!
      INTC2=0
      do I=1,Irange
      Ibasis=I+Iaos
```

```
      JF=Jrange
      do J=1,JF
      Jbasis=J+Jaos
      KF=Krange
      do K=1,KF
      Kbasis=K+Kaos
      LF=Lrange
      do L=1,LF
      Lbasis=L+Laos
      INTC2=INTC2+1
      write(6,*)'Ibasis; Jbasis; Kbasis: Lbasis; SSPQ',Ibasis, &
                 Jbasis,Kbasis,Lbasis,SSPQ(INTC2)
      if(dabs(SSPQ(INTC2)).le.1.0E-08)cycle
!     write(6,*)',Ibasis, Jbasis, SSPQ = ',Ibasis, Jbasis, SSPQ(INTC2)
!     write(6,*)'Ibasis; Jbasis; Nij', Ibasis, Jbasis, Nij(Ibasis,Jbasis)
!     write(6,*)'Kbasis; Lbasis; Nij', Kbasis, Lbasis, Nij(Kbasis,Lbasis)
      if((Ibasis.eq.Kbasis).and.(Jbasis.eq.Lbasis))then
      Nij(Ibasis,Jbasis)=1.0D0/(DSQRT(SSPQ(INTC2)))
      Nij(Jbasis,Ibasis)=Nij(Ibasis,Jbasis)
!     write(6,*)'Ibasis, Jbasis, SSPQ = ',Ibasis, Jbasis, SSPQ(INTC2)
      end if
      if((Ibasis.eq.Jbasis).and.(Kbasis.eq.Lbasis))then
      Nij(Ibasis,Kbasis)=1.0D0/(DSQRT(SSPQ(INTC2)))
```

```
      Nij(Kbasis,Ibasis)=Nij(Ibasis,Kbasis)
      end if
      ICOUNT=ICOUNT+1
      GGPQ(ICOUNT)%Ibasis=Ibasis
      GGPQ(ICOUNT)%Jbasis=Jbasis
      GGPQ(ICOUNT)%Kbasis=Kbasis
      GGPQ(ICOUNT)%Lbasis=Lbasis
      GGPQ(ICOUNT)%value=SSPQ(INTC2)
      end do ! L
      end do ! K
      end do ! J
      end do ! I
!     write(6,*)'INTC,INTC2 after I,J,K,L loop: ',Iatmshl,Jatmshl, &
                 Katmshl,Latmshl,': ',INTC,INTC2
!     write(6,*)'ICOUNT after I,J,K,L loop: ',ICOUNT

      do I=1,Irange
      Ibasis=I+Iaos
      do J=1,Jrange
      Jbasis=J+Jaos
!      write(UNIout,'(A,2I3,D18.6)')'Ibasis ; Jbasis ; Nij', &
                                    Ibasis, Jbasis, Nij(Ibasis,Jbasis)
      end do
```

```
      end do

      end do ! Latmshl
      end do ! Katmshl
      end do ! Jatmshl
      end do ! Iatmshl
      end do ! Lshell
      end do ! Kshell
      end do ! Jshell
      end do ! Ishell
      TOTint=ICOUNT
      write(UNIout,'(A,I10)') 'TOTint = ',TOTint

! End of loop over shells.
      do I=1,Irange
      Ibasis=I+Iaos
      do J=1,Jrange
      Jbasis=J+Jaos
      !write(UNIout,'(A,2I3,D18.6)')'Ibasis ; Jbasis ; Nij', Ibasis, &
                                     Jbasis, Nij(Ibasis,Jbasis)
      end do
      end do
```

```
do ICOUNT=1,TOTint
GGPQ(ICOUNT)%value=GGPQ(ICOUNT)%value*Nij(GGPQ(ICOUNT)%Ibasis,&
                   GGPQ(ICOUNT)%Jbasis)*Nij(GGPQ(ICOUNT)%Kbasis,&
                   GGPQ(ICOUNT)%Lbasis)
! write(UNIout,'(A,2I3,D18.6)')'Ibasis ; Jbasis ; Nij',&
!                               GGPQ(ICOUNT)%Ibasis, &
!                               GGPQ(ICOUNT)%Jbasis, &
!                               Nij(GGPQ(ICOUNT)%Ibasis,
!                               GGPQ(ICOUNT)%Jbasis) &
end do

write(UNIout,'(A)') 'Ibasis ; Jbasis ; Kbasis ; Lbasis ; GGPQ'
do ICOUNT=1,TOTint
write(UNIout,'(4I3,D18.6)') GGPQ(ICOUNT)%Ibasis, &
                            GGPQ(ICOUNT)%Jbasis, &
                            GGPQ(ICOUNT)%Kbasis, &
                            GGPQ(ICOUNT)%Lbasis, &
                            GGPQ(ICOUNT)%value
end do
write(UNIout,*)'OVERLAPP'
call PRT_matrix (OVRLAPP, MATlen, Basis%Nbasis)
write(UNIout,*)'OVERLAPQ'
call PRT_matrix (OVRLAPQ, MATlen, Basis%Nbasis)
```

```
      write(UNIout,*)'Nij'
      call PRT_matrix (Nij, Basis%Nbasis, Basis%Nbasis)
      call Fitting_Atoms
!
! End of routine GG_PROD_INT
      call PRG_manager ('exit', 'GG_PROD_INT', '1EINT%GG')
      return
CONTAINS!>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>!
      subroutine Fitting_Atoms
!*****************************************************************
!     Date last modified: Dec 17, 2012                  Version 1.0 *
!     Author: Devin G. S. Nippard                                   *
!     Description:                                                  *
!*****************************************************************
! Modules:

      implicit none

  ! Local scalars:
      integer :: Iatom,Iiter,Ic,Kbasis,Lbasis,Max_iter,IterC
      double precision :: check,TrPS,TRACEMS
  ! Local arrays:
      double precision ,ALLOCATABLE, dimension(:,:) :: F_hf,F_fit,Mkl,SSA,PMOSQ
```

```
      double precision ,ALLOCATABLE, dimension(:,:) :: F_diff,C_old,C_new,Pfit
      double precision , dimension(Basis%Nbasis) :: EIGVAL
      write(6,*)'It made it to FA'
      call flush(6)
!      Call PRT_matrix (PMO,MATlen,Basis%Nbasis)
!     call flush(6)
      if(.not.allocated(Mkl))then
        allocate (Mkl(Basis%Nbasis,Basis%Nbasis),&
                  F_diff(Basis%Nbasis,Basis%Nbasis), &
                  F_hf(Basis%Nbasis,Basis%Nbasis), &
                  F_fit(Basis%Nbasis,Basis%Nbasis), &
  SSA(Basis%Nbasis,Basis%Nbasis), &
  PMOSQ(Basis%Nbasis,Basis%Nbasis))
      else
        if(Basis%Nbasis.ne.size(Mkl,1))then
          deallocate (Mkl,F_diff,F_hf,F_fit,SSA,PMOSQ)
          allocate (Mkl(Basis%Nbasis,Basis%Nbasis), &
                    F_diff(Basis%Nbasis,Basis%Nbasis), &
            F_hf(Basis%Nbasis,Basis%Nbasis), &
            F_fit(Basis%Nbasis,Basis%Nbasis), &
                    SSA(Basis%Nbasis,Basis%Nbasis), &
                    PMOSQ(Basis%Nbasis,Basis%Nbasis))
        end if
```

```
end if

call UPT_to_SQS (OVRLAPP, MATlen, OVRLAP, Basis%Nbasis)
call UPT_to_SQS (PM0, MATlen, PM0SQ, Basis%Nbasis)

write(6,*)'Byond get initalization '
call flush(6)

IterC=0
Max_iter=25

write(6,*)'Hello'
call flush(6)

Mkl(1:Basis%Nbasis,1:basis%Nbasis)=0.0D0
C_old(1:Basis%Nbasis,1:basis%Nbasis)=0.0D0
TrPS=0.0D0

do Iiter=1,Max_iter
do Iatom=1,Natoms

do Kbasis = 1,Basis%Nbasis
do Lbasis = 1,Basis%Nbasis
```

```
do Ic = 1,TOTint

if ((Kbasis.eq.GGPQ(Ic)%Kbasis).and.(Lbasis.eq.GGPQ(Ic)%Lbasis)) then
  Mkl(GGPQ(Ic)%Ibasis,GGPQ(Ic)%Jbasis) = GGPQ(Ic)%value
  Mkl(GGPQ(Ic)%Jbasis,GGPQ(Ic)%Ibasis) = GGPQ(Ic)%value
endif
end do !Ic
F_hf(Kbasis,Lbasis) = TRACEMS(PMOSQ,Mkl)
Pfit = MATMUL(C_old,TRANSPOSE(C_old))
F_fit(Kbasis,Lbasis) = TRACEMS(C_old,Mkl)
end do !L
end do !K

F_diff=F_hf - F_fit

call DETlinW (SSA, F_diff , C_new , EIGVAL)

Pfit = MATMUL(C_new,TRANSPOSE(C_new))

TrPS = TrPS + TraceMS(Pfit,SSA)

end do !Iatom
```

```
      check = Nelectrons - TrPS


      if ((check.le.1.0E-08))exit

      IterC=IterC+1

      end do !Iiter


      write(UNIout,'(A,I3)')'IterC',IterC

      Call PRT_matrix (Pfit, Basis%Nbasis, Basis%Nbasis)

      Call PRT_matrix (PMOSQ, Basis%Nbasis, Basis%Nbasis)

      return

      end subroutine Fitting_Atoms

!END CONTAINS                        !

      end subroutine GG_prod_int
```

A.2 Molecular Orbital Radial Electron Density

```
      subroutine RADIAL_MO_DENSITY_MESH

!*********************************************************************************

!     Date last modified: Sept 26, 2013

!     Authors: Devin Nippard

!     Description:

!                 Calculate the radial MO density on a set of grid points (mesh)

!*********************************************************************************

! Modules:

      USE program_files

      USE program_interface

      USE QM_interface

      USE QM_defaults

      USE QM_objects

      USE type_grids

      USE type_density

      USE type_molecule

      USE type_elements

      USE scalar_objects

      USE N_integration


      implicit none
```

```
      integer :: Ipoint,IJ,FAC,Nbasis,NMOs,Mu,Nu,MATlen,Iatom,IMO,File_unit
      double precision :: Xg,Yg,Zg,Xa,Ya,Za
      double precision :: BeckeW,radSq,MO_den
      double precision, allocatable,dimension(:) :: MOradA
!
      call PRG_manager ('enter','RADIAL_MO_DENSITY_MESH', 'UTILITY')

      call GET_object ('MOL', 'GRIDS', 'MESH')
      call GET_object ('QM', 'CMO', 'RHF')
!
      call BLD_NI_XI

      MATlen=size(PMO)
      NMOs=CMO%NoccMO
      Nbasis=size(CMO%coeff,1)
      allocate (AOprod(1:MATlen))
      allocate (MOradA(NGridPoints))

      do Iatom=1,Natoms
        Xa=CARTESIAN(Iatom)%X
        Ya=CARTESIAN(Iatom)%Y
        Za=CARTESIAN(Iatom)%Z
        do IMO=1,NMOs
```

```
          do Ipoint=1,NGridPoints
          Xg=grid_point(Ipoint)%x
          Yg=grid_point(Ipoint)%y
          Zg=grid_point(Ipoint)%z
          radSq = (Xg-Xa)**2+(Yg-Ya)**2+(Zg-Za)**2
          call BeckeW1(Xg, Yg, Zg, Iatom, XI, BeckeW)
          call AO_products (Xg, Yg, Zg)
          MO_den=0.0d0
          do Mu=1,Nbasis
          do Nu=1,Mu
            FAC=1.0d0
            if(Mu.ne.Nu)Fac=2.0d0
            IJ=Mu*(Mu-1)/2+Nu
            MO_den=MO_den+FAC*CMO%coeff(Mu,IMO)*CMO%coeff(Nu,IMO)*AOprod(IJ)
          end do ! Nu
          end do ! Mu
          MOradA(Ipoint)=BeckeW*radSq*MO_den
        end do !Ipoint
! Now print the radial densities for Iatom and IMO
        if(MoleculePrint.or.AtomPrint)then
          call BLD_plot_file_MO ('RDEN_MESH', NGridPoints, Iatom, IMO, File_unit)
        else
          File_unit=UNIout
```

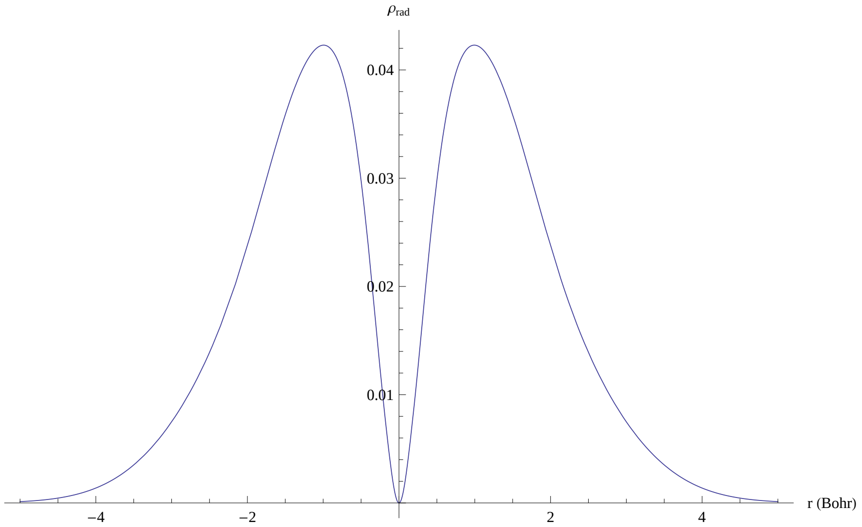
```
end if
write(File_unit,'(a,i8,a,i8)')'MO: ',IMO,' for atom: ',Iatom
call PRT_DENSITY_MESH (File_unit, MOradA, NGridPoints)
if(MoleculePrint.or.AtomPrint)close (unit=File_unit)
end do ! IMO
end do ! Iatom

call PRG_manager ('exit','RADIAL_MO_DENSITY_MESH', 'UTILITY')

end subroutine RADIAL_MO_DENSITY_MESH
```

Appendix B

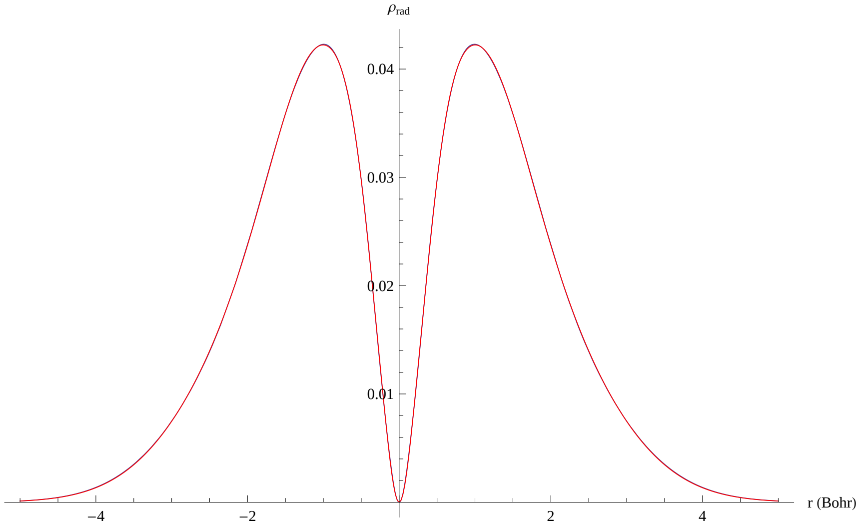
Radial Electron Densities

B.1 Individual Atom Densities in 1 Dimension

B.1.1 Period One Atoms (H-He)

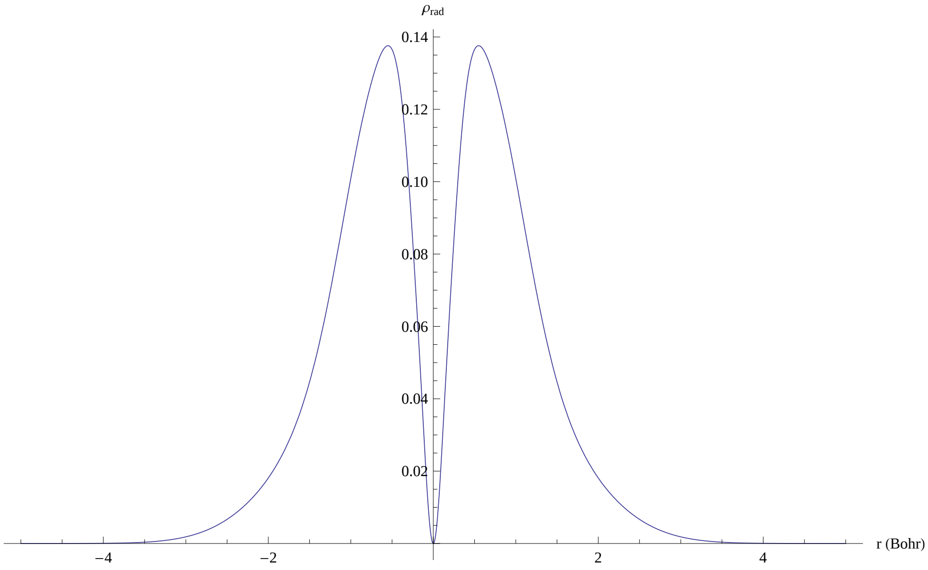


(a) Fitted radial density.

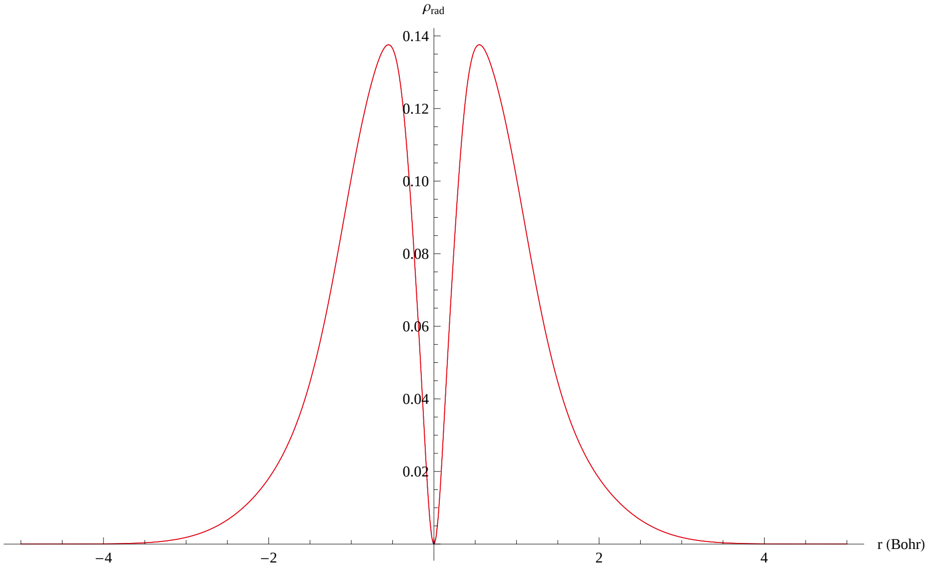


(b) Fitted (Blue) vs actual (Red) radial density.

Figure B.1.1: (a) Fit of the radial electron density of the H atom and (b) comparison of the fit and the actual radial electron density.



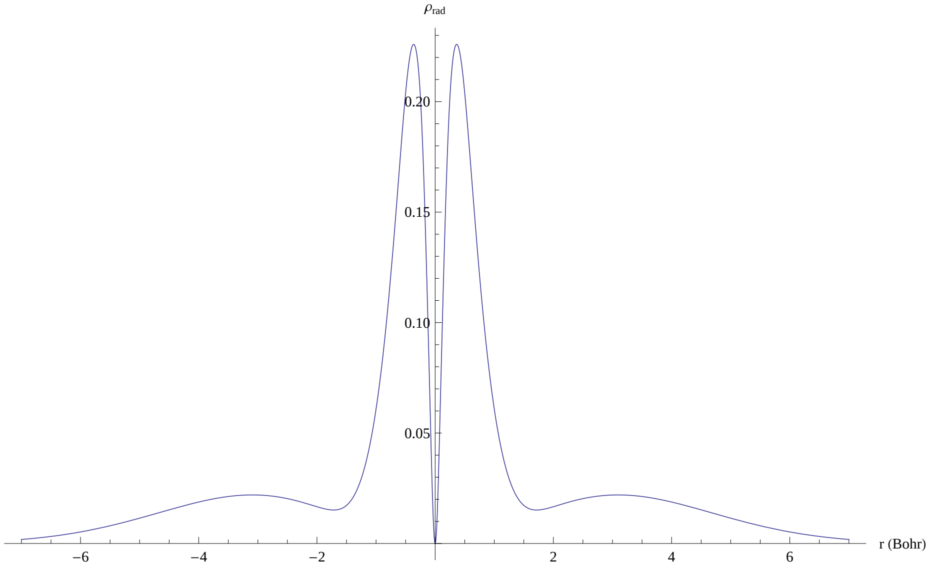
(a) Fitted radial density.



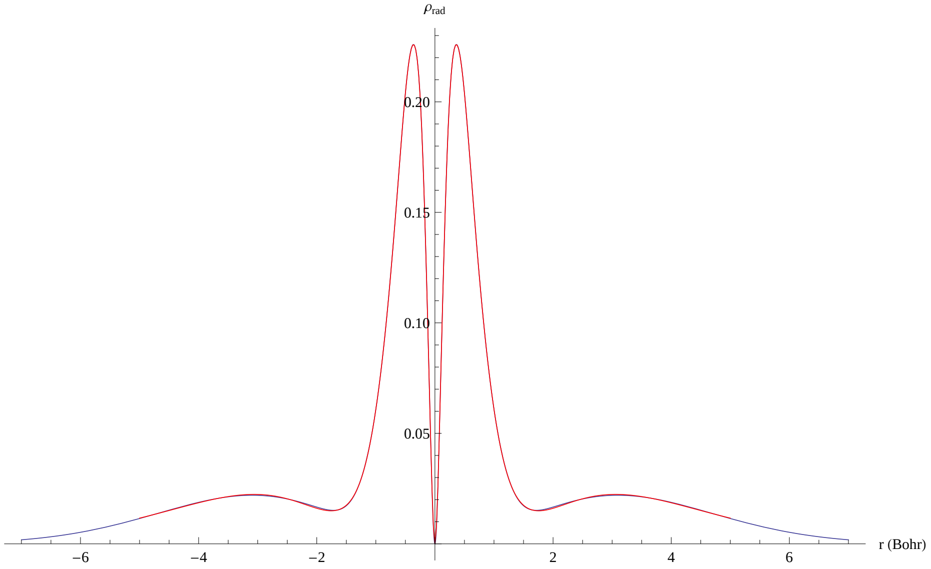
(b) Fitted (Blue) vs actual (Red) radial density.

Figure B.1.2: (a) Fit of the radial electron density of the He atom and (b) comparison of the fit and the actual radial electron density.

B.1.2 Period Two Atoms (Li-Ne)

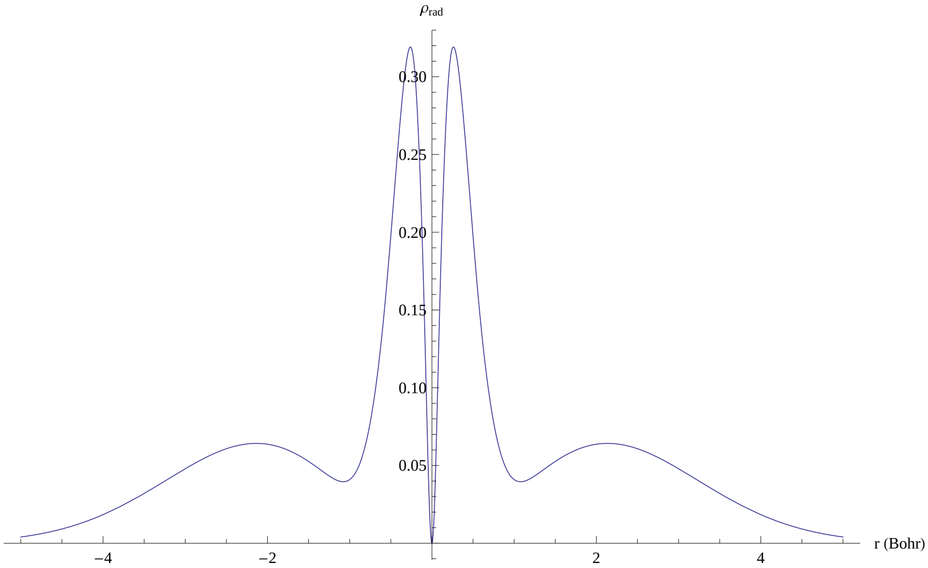


(a) Fitted radial density.

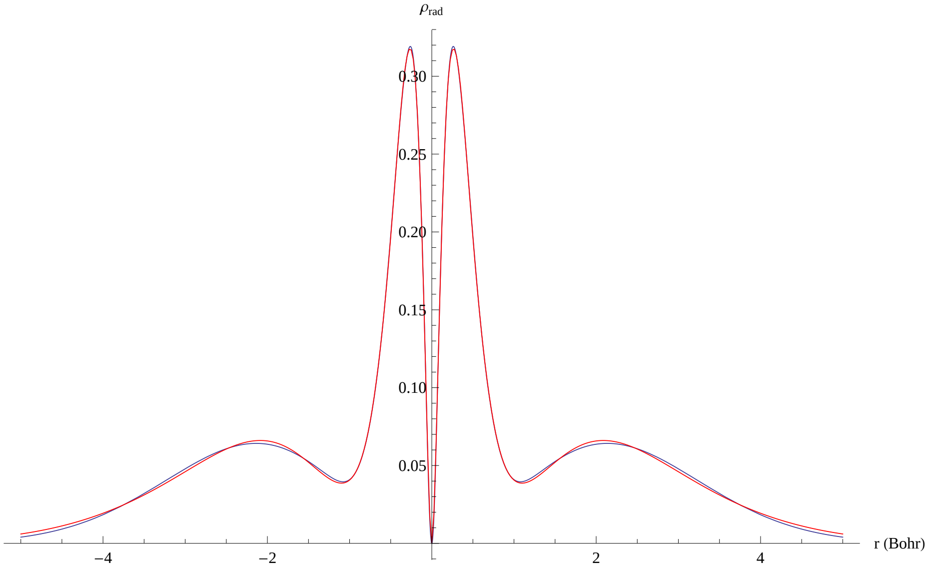


(b) Fitted (Blue) vs actual (Red) radial density.

Figure B.1.3: (a) Fit of the radial electron density of the Li atom and (b) comparison of the fit and the actual radial electron density.

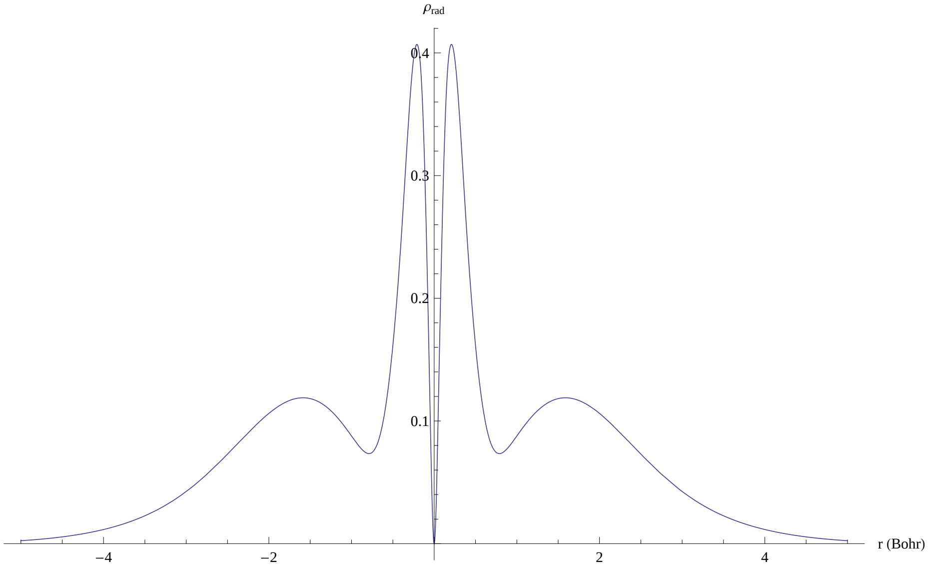


(a) Fitted radial density.

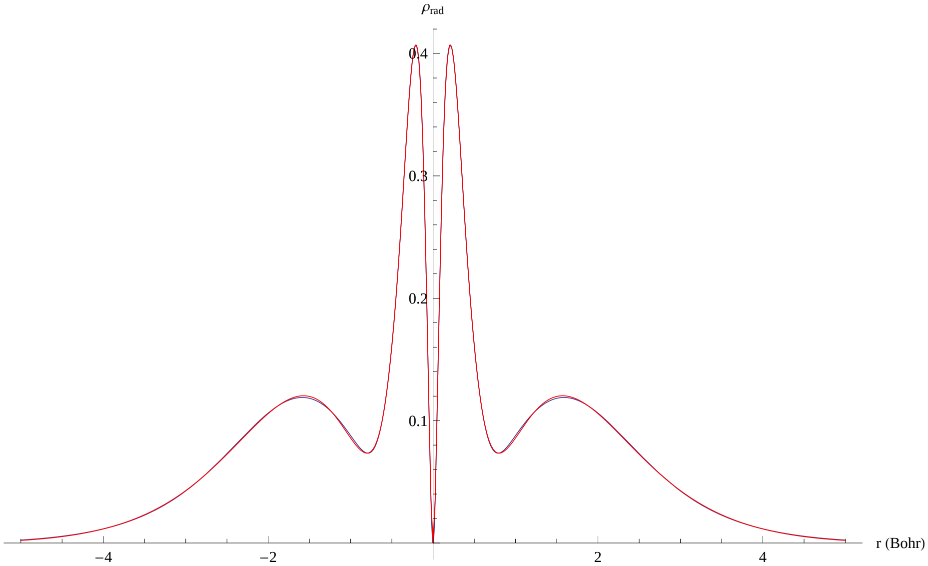


(b) Fitted (Blue) vs actual (Red) radial density.

Figure B.1.4: (a) Fit of the radial electron density of the Be atom and (b) comparison of the fit and the actual radial electron density.

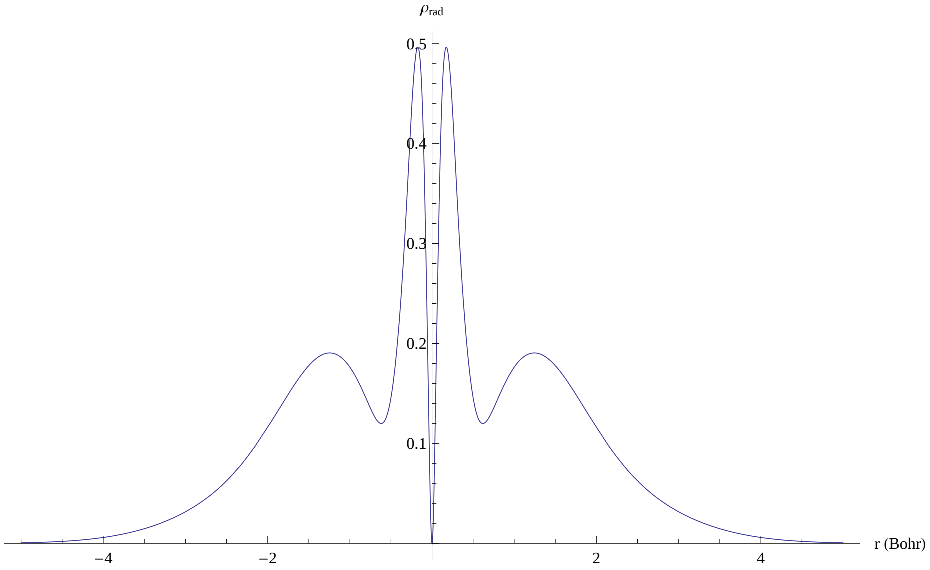


(a) Fitted radial density.

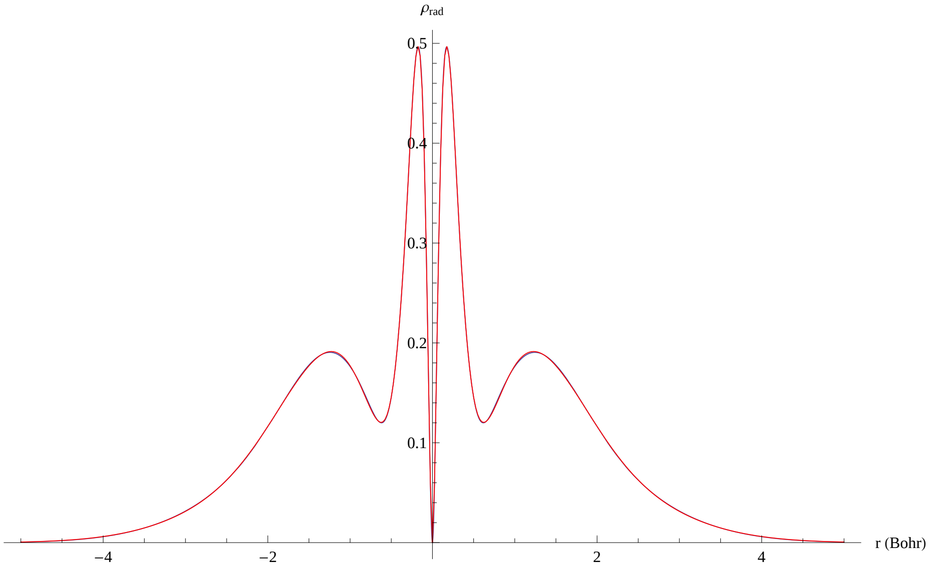


(b) Fitted (Blue) vs actual (Red) radial density.

Figure B.1.5: (a) Fit of the radial electron density of the B atom and (b) comparison of the fit and the actual radial electron density.

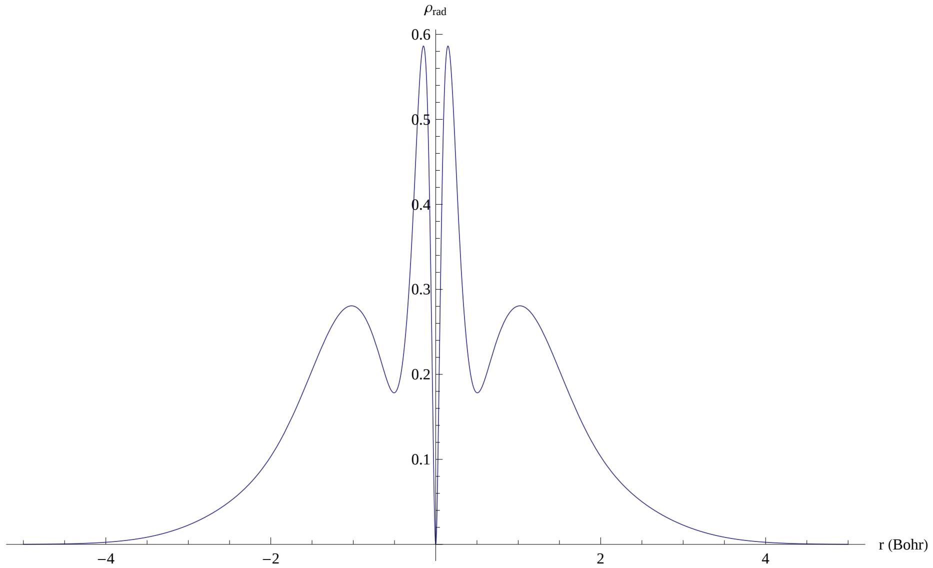


(a) Fitted radial density.

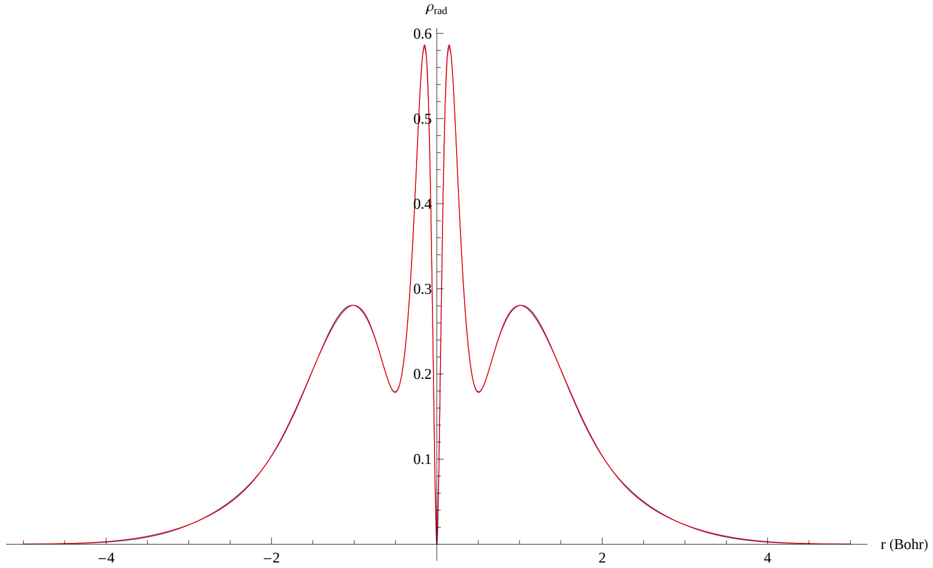


(b) Fitted (Blue) vs actual (Red) radial density.

Figure B.1.6: (a) Fit of the radial electron density of the C atom and (b) comparison of the fit and the actual radial electron density.

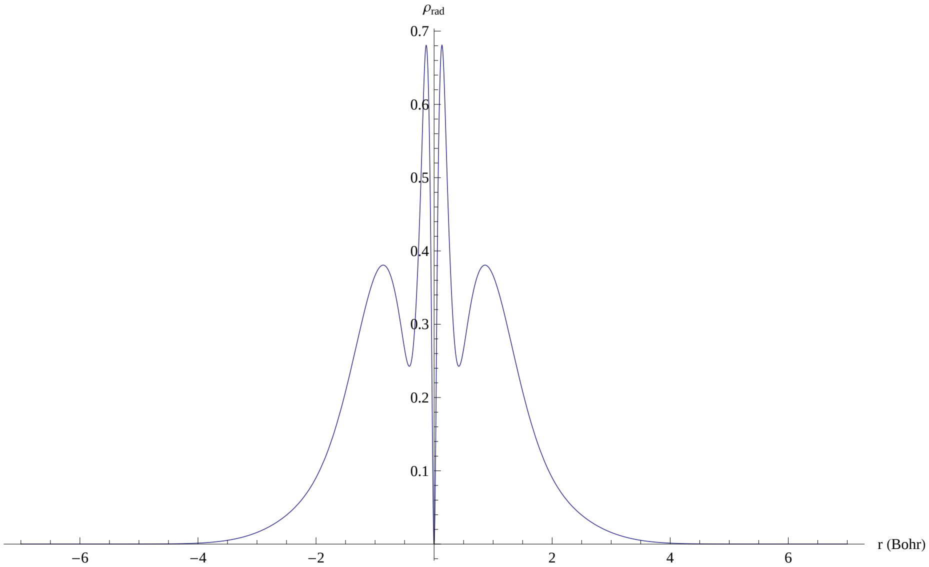


(a) Fitted radial density.

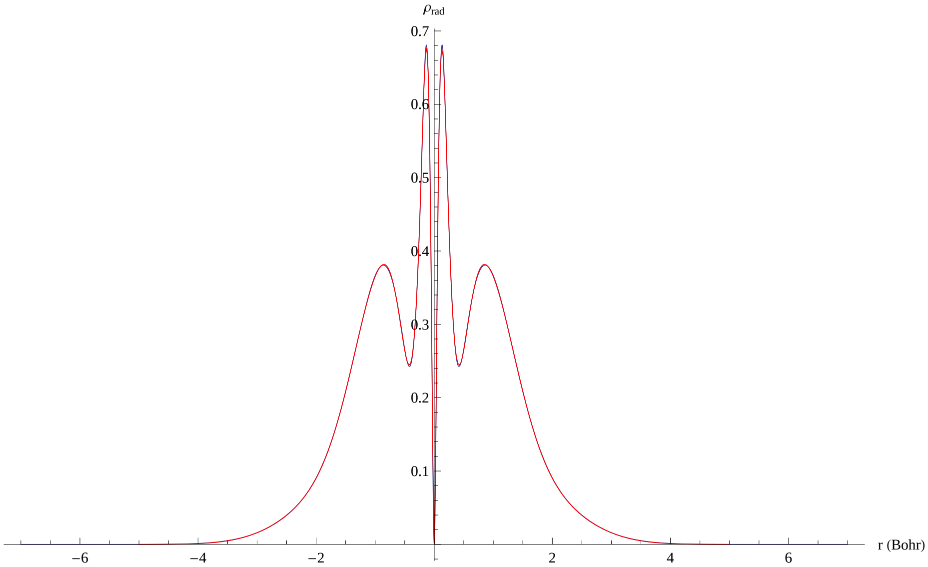


(b) Fitted (Blue) vs actual (Red) radial density.

Figure B.1.7: (a) Fit of the radial electron density of the N atom and (b) comparison of the fit and the actual radial electron density.

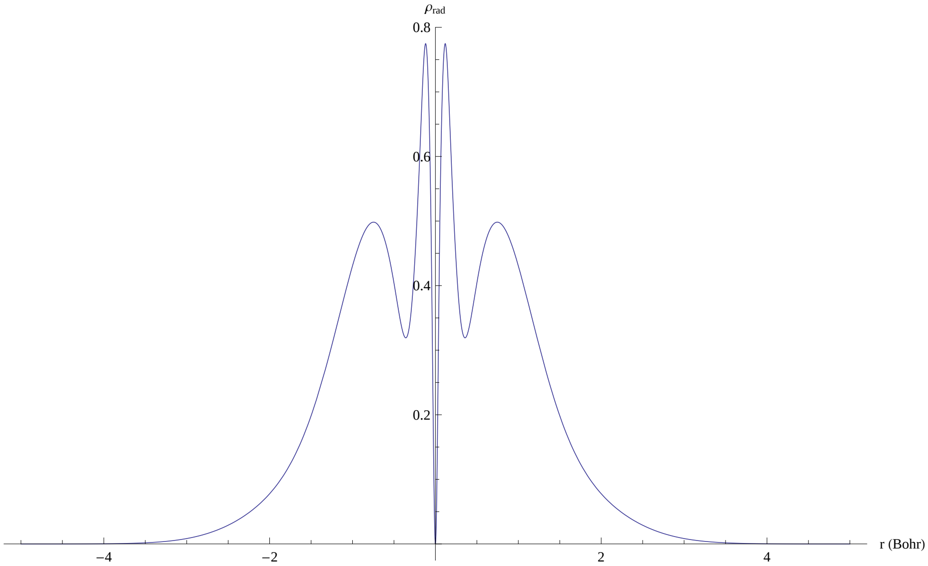


(a) Fitted radial density.

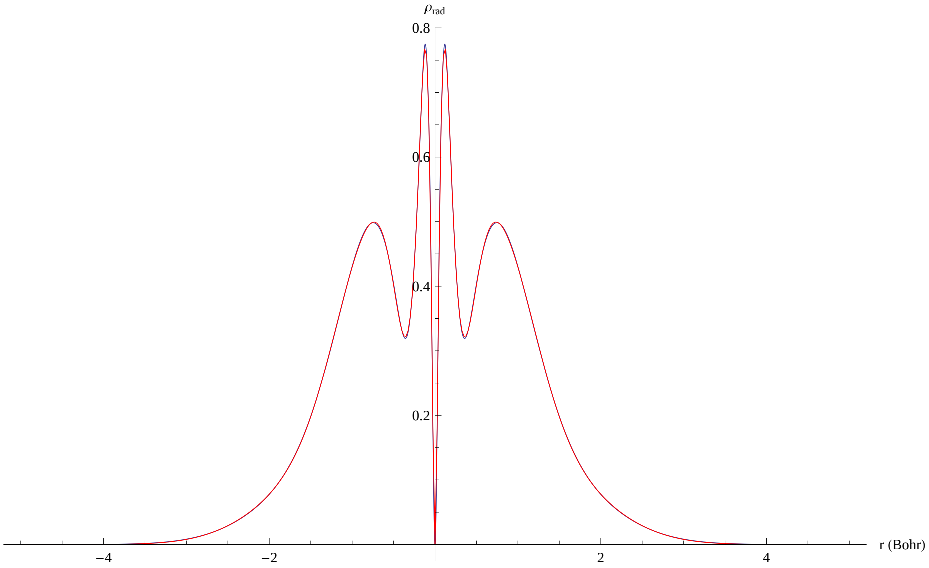


(b) Fitted (Blue) vs actual (Red) radial density.

Figure B.1.8: (a) Fit of the radial electron density of the O atom and (b) comparison of the fit and the actual radial electron density.

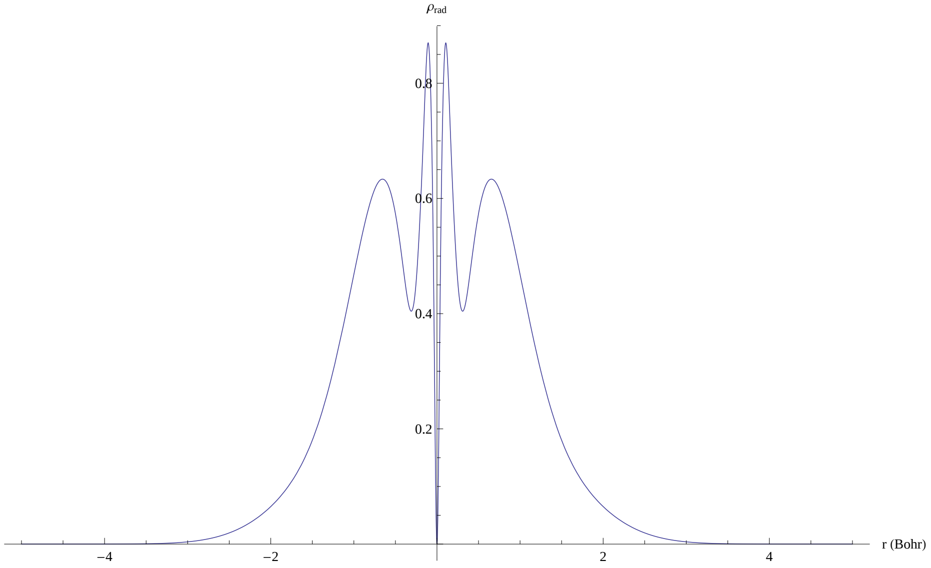


(a) Fitted radial density.

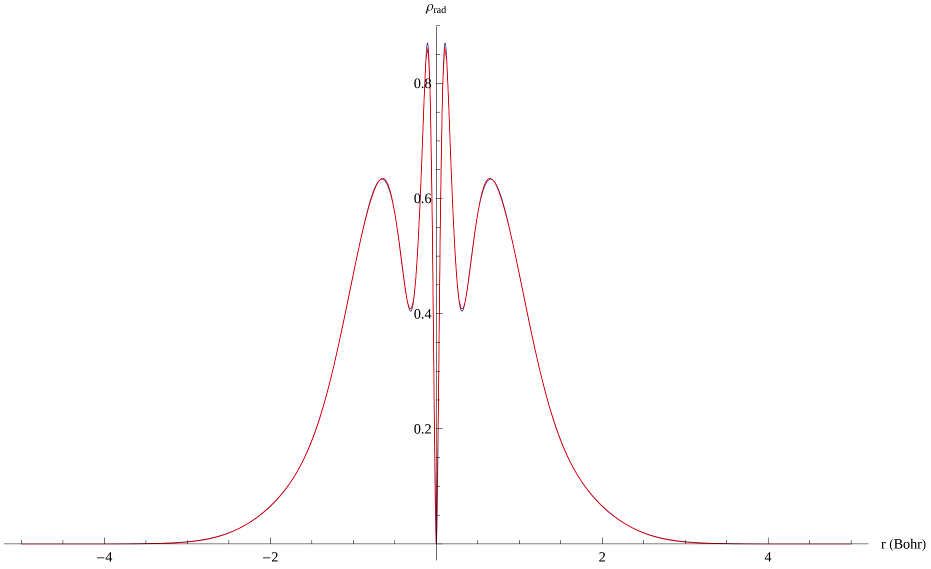


(b) Fitted (Blue) vs actual (Red) radial density.

Figure B.1.9: (a) Fit of the radial electron density of the F atom and (b) comparison of the fit and the actual radial electron density.



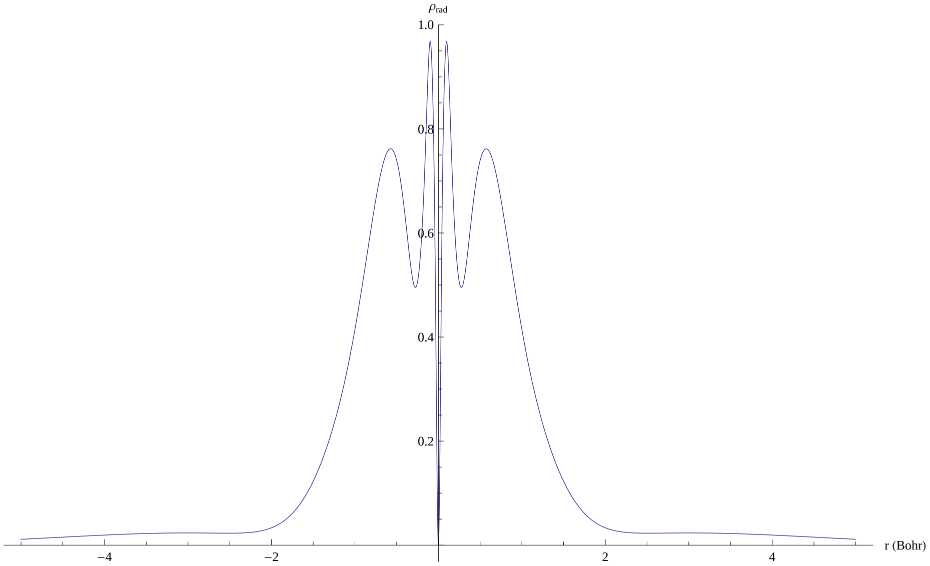
(a) Fitted radial density.



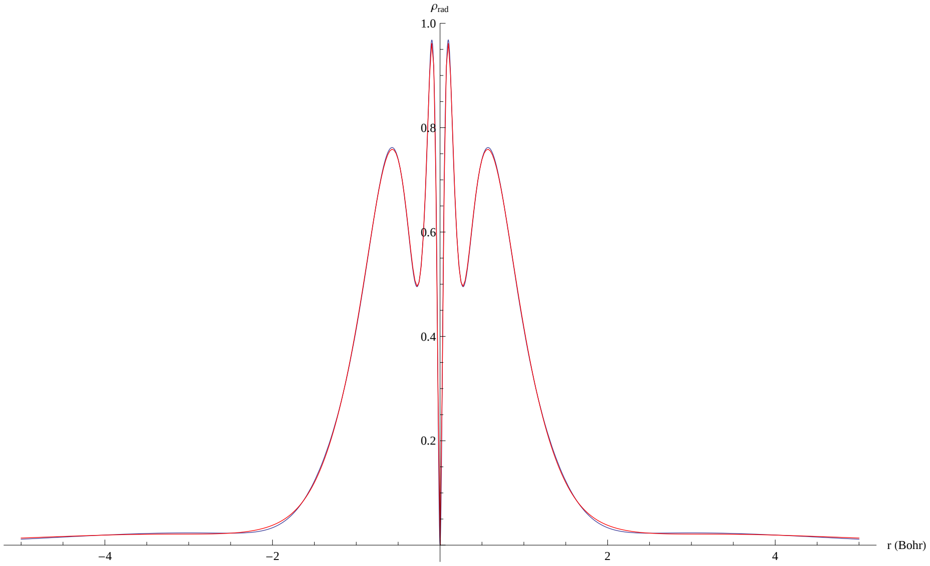
(b) Fitted (Blue) vs actual (Red) radial density.

Figure B.1.10: (a) Fit of the radial electron density of the Ne atom and (b) comparison of the fit and the actual radial electron density.

B.1.3 Period Three Atoms (Na-Ar)

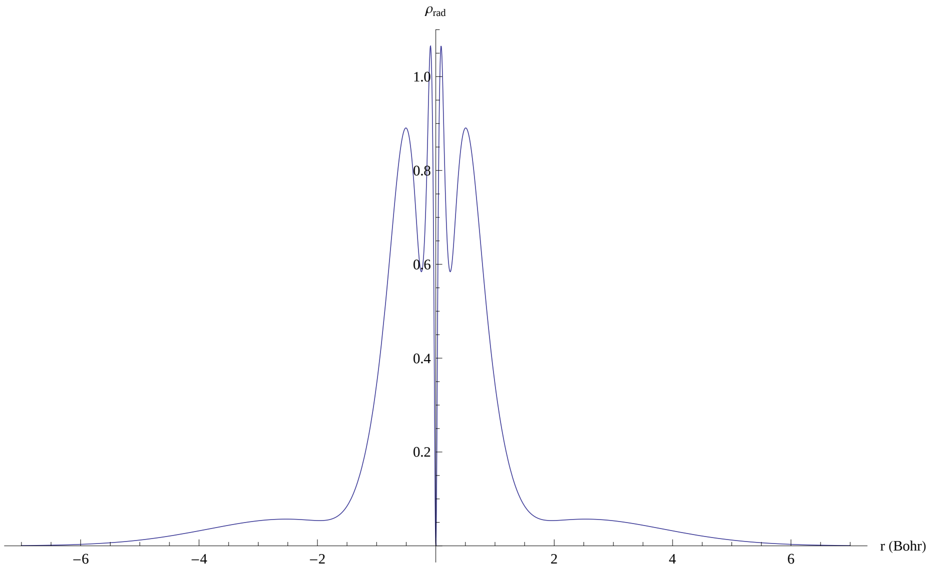


(a) Fitted radial density.

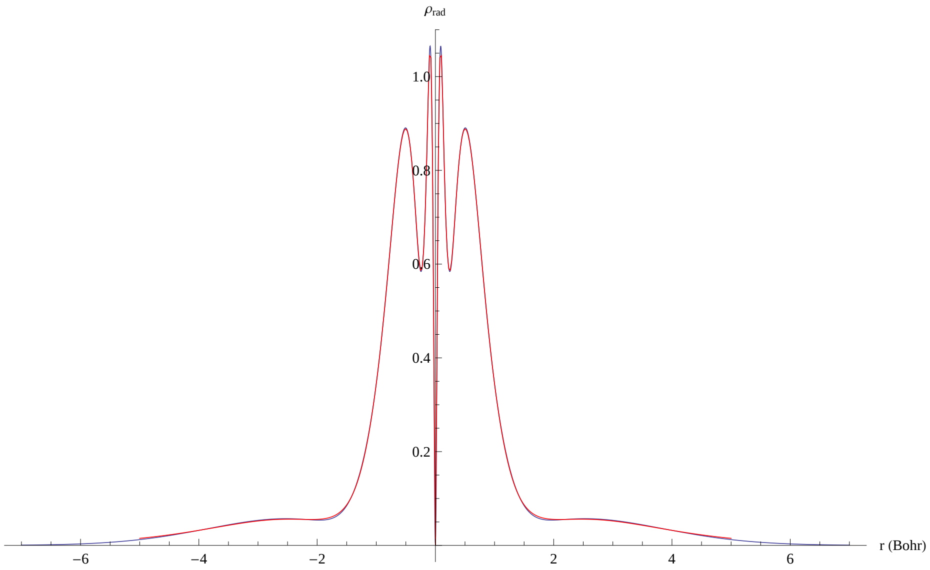


(b) Fitted (Blue) vs actual (Red) radial density.

Figure B.1.11: (a) Fit of the radial electron density of the Na atom and (b) comparison of the fit and the actual radial electron density.

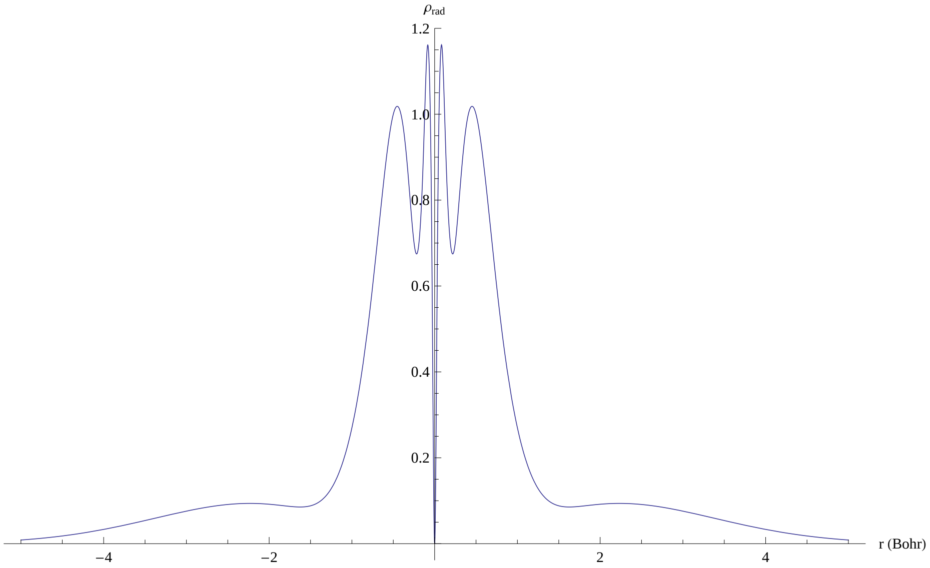


(a) Fitted radial density.

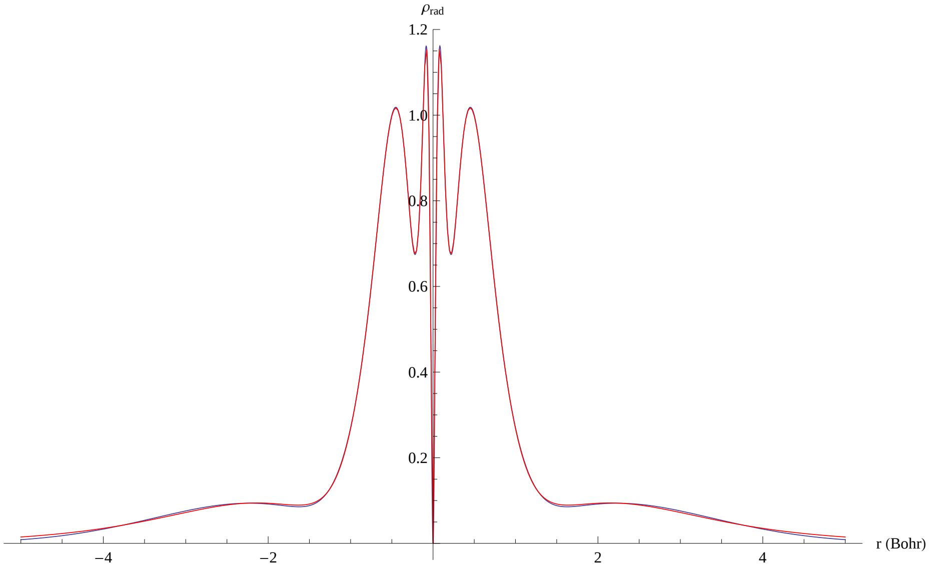


(b) Fitted (Blue) vs actual (Red) radial density.

Figure B.1.12: (a) Fit of the radial electron density of the Mg atom and (b) comparison of the fit and the actual radial electron density.

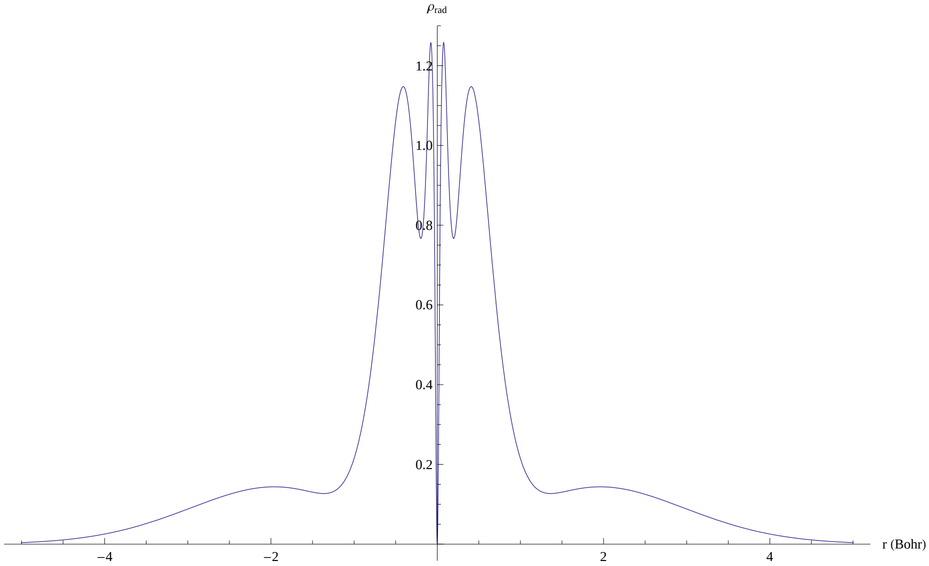


(a) Fitted radial density.

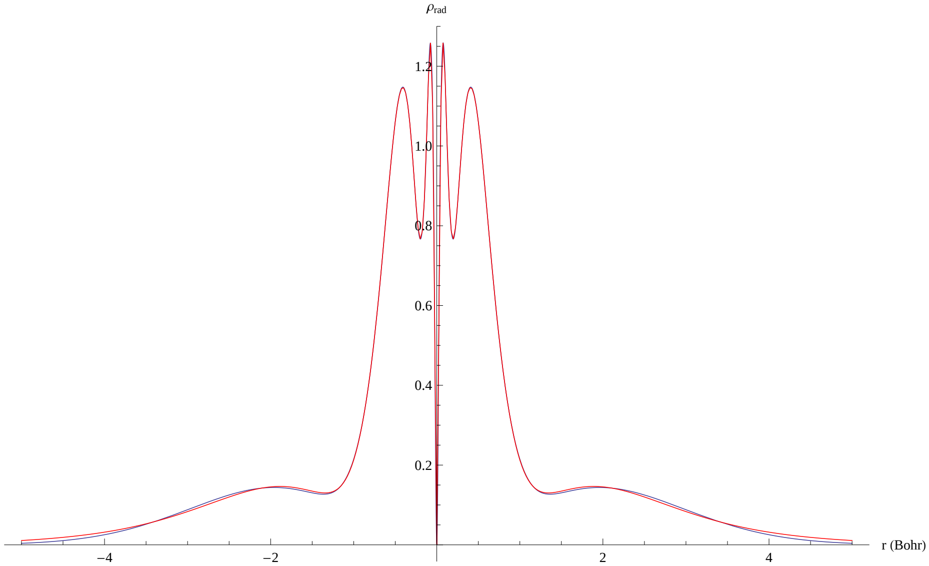


(b) Fitted (Blue) vs actual (Red) radial density.

Figure B.1.13: (a) Fit of the radial electron density of the Al atom and (b) comparison of the fit and the actual radial electron density.

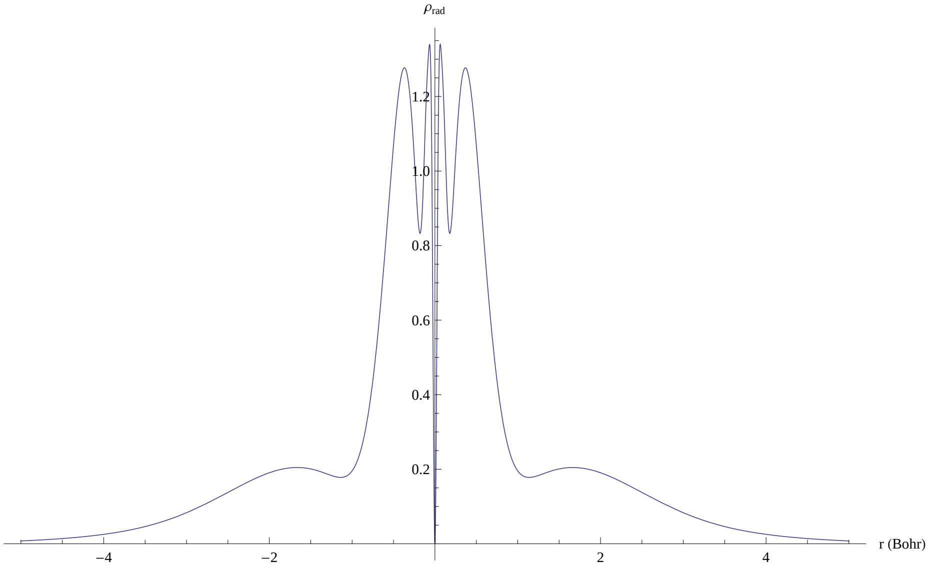


(a) Fitted radial density.

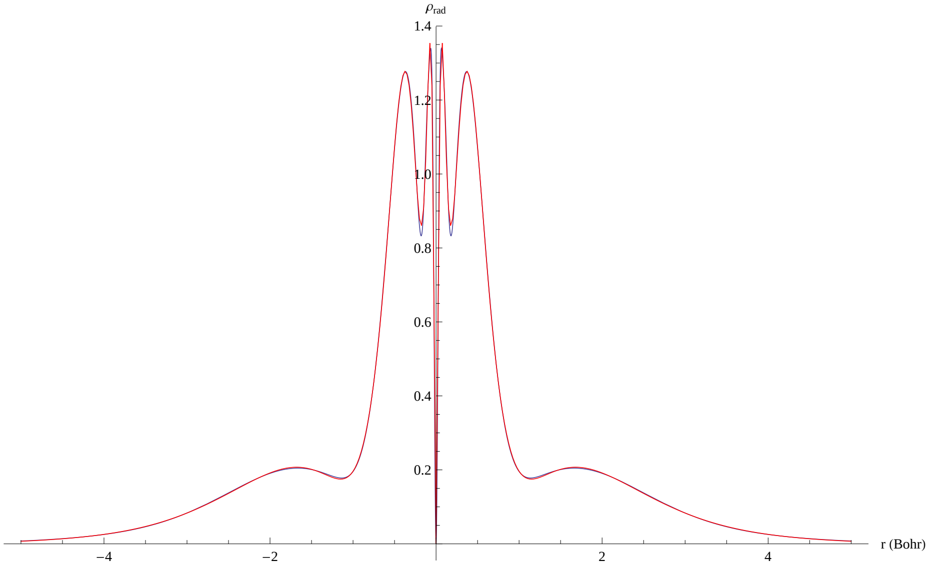


(b) Fitted (Blue) vs actual (Red) radial density.

Figure B.1.14: (a) Fit of the radial electron density of the Si atom and (b) comparison of the fit and the actual radial electron density.

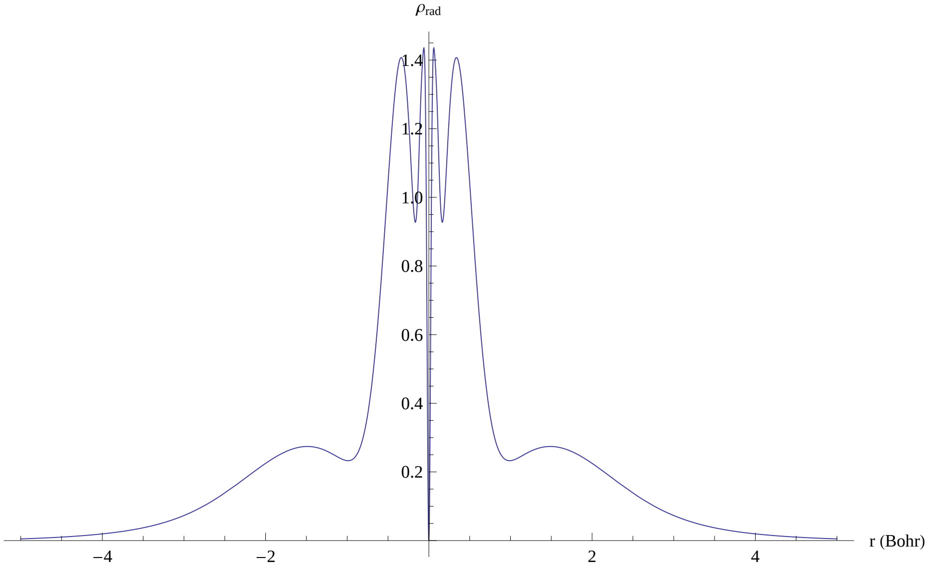


(a) Fitted radial density.

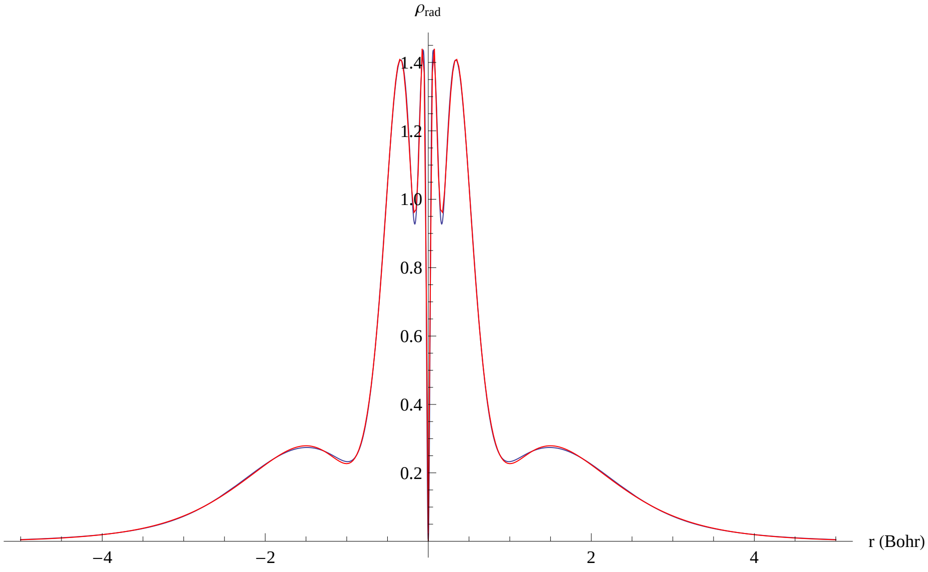


(b) Fitted (Blue) vs actual (Red) radial density.

Figure B.1.15: (a) Fit of the radial electron density of the P atom and (b) comparison of the fit and the actual radial electron density.

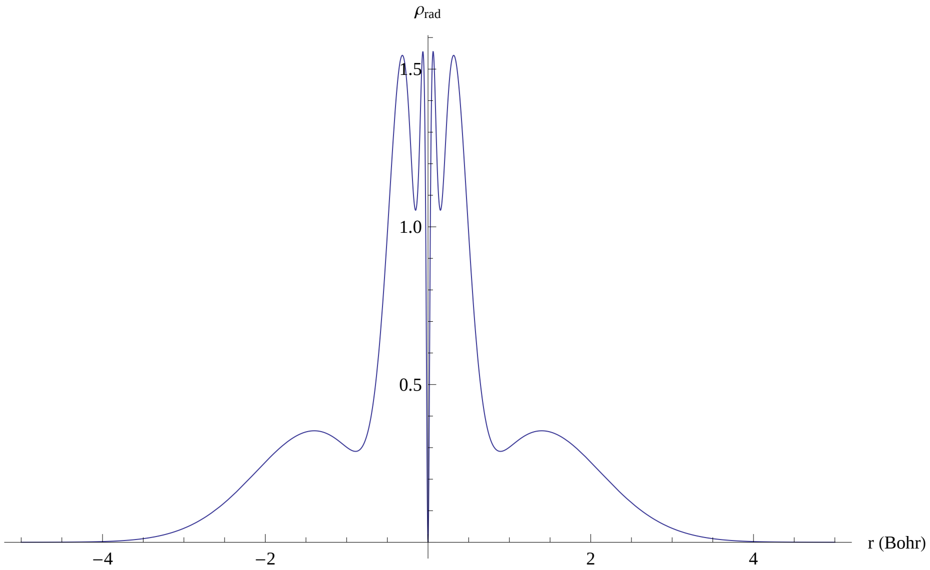


(a) Fitted radial density.

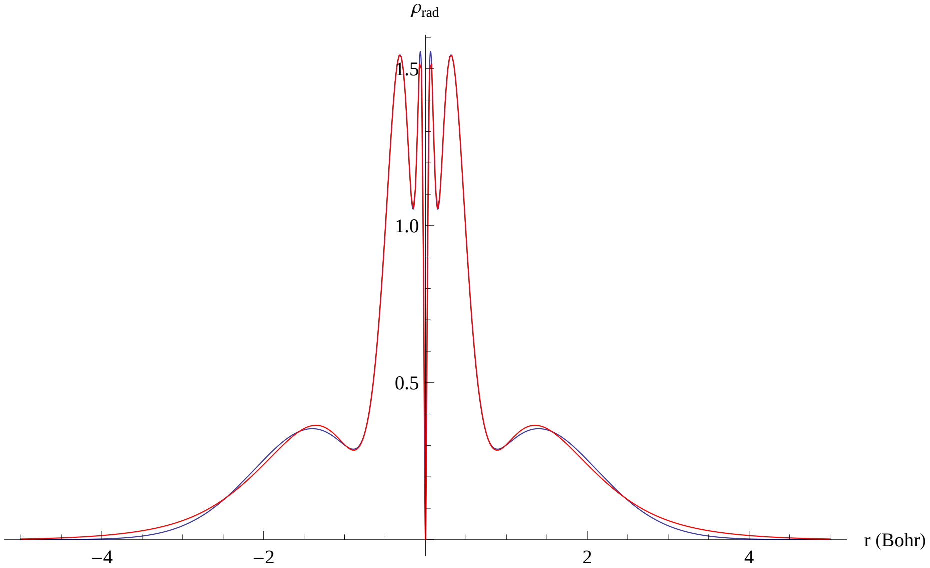


(b) Fitted (Blue) vs actual (Red) radial density.

Figure B.1.16: (a) Fit of the radial electron density of the S atom and (b) comparison of the fit and the actual radial electron density.

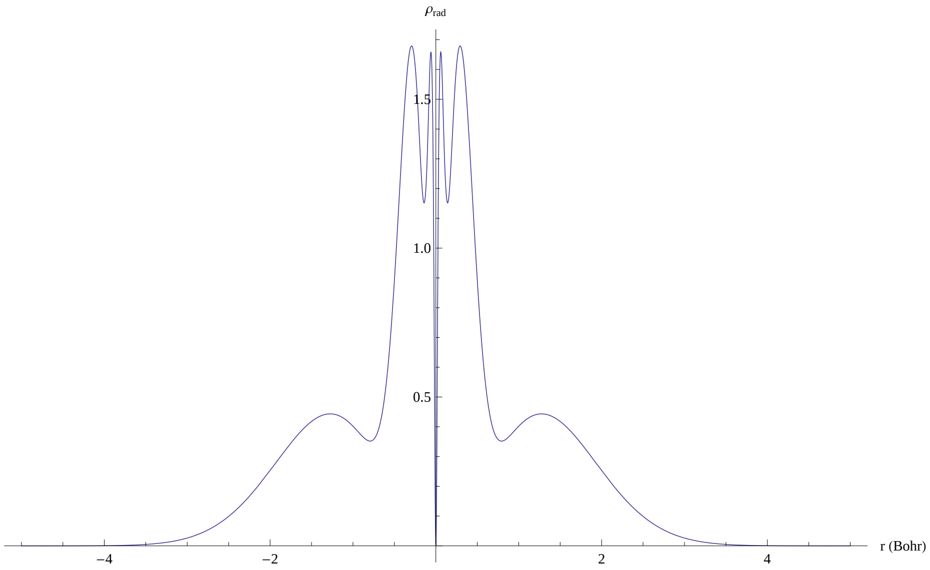


(a) Fitted radial density.

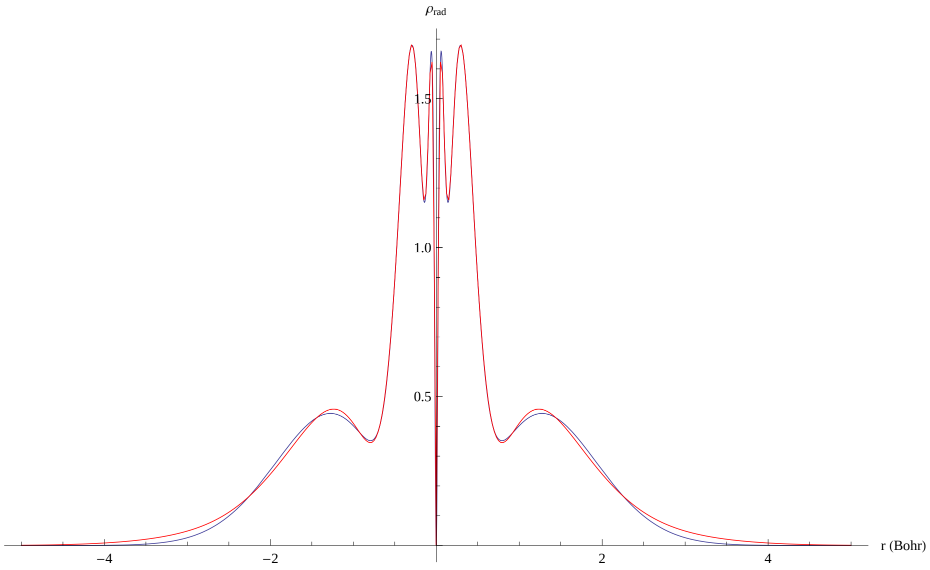


(b) Fitted (Blue) vs actual (Red) radial density.

Figure B.1.17: (a) Fit of the radial electron density of the Cl atom and (b) comparison of the fit and the actual radial electron density.



(a) Fitted radial density.

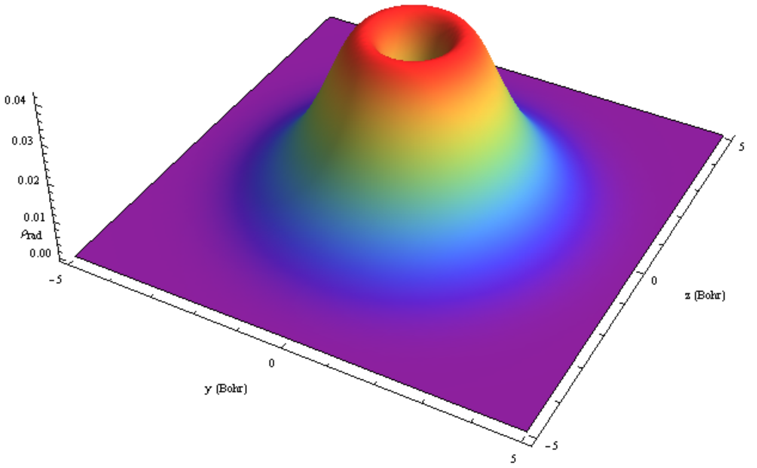


(b) Fitted (Blue) vs actual (Red) radial density.

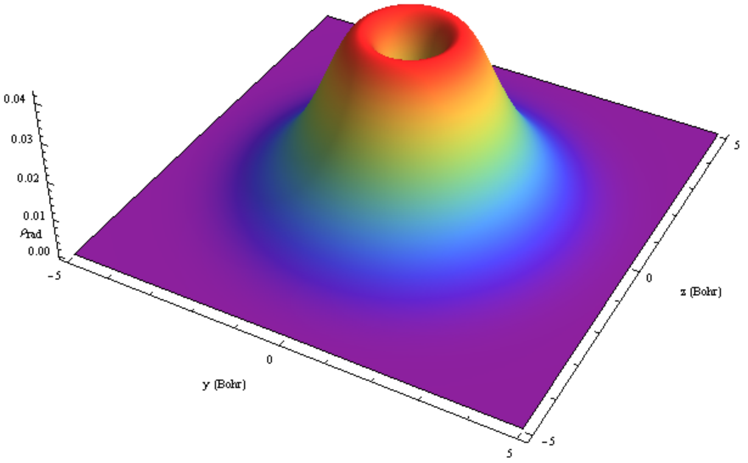
Figure B.1.18: (a) Fit of the radial electron density of the Ar atom and (b) comparison of the fit and the actual radial electron density.

B.2 Individual Atom Densities in 2 Dimensions

B.2.1 Period One Atoms (H-He)

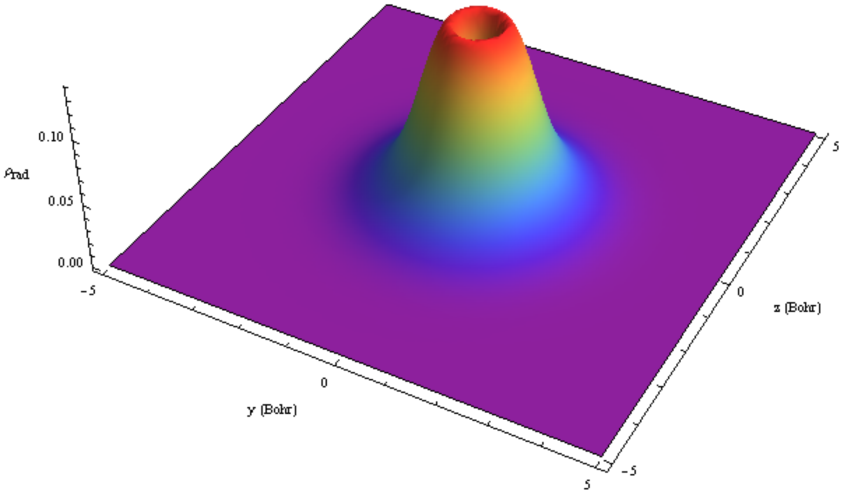


(a) Fitted radial electron density.

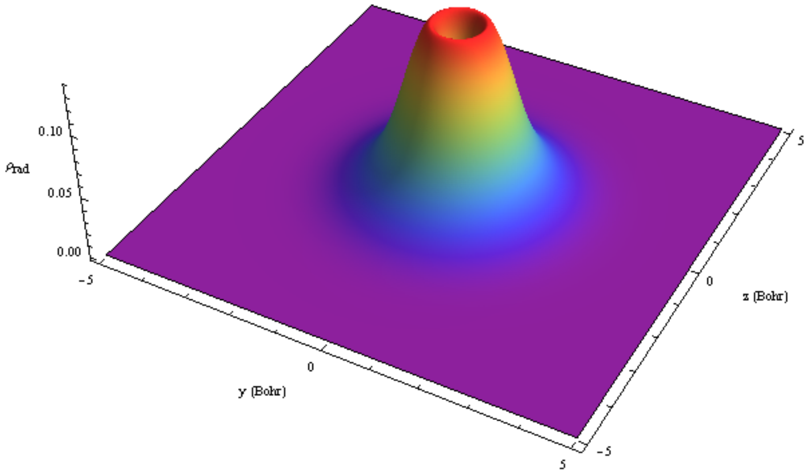


(b) Actual radial electron density.

Figure B.2.1: The fitted (a) and actual (b) radial electron density of H in the yz-plane.



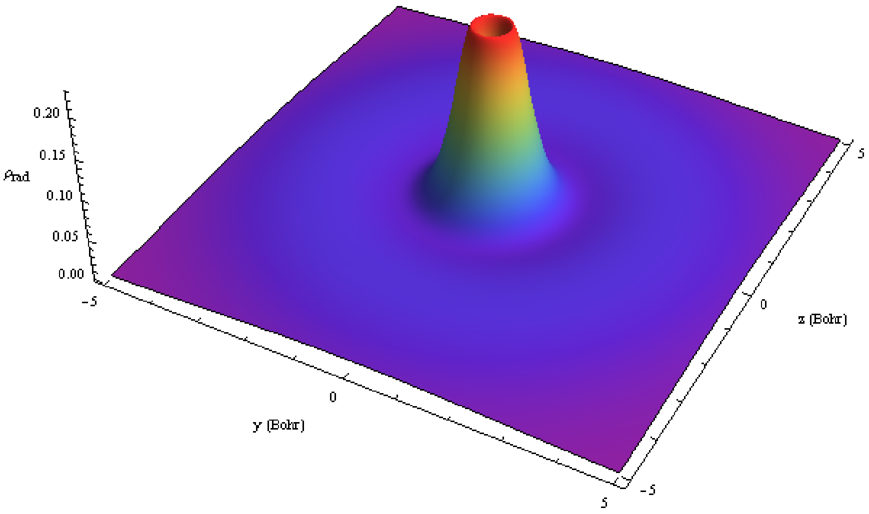
(a) Fitted radial electron density.



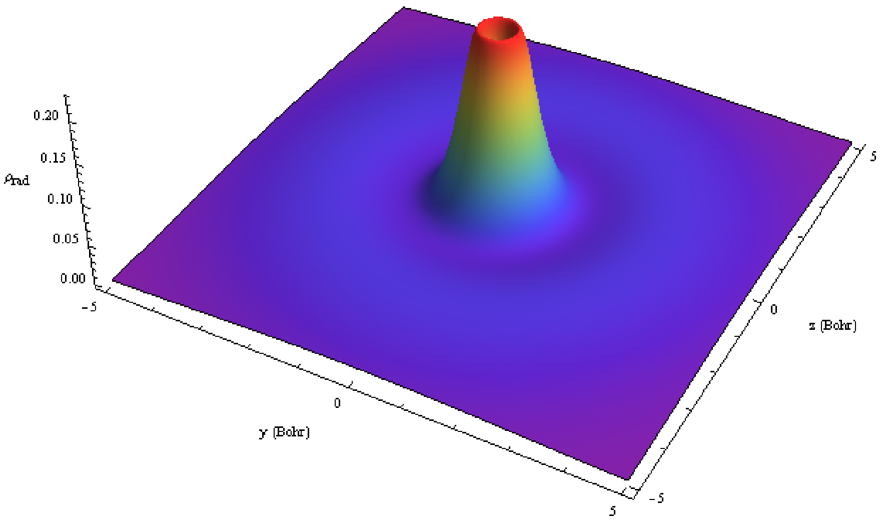
(b) Actual radial electron density.

Figure B.2.2: The fitted (a) and actual (b) radial electron density of He in the yz-plane.

B.2.2 Period Two Atoms (Li-Ne)

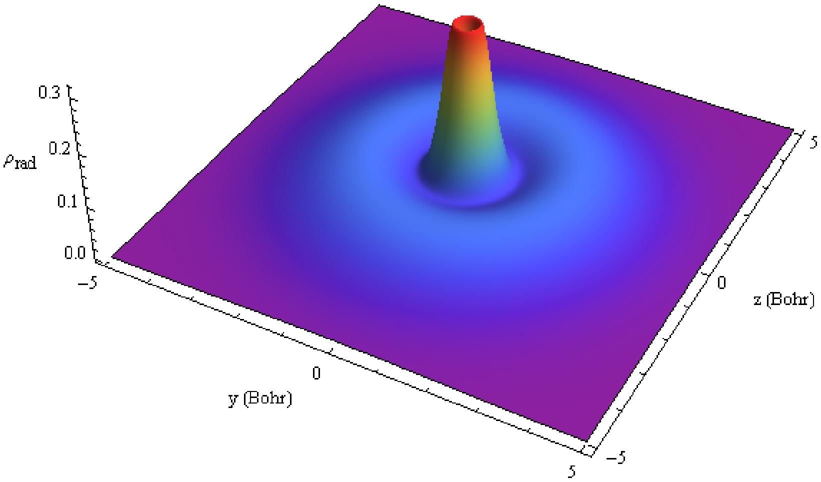


(a) Fitted radial electron density.

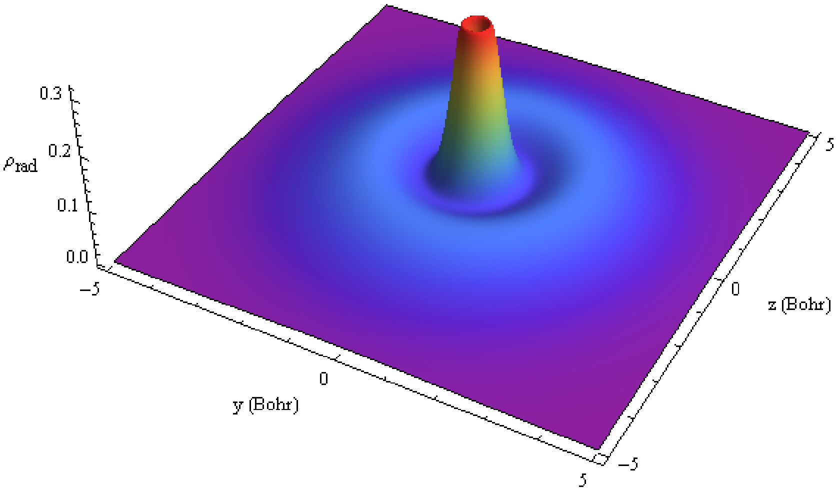


(b) Actual radial electron density.

Figure B.2.3: The fitted (a) and actual (b) radial electron density of Li in the yz-plane.

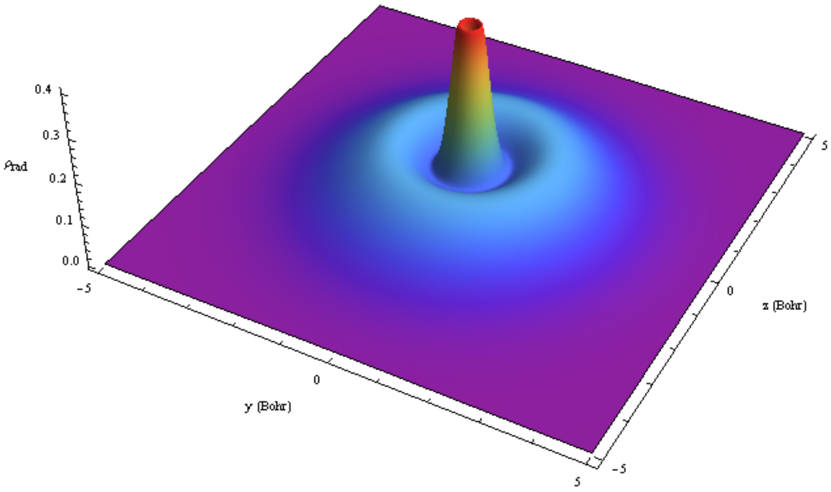


(a) Fitted radial electron density.

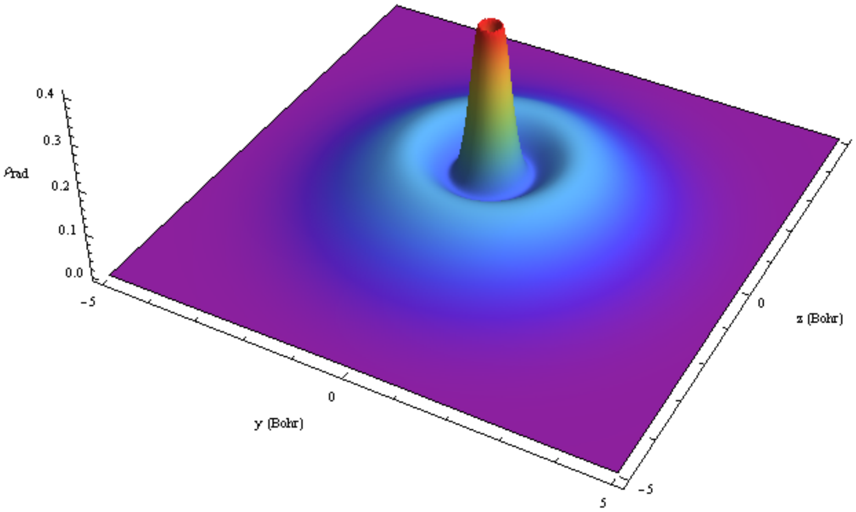


(b) Actual radial electron density.

Figure B.2.4: The fitted (a) and actual (b) radial electron density of Be in the yz-plane.

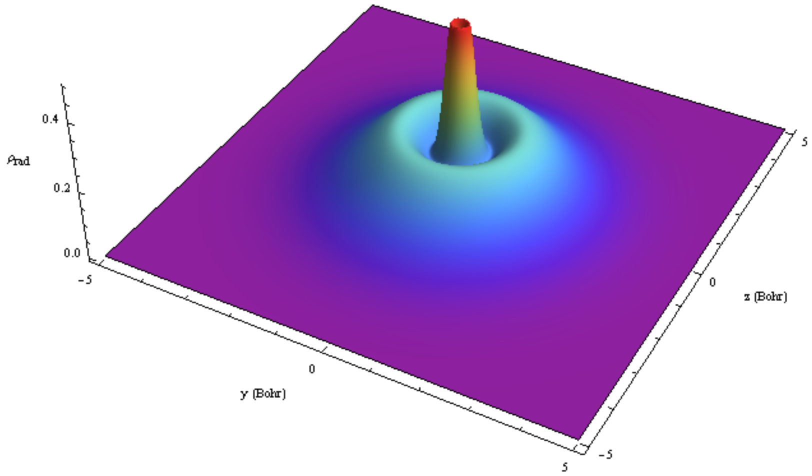


(a) Fitted radial electron density.

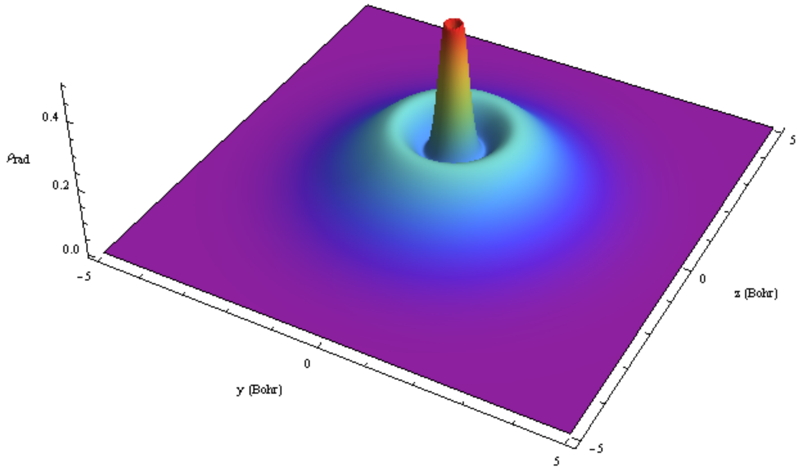


(b) Actual radial electron density.

Figure B.2.5: The fitted (a) and actual (b) radial electron density of B in the yz-plane.

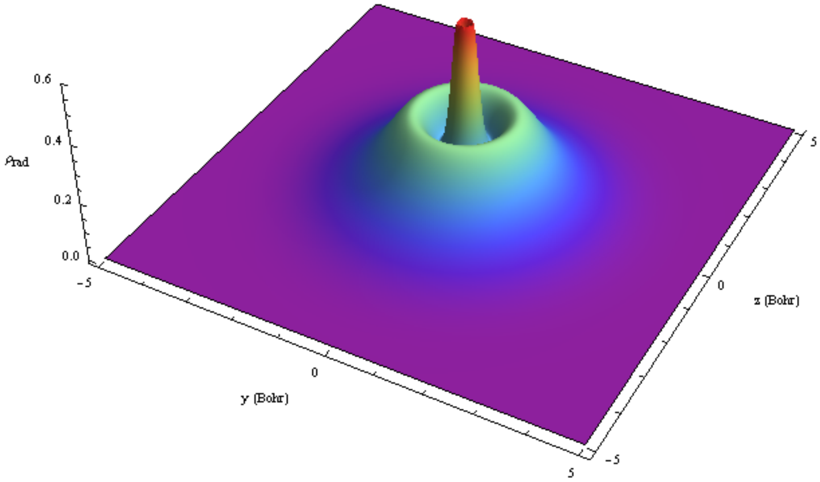


(a) Fitted radial electron density.

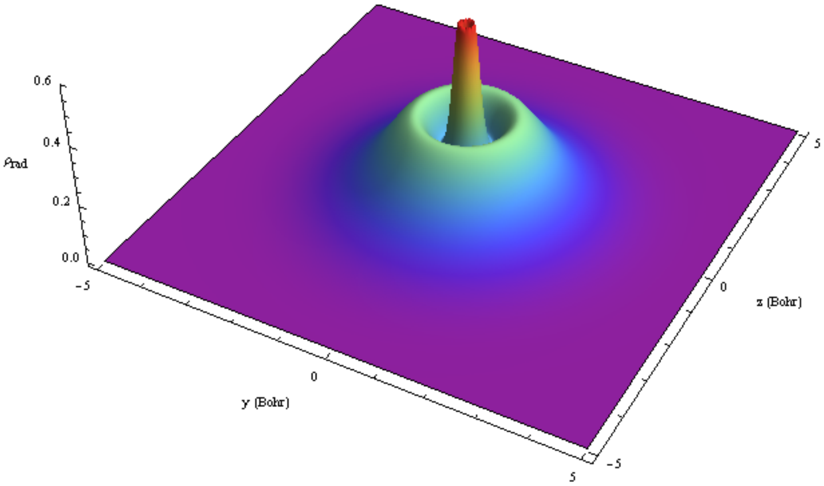


(b) Actual radial electron density.

Figure B.2.6: The fitted (a) and actual (b) radial electron density of C in the yz-plane.

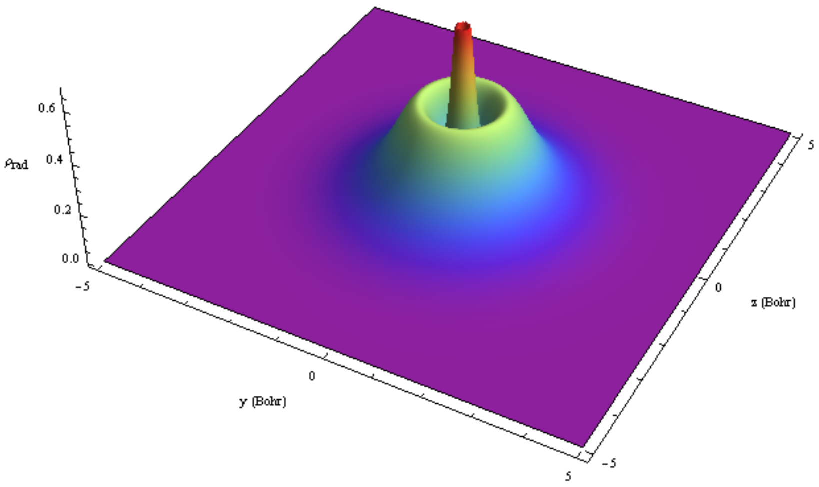


(a) Fitted radial electron density.

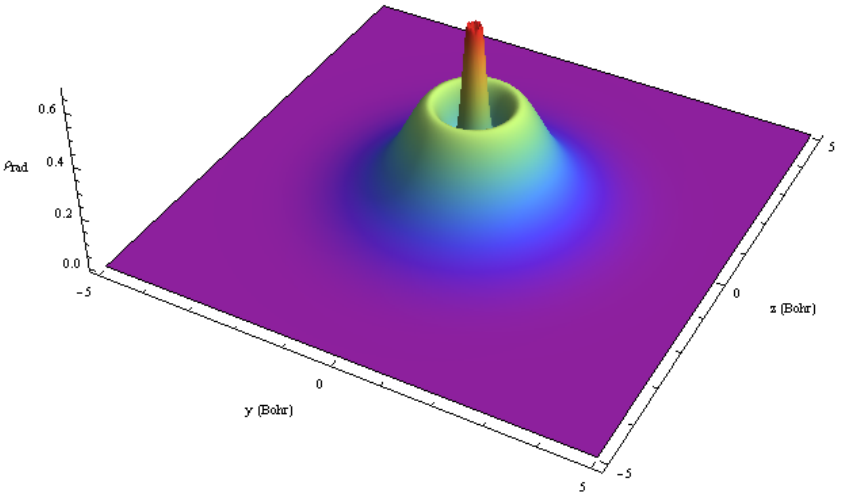


(b) Actual radial electron density.

Figure B.2.7: The fitted (a) and actual (b) radial electron density of N in the yz-plane.

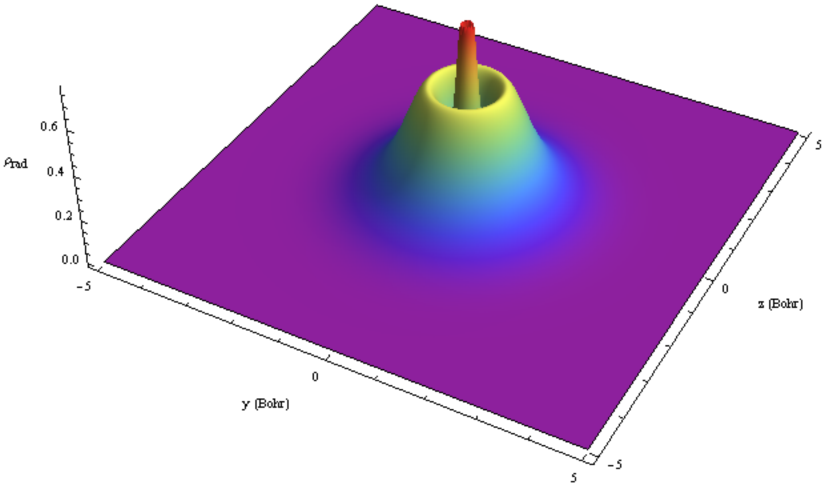


(a) Fitted radial electron density.

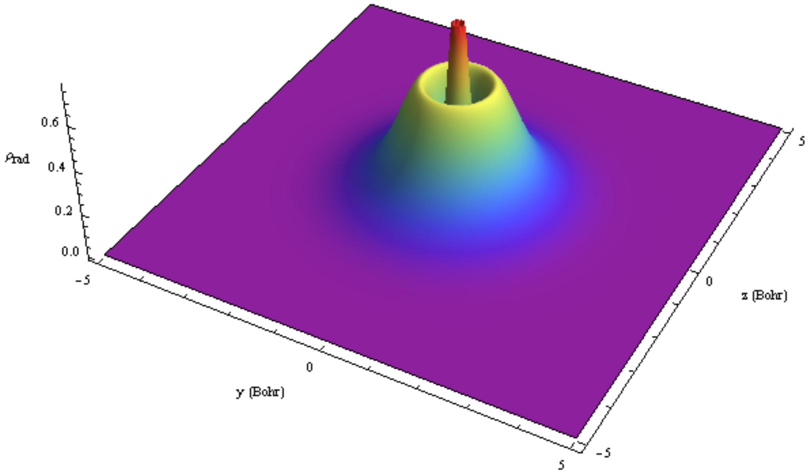


(b) Actual radial electron density.

Figure B.2.8: The fitted (a) and actual (b) radial electron density of O in the yz-plane.

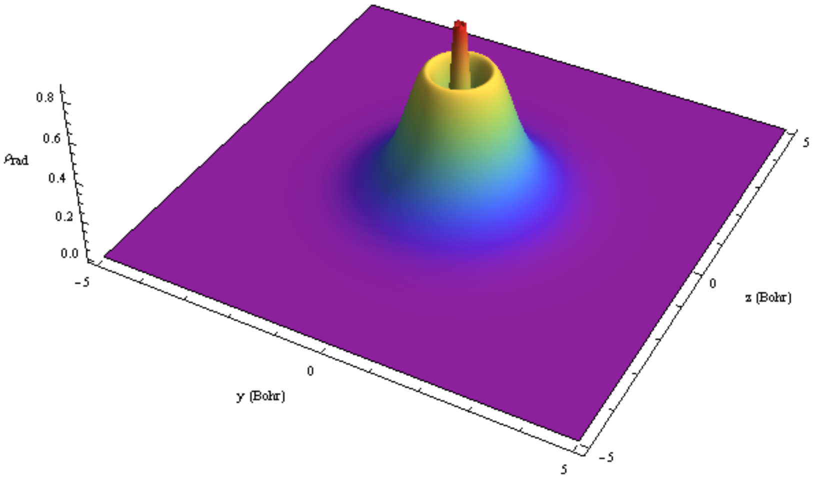


(a) Fitted radial electron density.

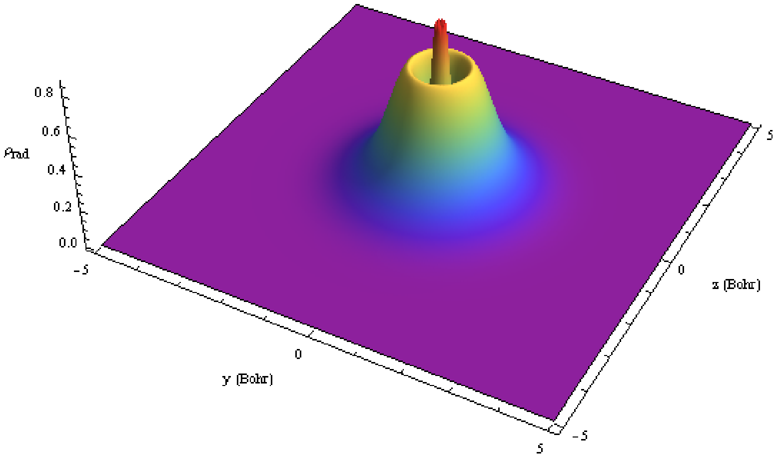


(b) Actual radial electron density.

Figure B.2.9: The fitted (a) and actual (b) radial electron density of F in the yz-plane.



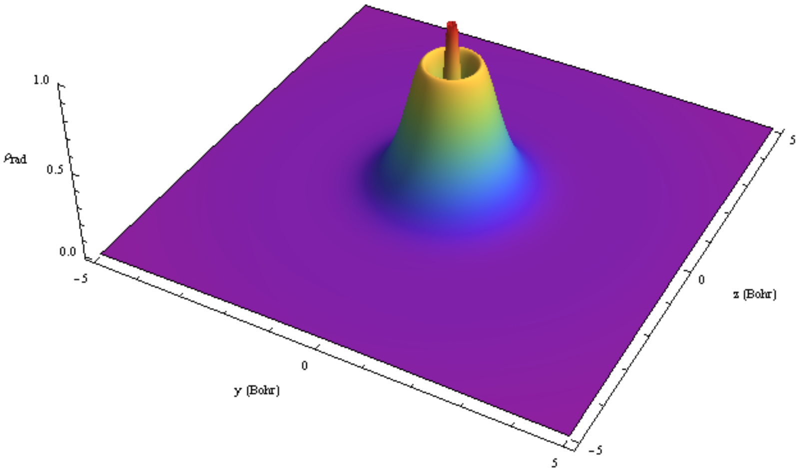
(a) Fitted radial electron density.



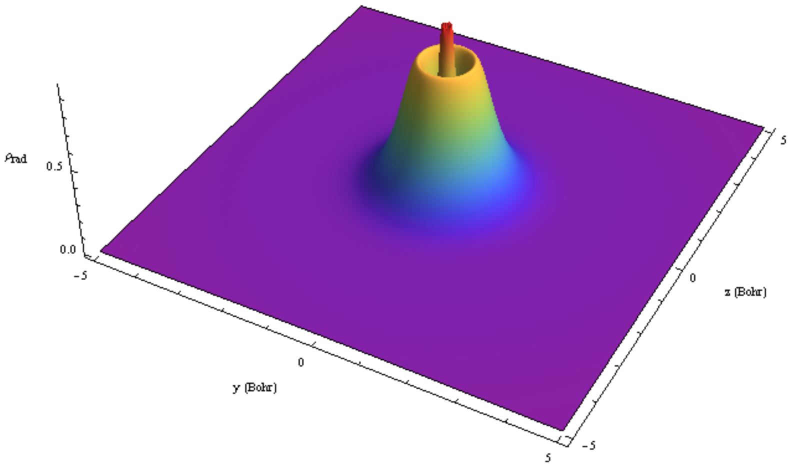
(b) Actual radial electron density.

Figure B.2.10: The fitted (a) and actual (b) radial electron density of Ne in the yz-plane.

B.2.3 Period Three Atoms (Na-Ar)

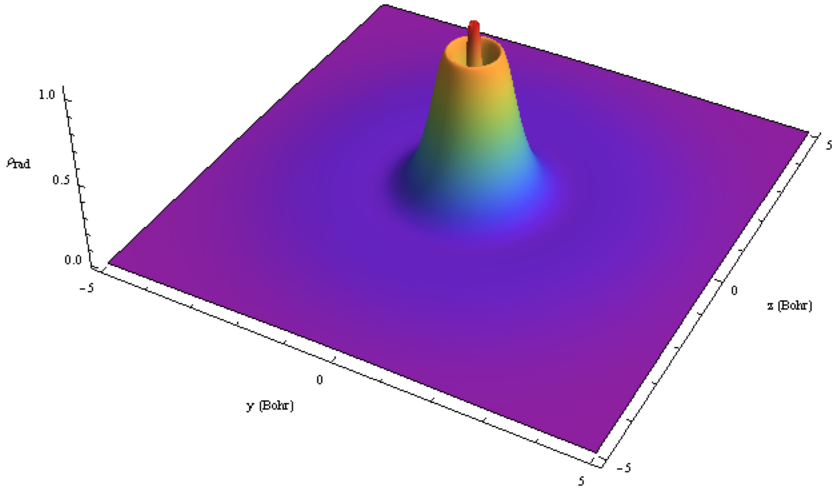


(a) Fitted radial electron density.

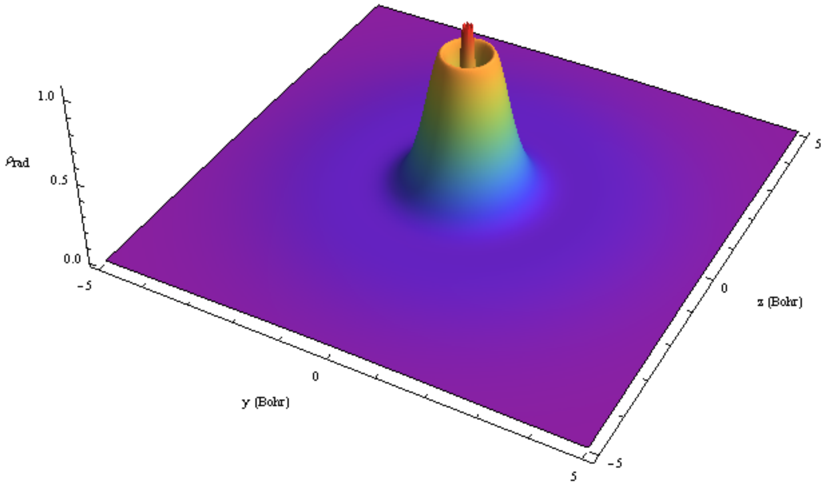


(b) Actual radial electron density.

Figure B.2.11: The fitted (a) and actual (b) radial electron density of Na in the yz-plane.

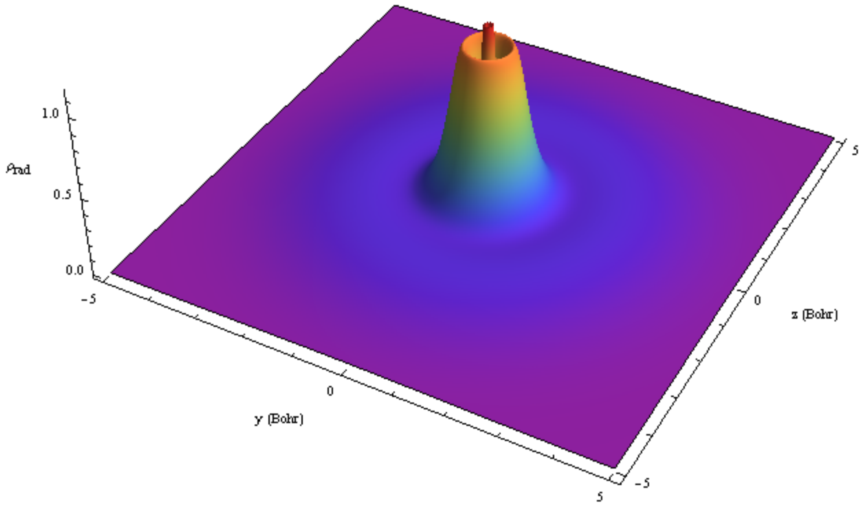


(a) Fitted radial electron density.

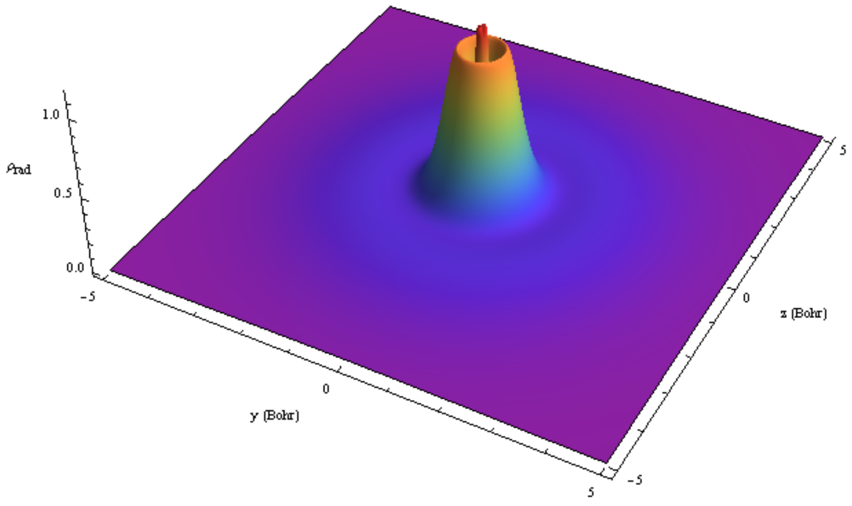


(b) Actual radial electron density.

Figure B.2.12: The fitted (a) and actual (b) radial electron density of Mg in the yz-plane.

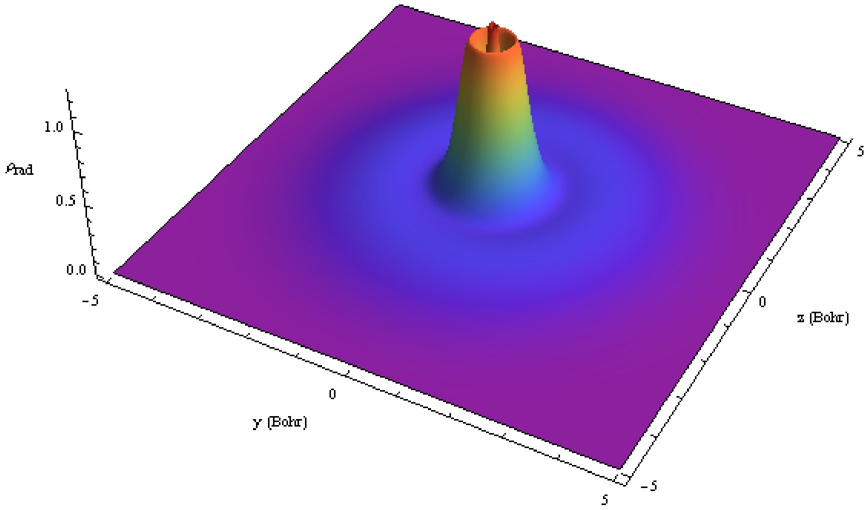


(a) Fitted radial electron density.

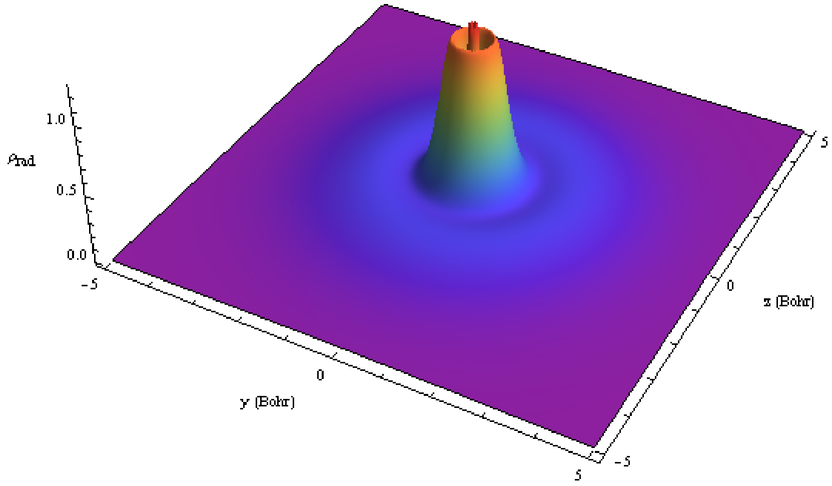


(b) Actual radial electron density.

Figure B.2.13: The fitted (a) and actual (b) radial electron density of Al in the yz-plane.

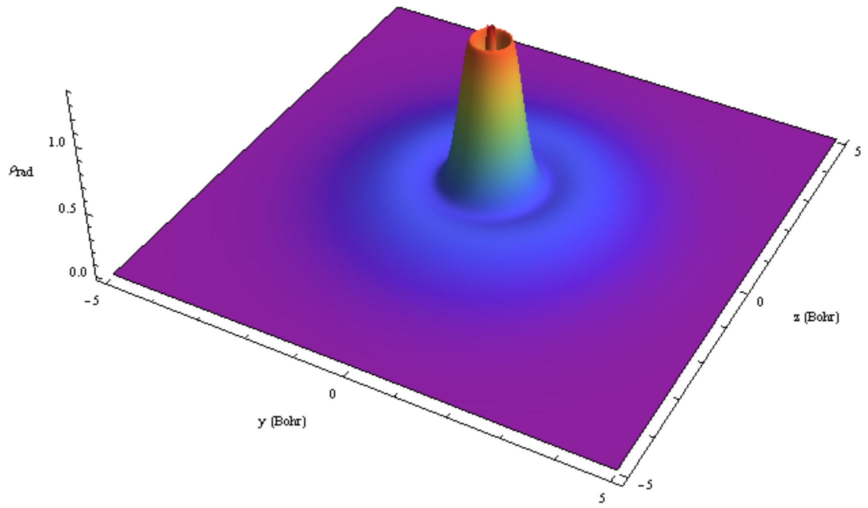


(a) Fitted radial electron density.

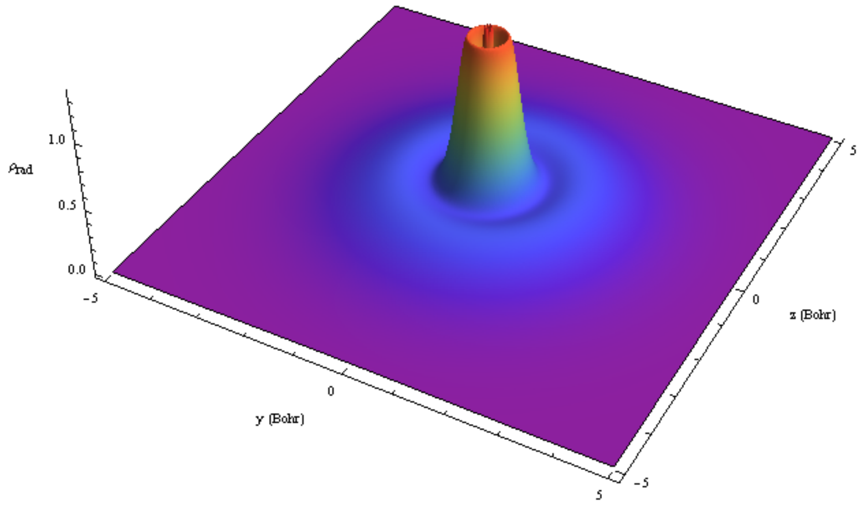


(b) Actual radial electron density.

Figure B.2.14: The fitted (a) and actual (b) radial electron density of Si in the yz-plane.

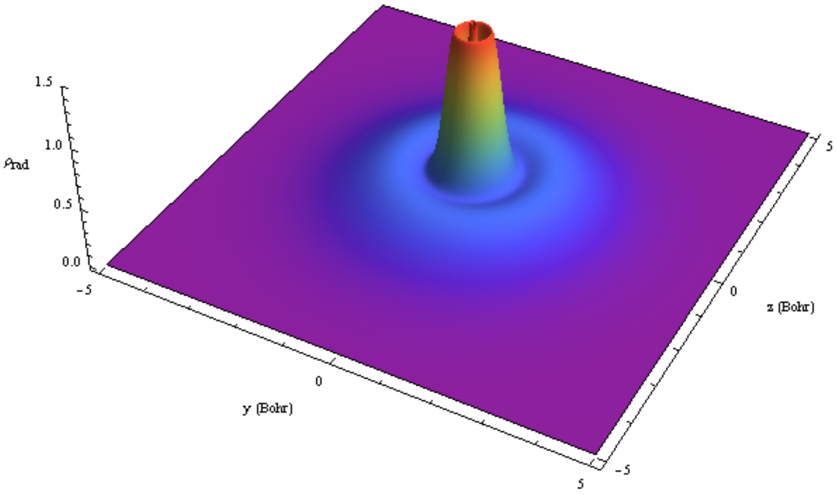


(a) Fitted radial electron density.

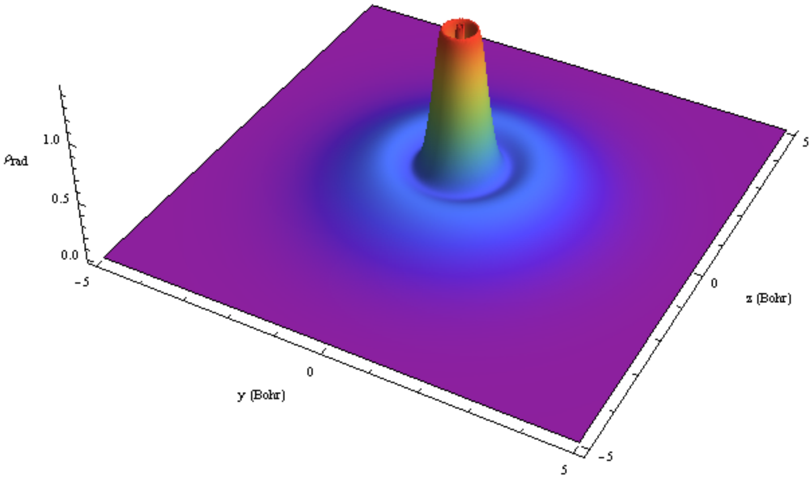


(b) Actual radial electron density.

Figure B.2.15: The fitted (a) and actual (b) radial electron density of P in the yz-plane.

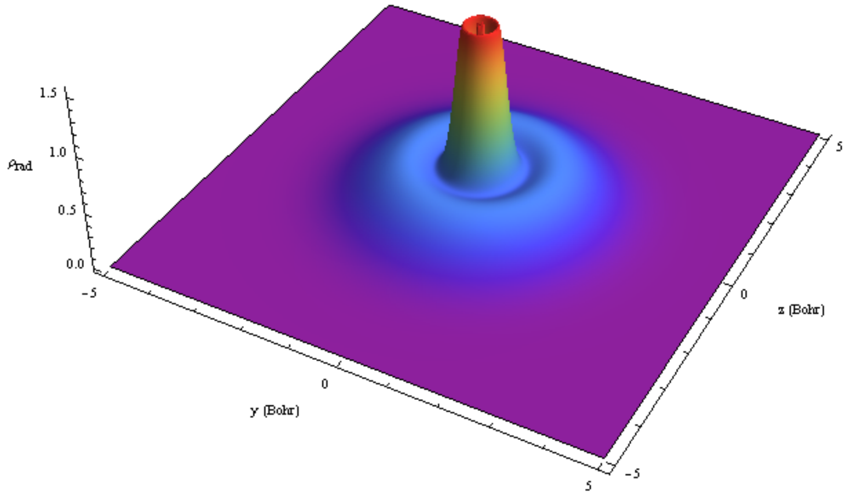


(a) Fitted radial electron density.

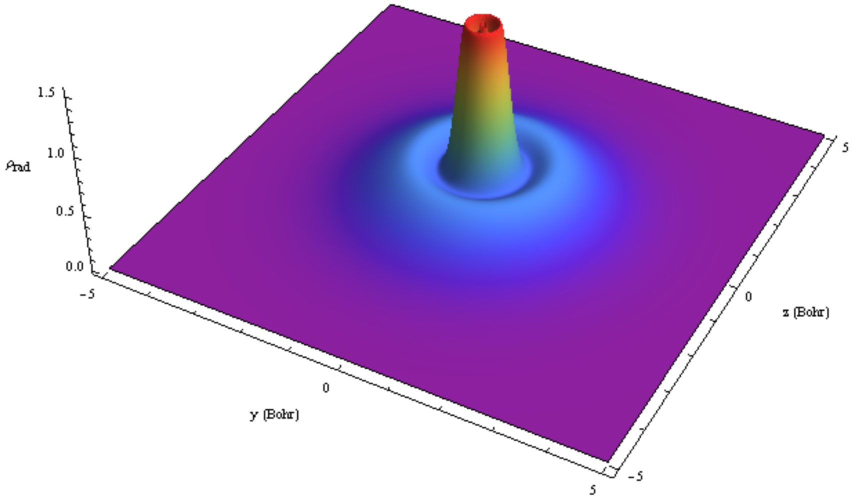


(b) Actual radial electron density.

Figure B.2.16: The fitted (a) and actual (b) radial electron density of S in the yz-plane.

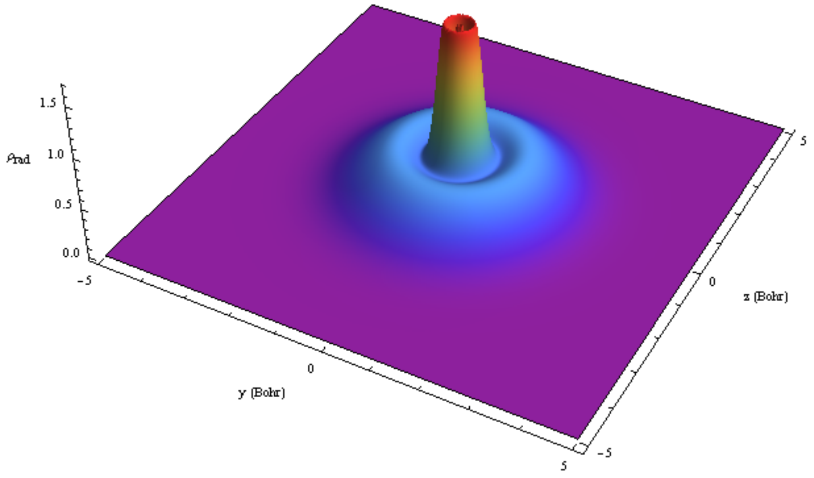


(a) Fitted radial electron density.

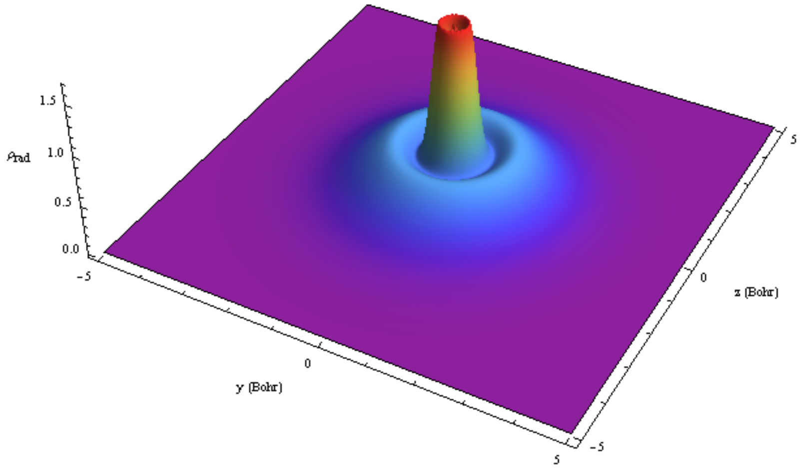


(b) Actual radial electron density.

Figure B.2.17: The fitted (a) and actual (b) radial electron density of Cl in the yz-plane.



(a) Fitted radial electron density.

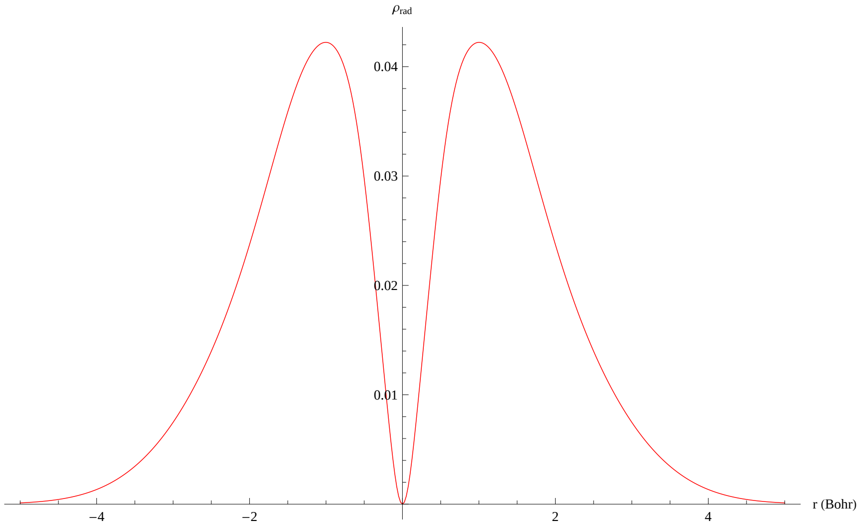


(b) Actual radial electron density.

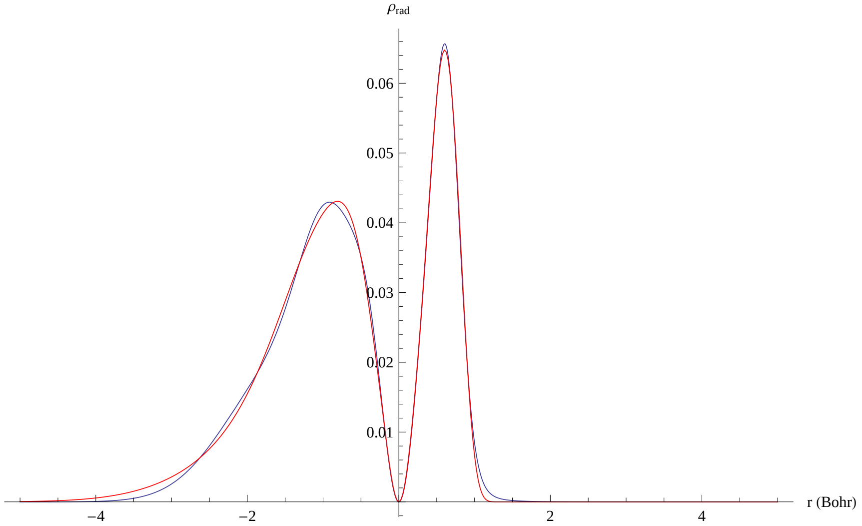
Figure B.2.18: The fitted (a) and actual (b) radial electron density of Ar in the yz-plane.

B.3 AIM Densities

B.3.1 H_2

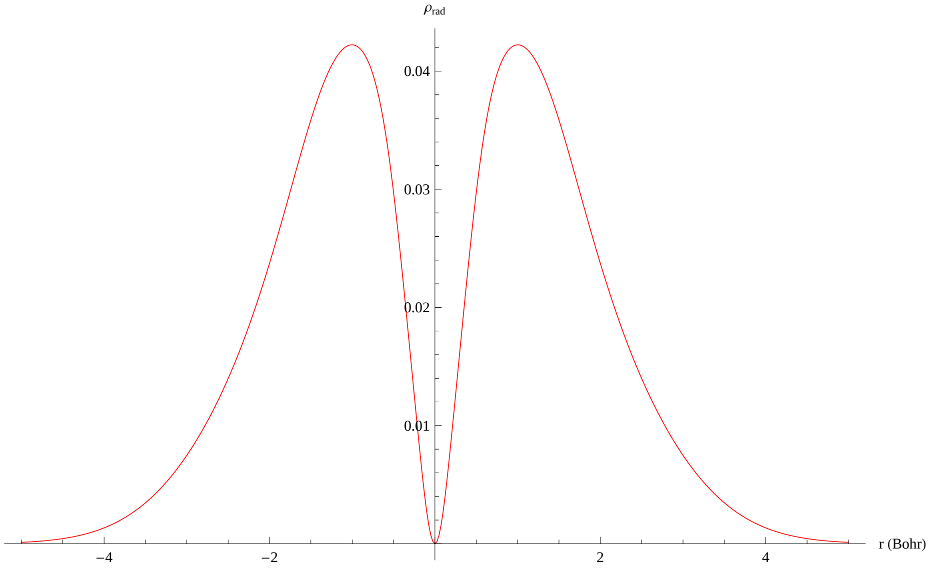


(a) Individual H atom radial electron density.

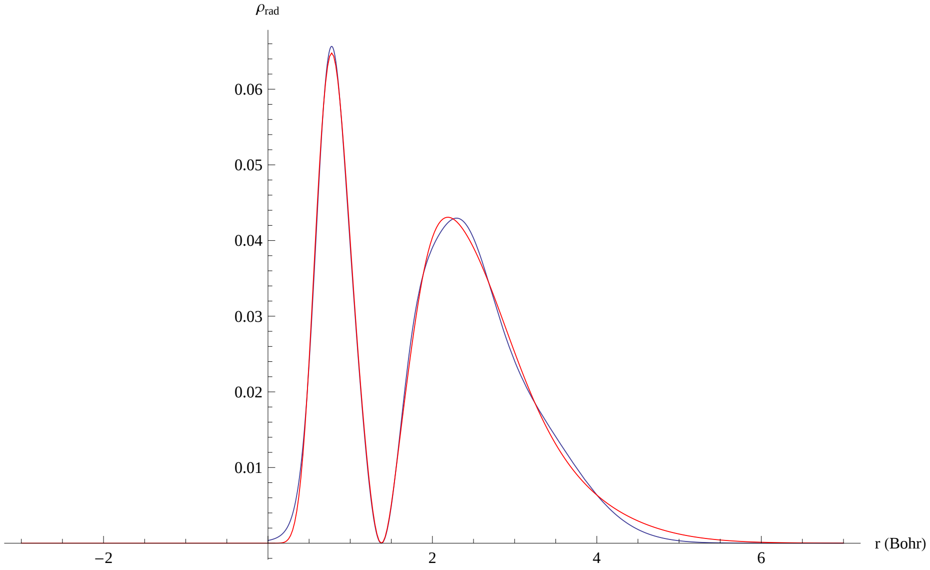


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.3.1: Radial electron density of the H atom (a) and a comparison of the fit and the actual radial electron density of H at (0,0,0) in H_2 (b) vs r in bohr.

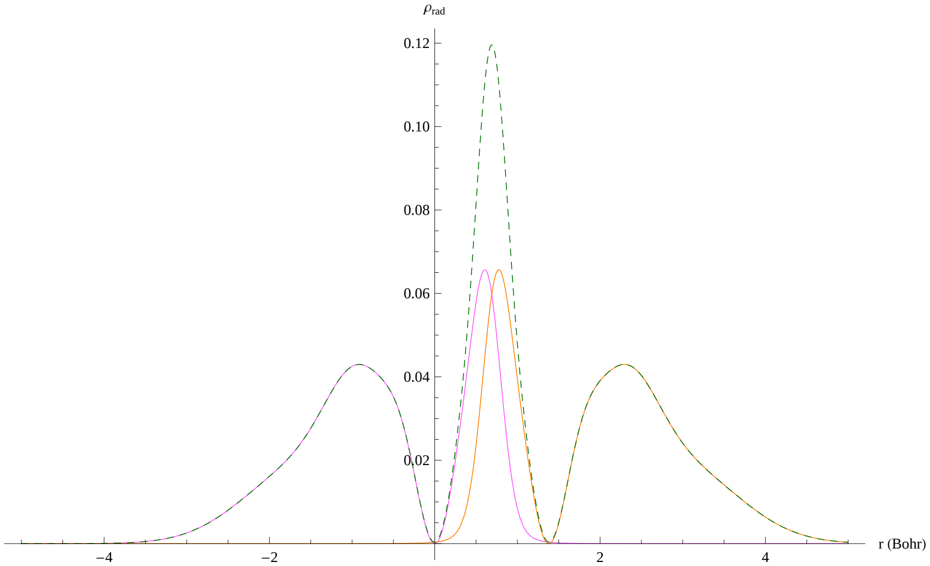


(a) Individual H atom radial electron density.

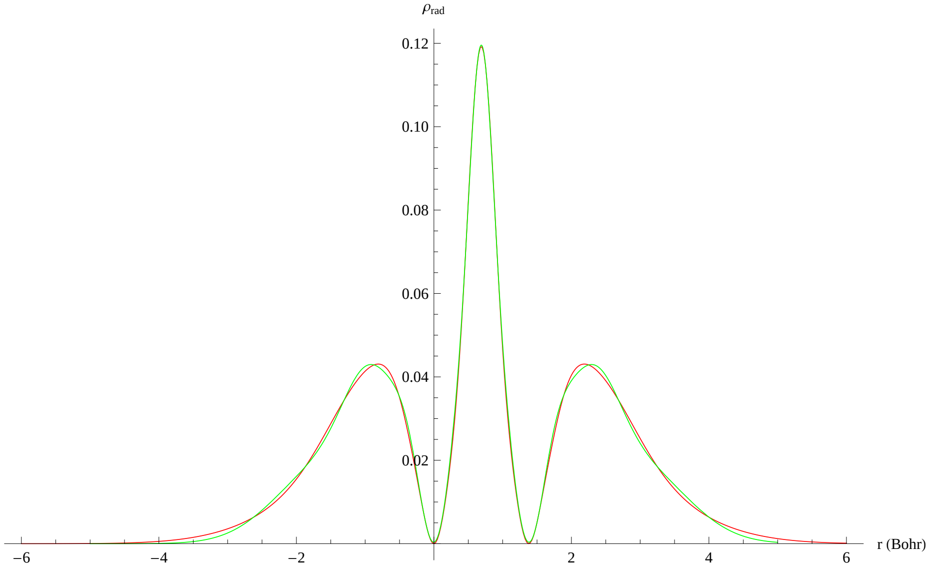


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.3.2: Radial electron density of the H atom (a) and a comparison of the fit and the actual radial electron density of H at (0,0,1.3) in H_2 (b) vs r in bohr.



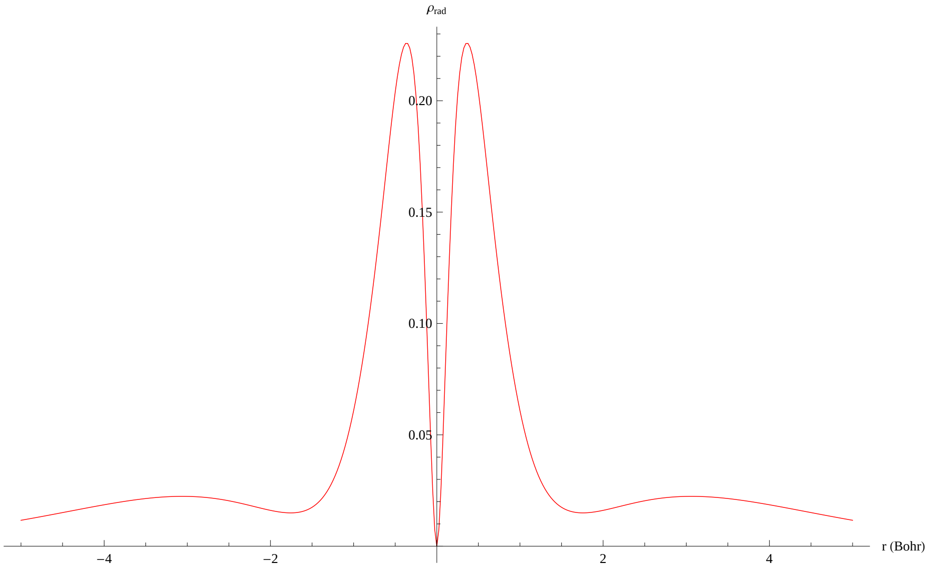
(a) Fitted radial electron densities of H at (0,0,0) (Pink) and H at (0,0,1.3) (Orange) in H_2 (Green).



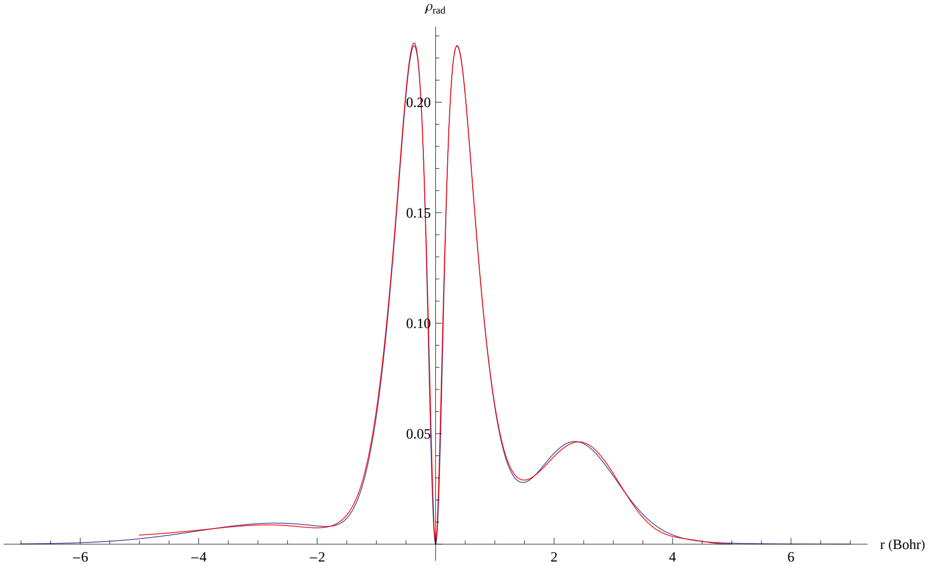
(b) Fitted (Green) vs actual (Red) molecular radial electron density.

Figure B.3.3: A comparison of radial electron density of the fitted AIM and the total molecular radial electron density (a) and a comparison of the fitted and the actual molecular radial electron density (b) of H_2 vs r in bohr.

B.3.2 Li_2

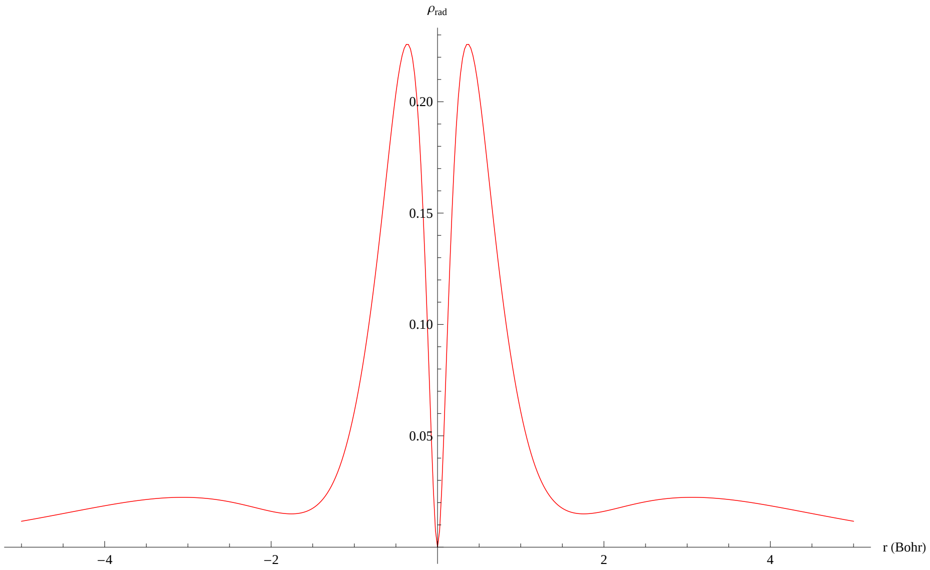


(a) Individual Li atom radial electron density.

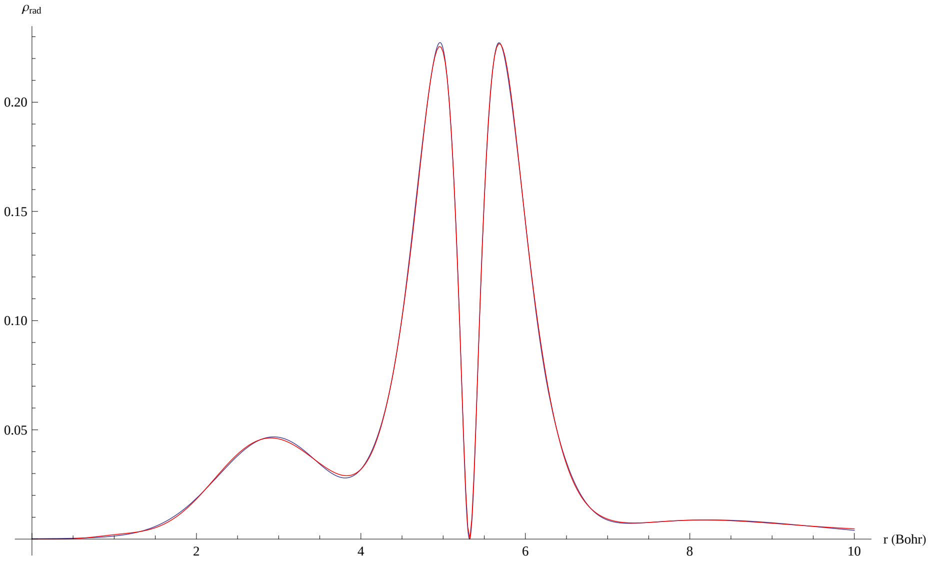


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.3.4: Radial electron density of the Li atom (a) and a comparison of the fit and the actual radial electron density of Li at (0,0,0) in Li_2 (b) vs r in bohr.

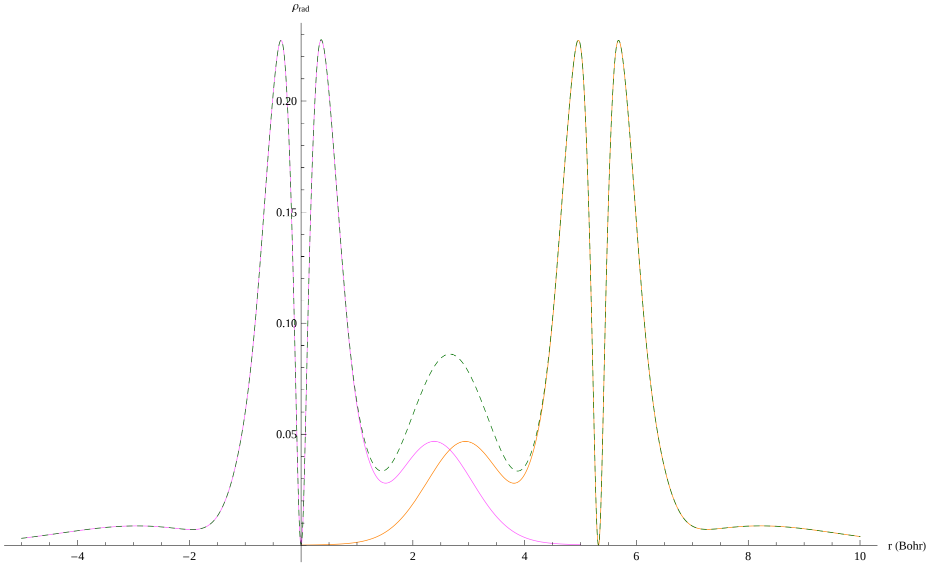


(a) Individual Li atom radial electron density.

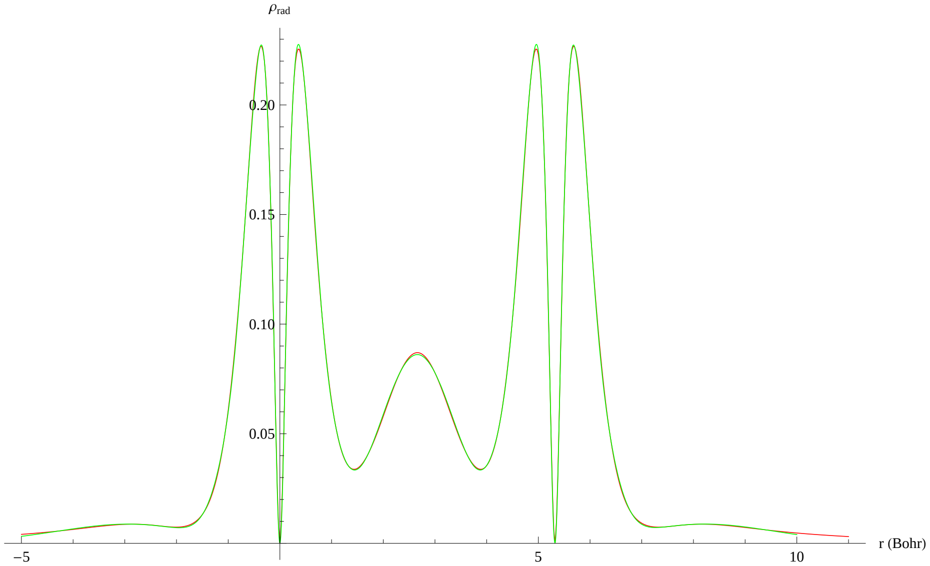


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.3.5: Radial electron density of the Li atom (a) and a comparison of the fit and the actual radial electron density of Li at (0,0,5.3) in Li_2 (b) vs r in bohr.



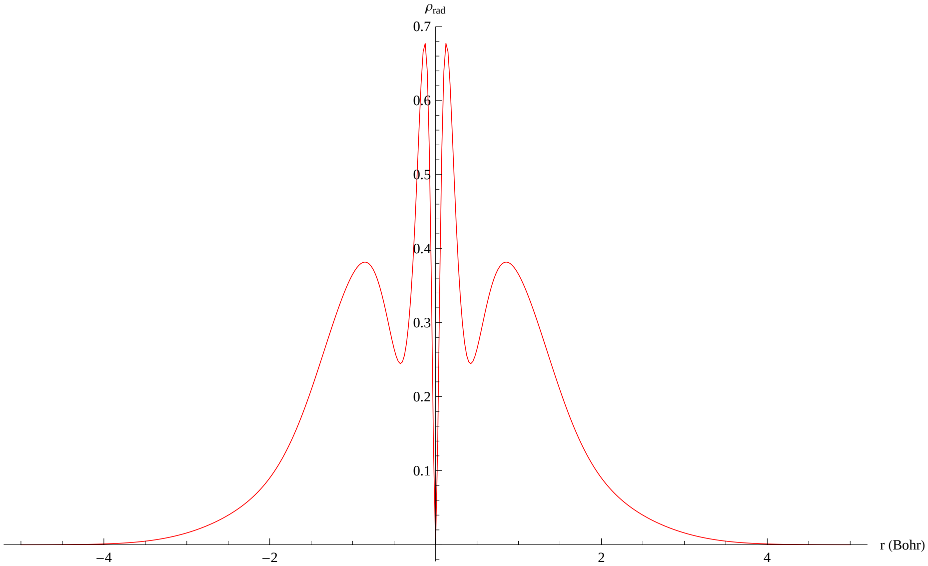
(a) Fitted radial electron densities of Li at (0,0,0) (Pink) and Li at (0,0,5.3) (Orange) in Li_2 (Green).



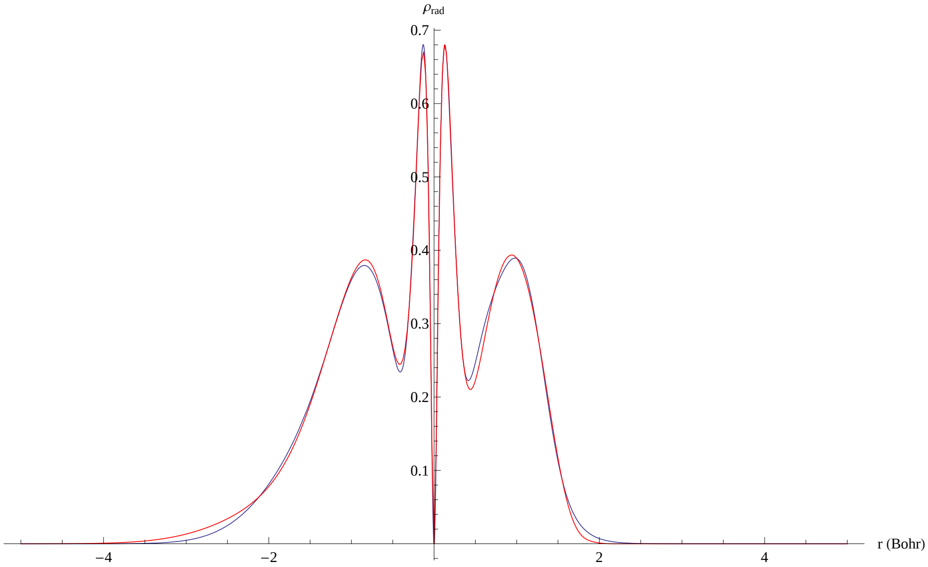
(b) Fitted (Green) vs actual (Red) molecular radial electron density.

Figure B.3.6: A comparison of radial electron density of the fitted AIM and the total molecular radial electron density (a) and a comparison of the fitted and the actual molecular radial electron density (b) of Li_2 vs r in bohr.

B.3.3 O_2

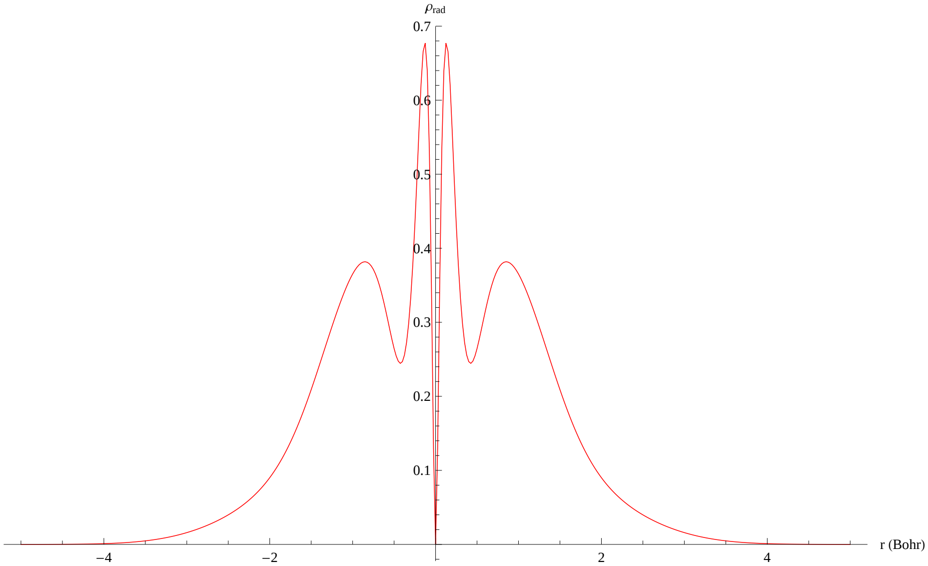


(a) Individual O atom radial electron density.

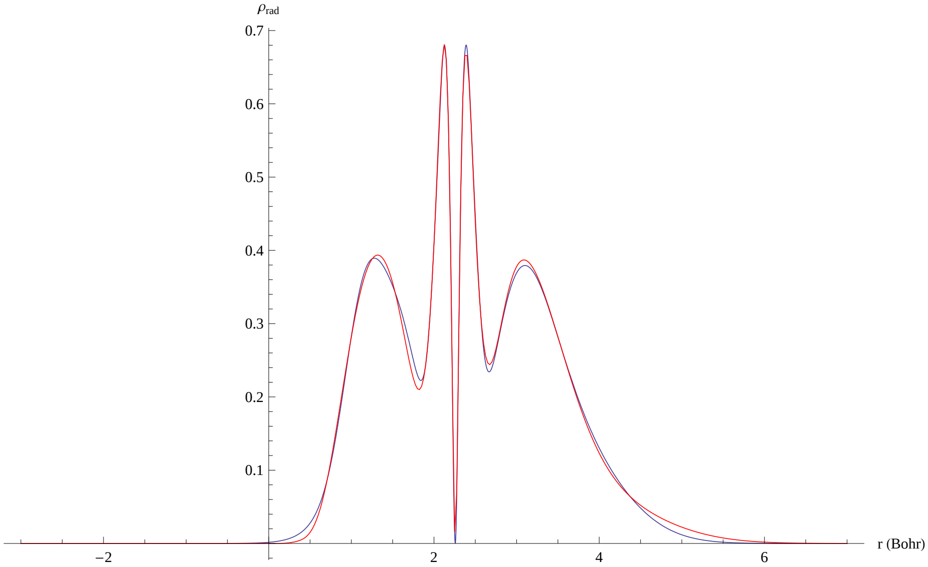


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.3.7: Radial electron density of the O atom (a) and a comparison of the fit and the actual radial electron density of O at (0,0,0) in O_2 (b) vs r in bohr.

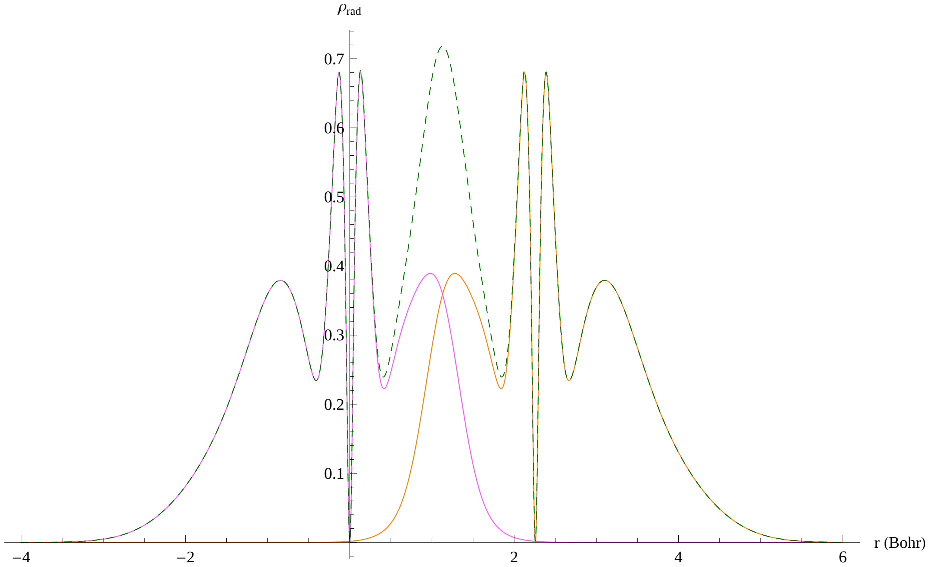


(a) Individual O atom radial electron density.

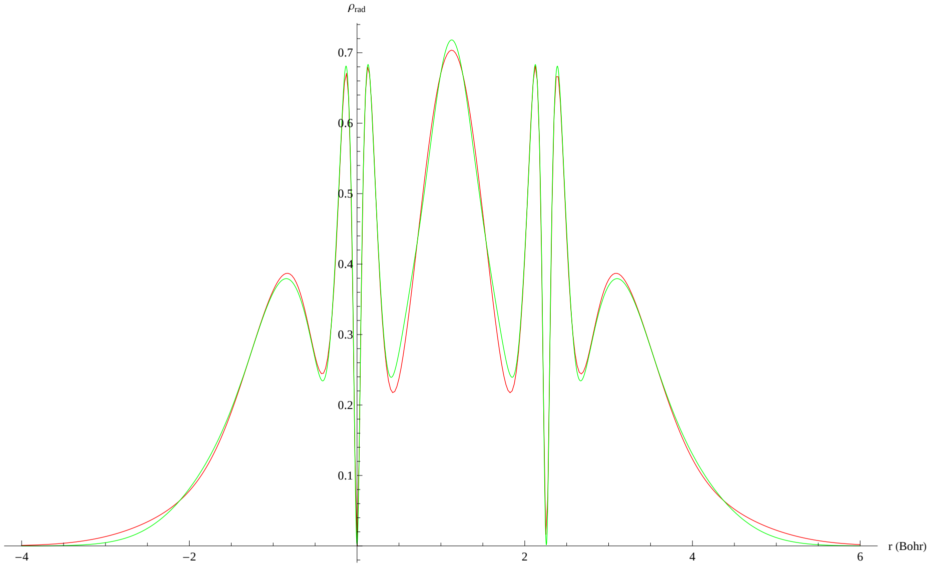


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.3.8: Radial electron density of the O atom (a) and a comparison of the fit and the actual radial electron density of O at (0,0,2.3) in O_2 (b) vs r in bohr.



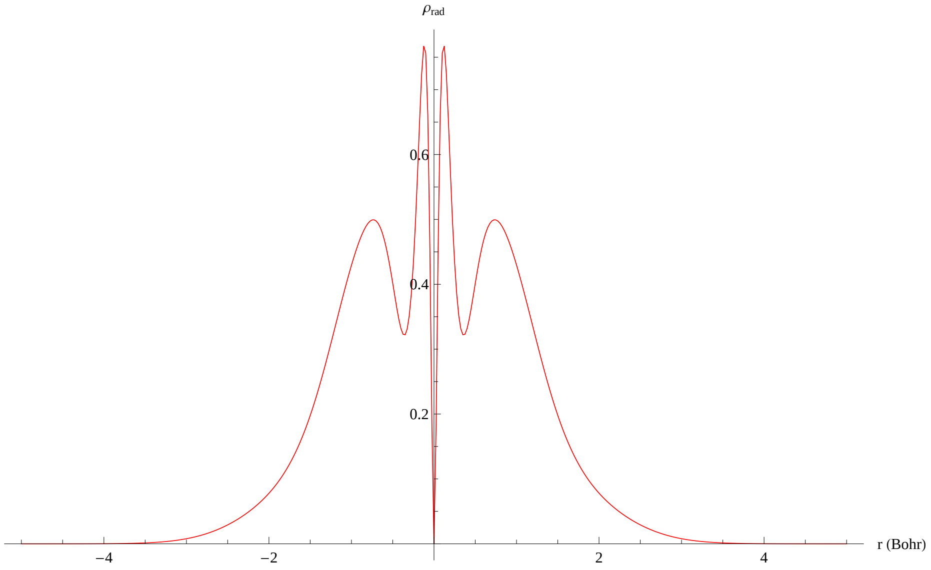
(a) Fitted radial electron densities of O at (0,0,0) (Pink) and O at (0,0,2.3) (Orange) in O_2 (Green).



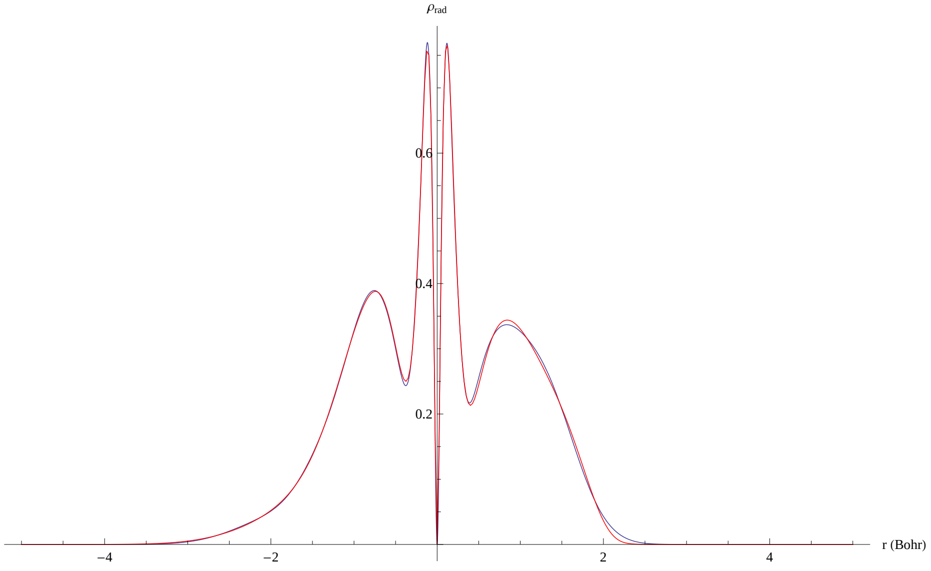
(b) Fitted (Green) vs actual (Red) molecular radial electron density.

Figure B.3.9: A comparison of radial electron density of the fitted AIM and the total molecular radial electron density (a) and a comparison of the fitted and the actual molecular radial electron density (b) of O_2 (b) vs r in bohr.

B.3.4 F_2

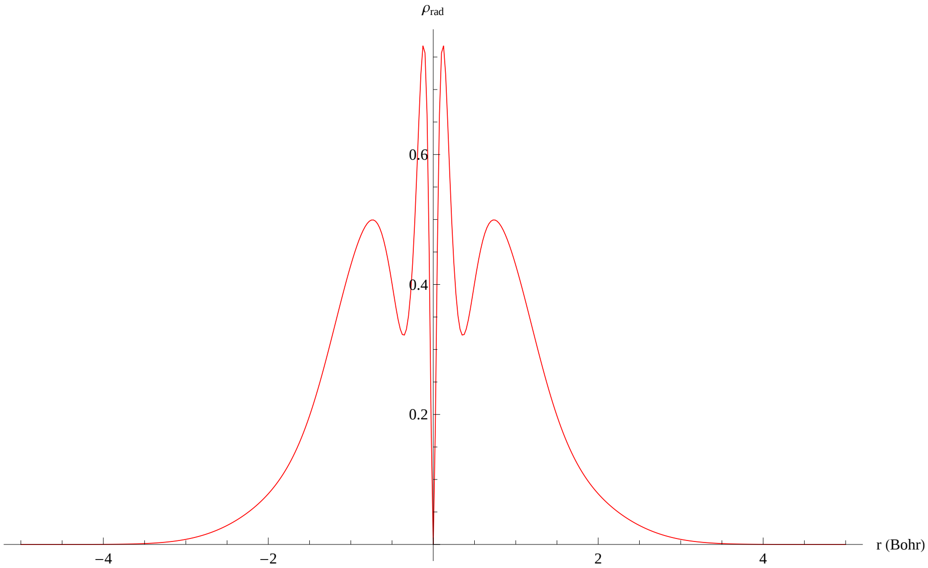


(a) Individual F atom radial electron density.

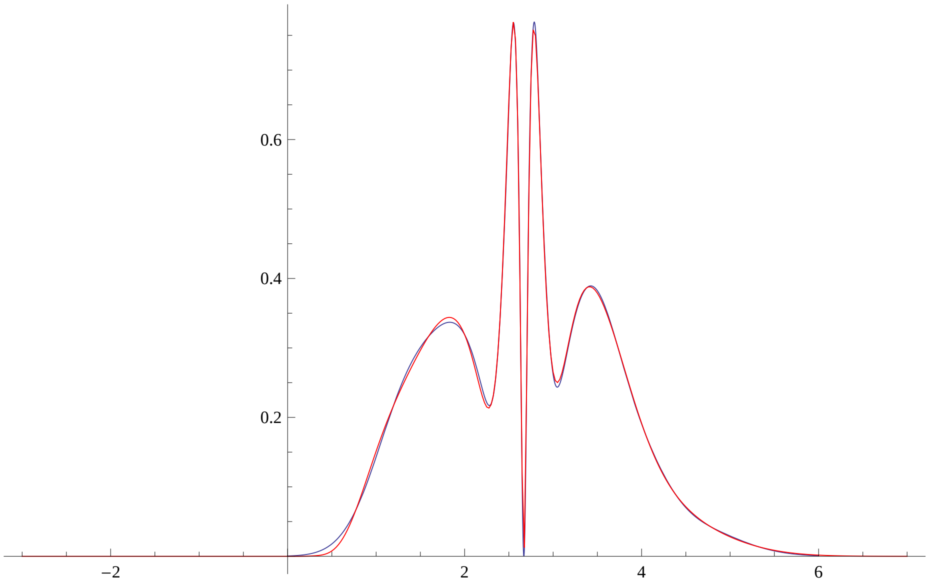


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.3.10: Radial electron density of the F atom (a) and a comparison of the fit and the actual radial electron density of F at (0,0,0) in F_2 (b) vs r in bohr.

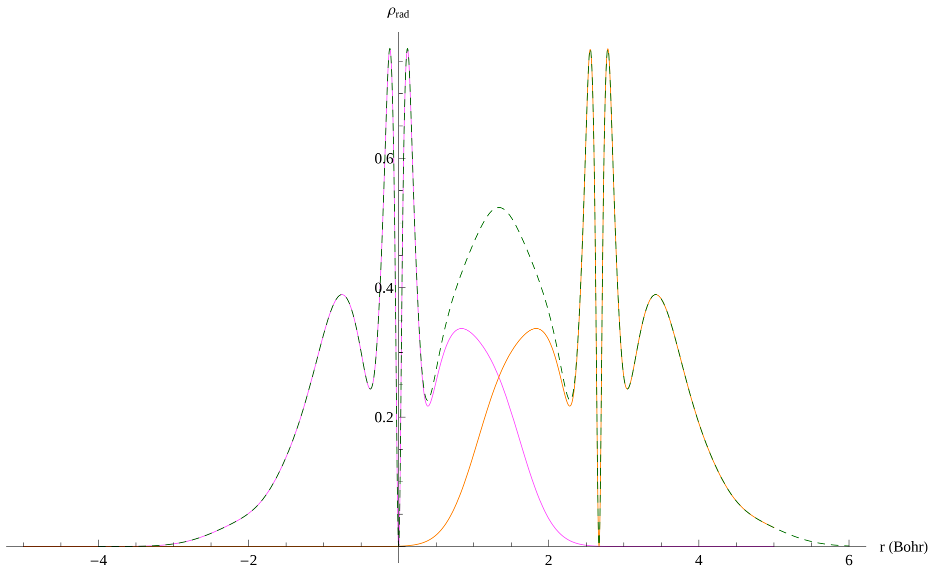


(a) Individual F atom radial electron density.

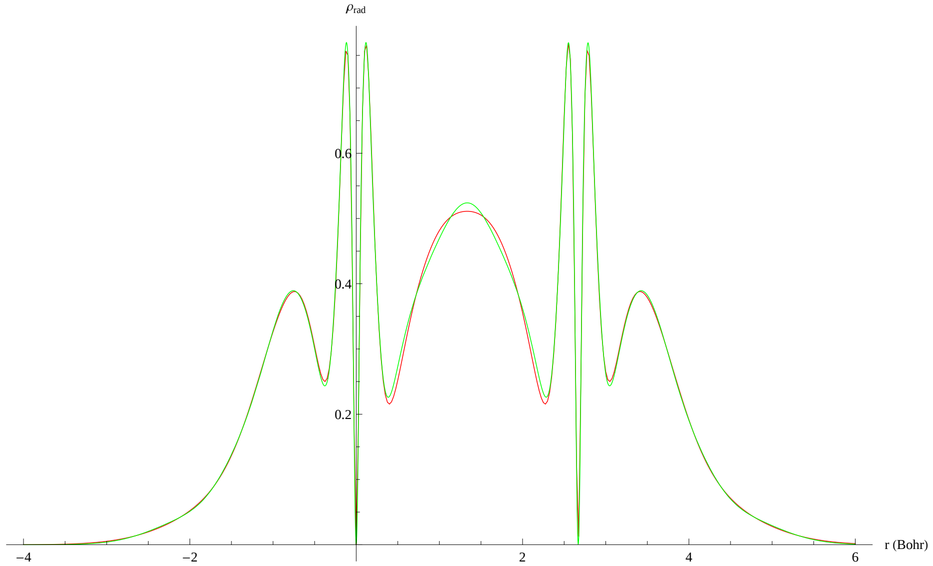


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.3.11: Radial electron density of the F atom (a) and a comparison of the fit and the actual radial electron density of F at (0,0,2.7) in F_2 (b) vs r in bohr.



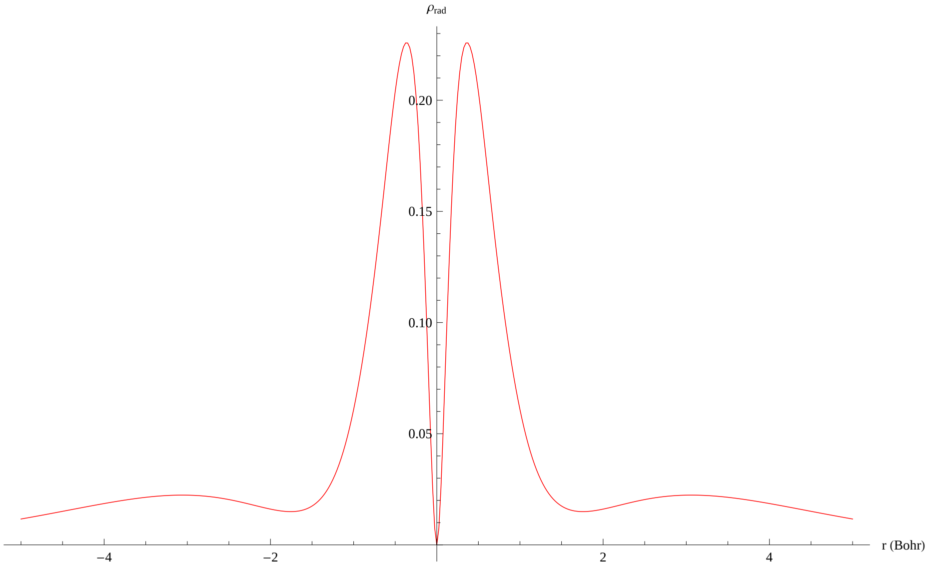
(a) Fitted radial electron densities of F at (0,0,0) (Pink) and F at (0,0,2.7) (Orange) in F_2 (Green).



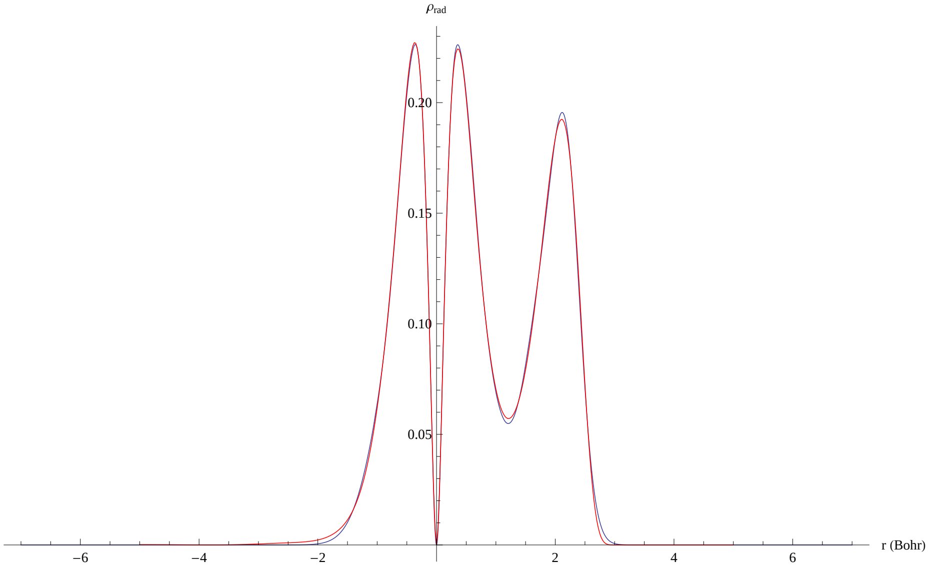
(b) Fitted (Green) vs actual (Red) molecular radial electron density.

Figure B.3.12: A comparison of radial electron density of the fitted AIM and the total molecular radial electron density (a) and a comparison of the fitted and the actual molecular radial electron density (b) of F_2 (b) vs r in bohr.

B.3.5 LiH

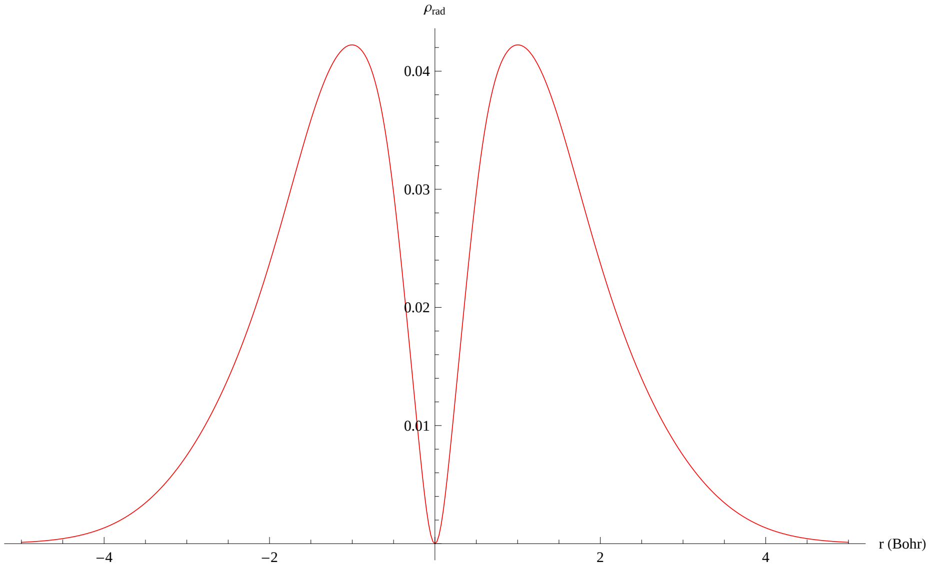


(a) Individual Li atom radial electron density.

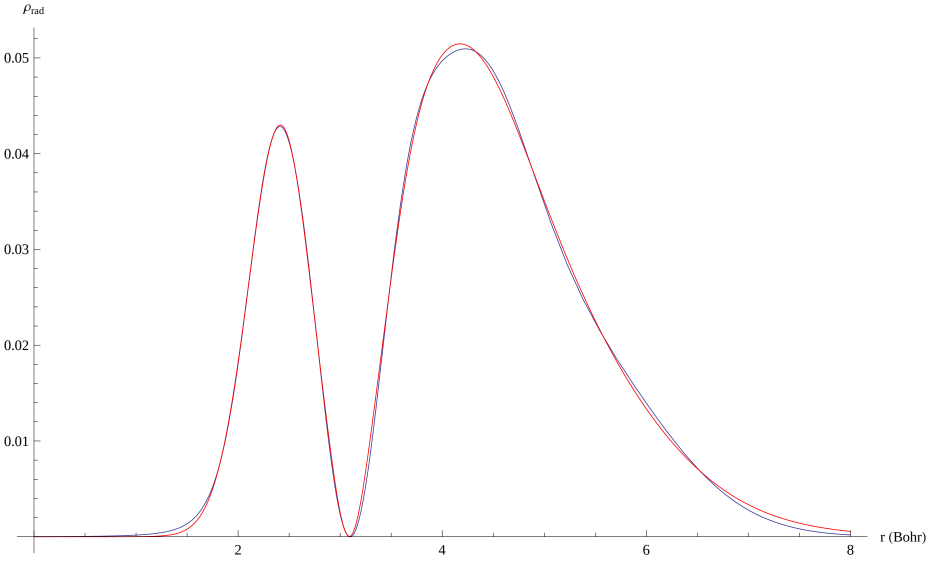


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.3.13: Radial electron density of the Li atom (a) and a comparison of the fit and the actual radial electron density of Li at (0,0,0) in LiH (b) vs r in bohr.

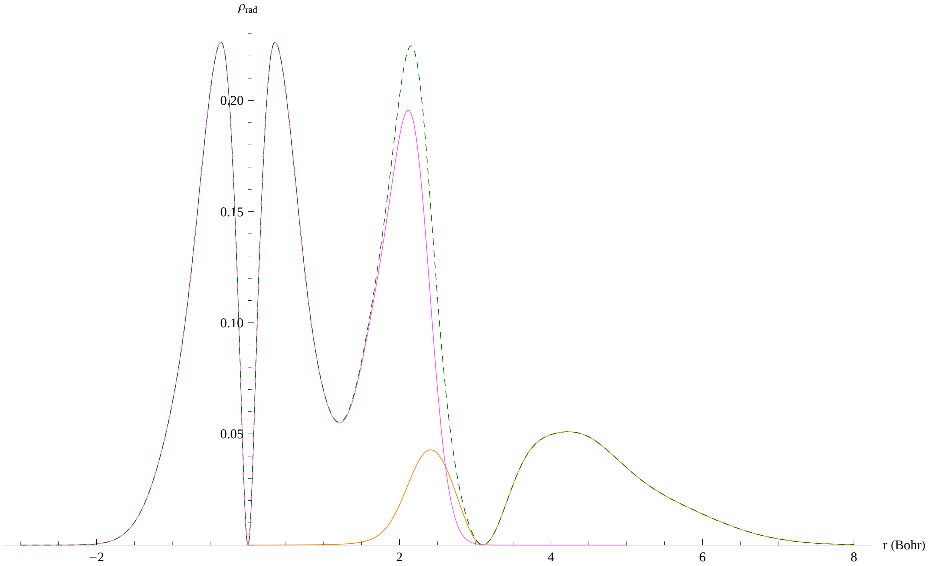


(a) Individual H atom radial electron density.

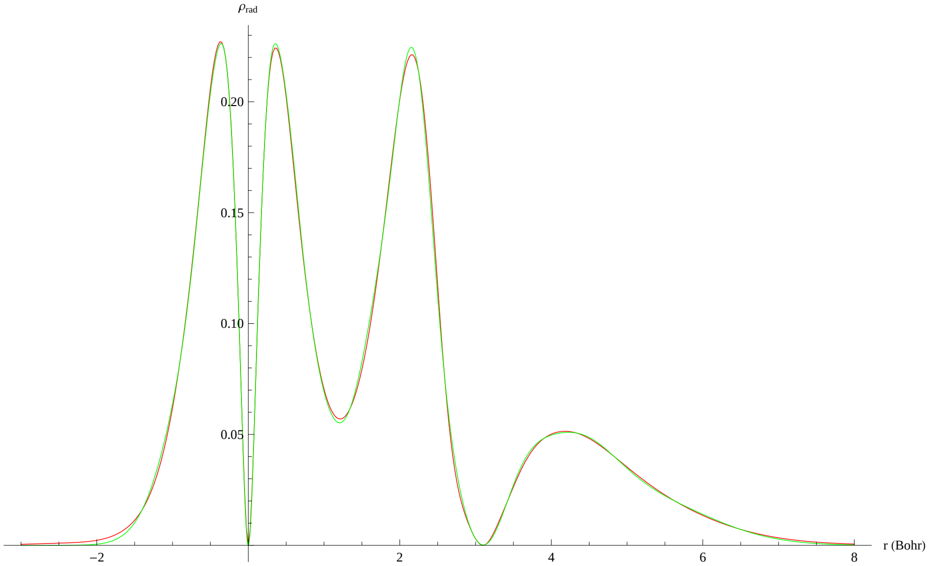


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.3.14: Radial electron density of the H atom (a) and a comparison of the fit and the actual radial electron density of H at (0,0,3.1) in LiH (b) vs r in bohr.



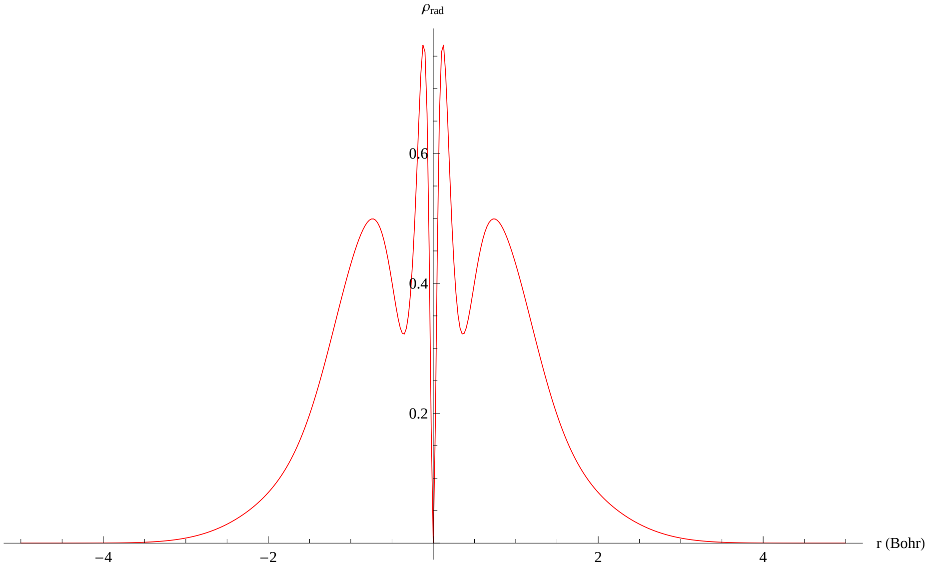
(a) Fitted radial electron densities of Li at (0,0,0) (Pink) and H at (0,0,3.1) (Orange) in LiH (Green).



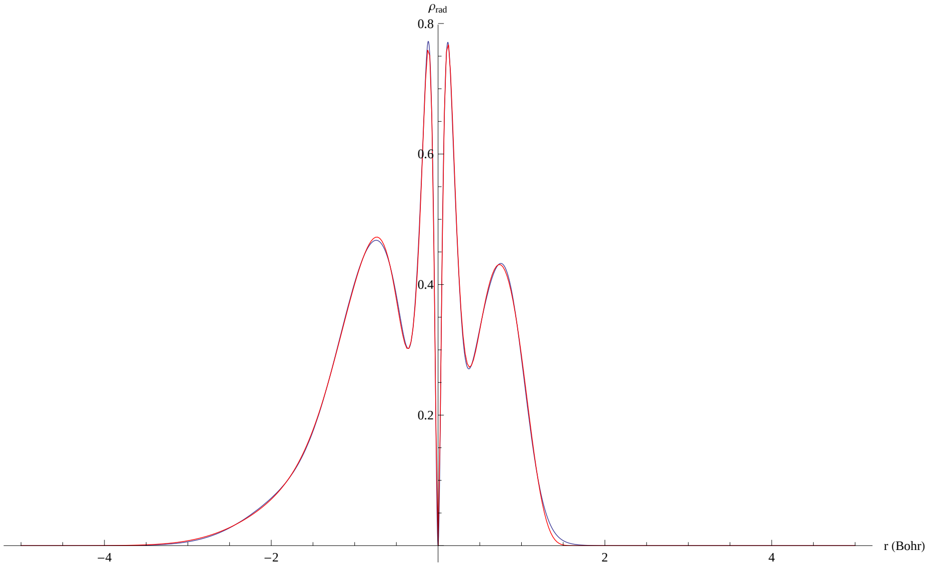
(b) Fitted (Green) vs actual (Red) molecular radial electron density.

Figure B.3.15: A comparison of radial electron density of the fitted AIM and the total molecular radial electron density (a) and a comparison of the fitted and the actual molecular radial electron density (b) of LiH (b) vs r in bohr.

B.3.6 HF

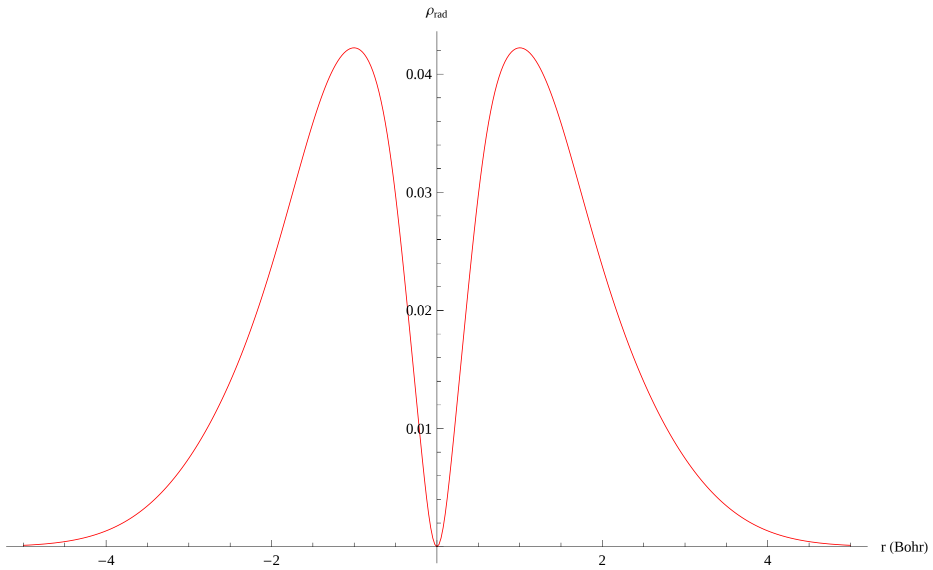


(a) Individual F atom radial electron density.

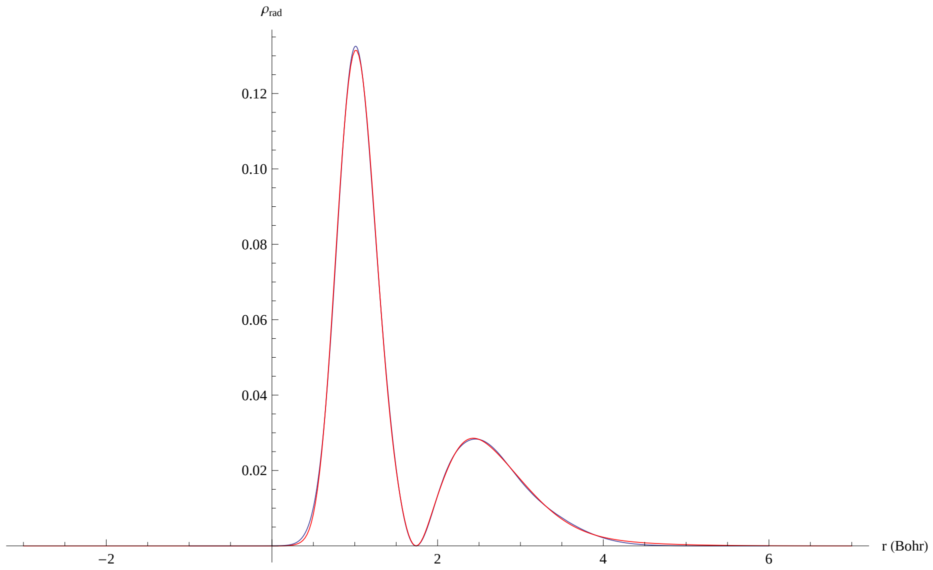


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.3.16: Radial electron density of the F atom (a) and a comparison of the fit and the actual radial electron density of F at (0,0,0) in HF (b) vs r in bohr.

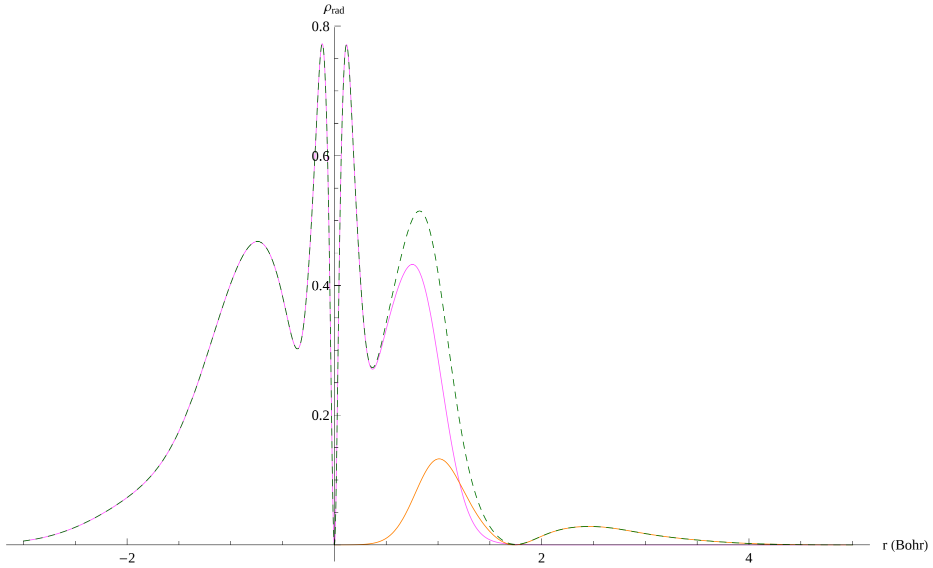


(a) Individual H atom radial electron density.

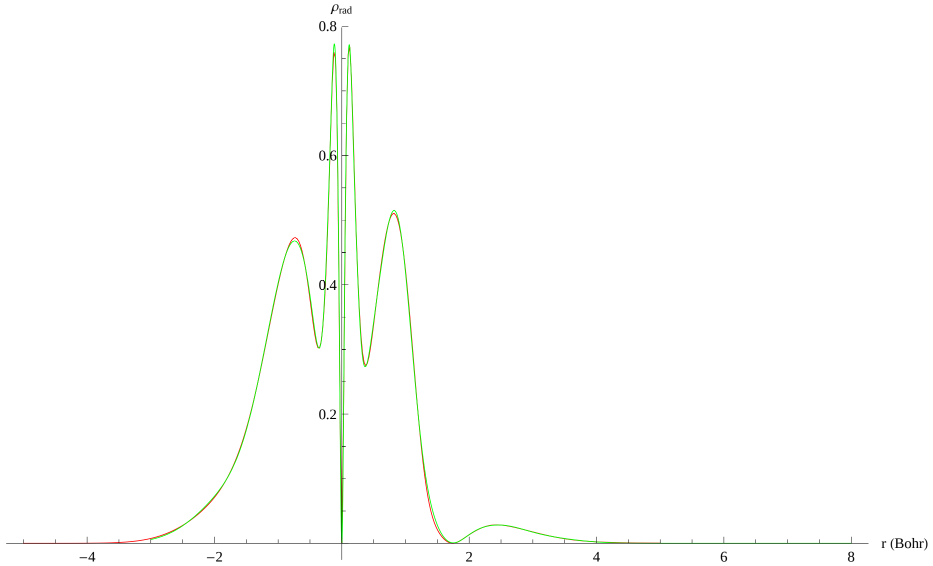


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.3.17: Radial electron density of the H atom (a) and a comparison of the fit and the actual radial electron density of H at (0,0,1.7) in HF (b) vs r in bohr.



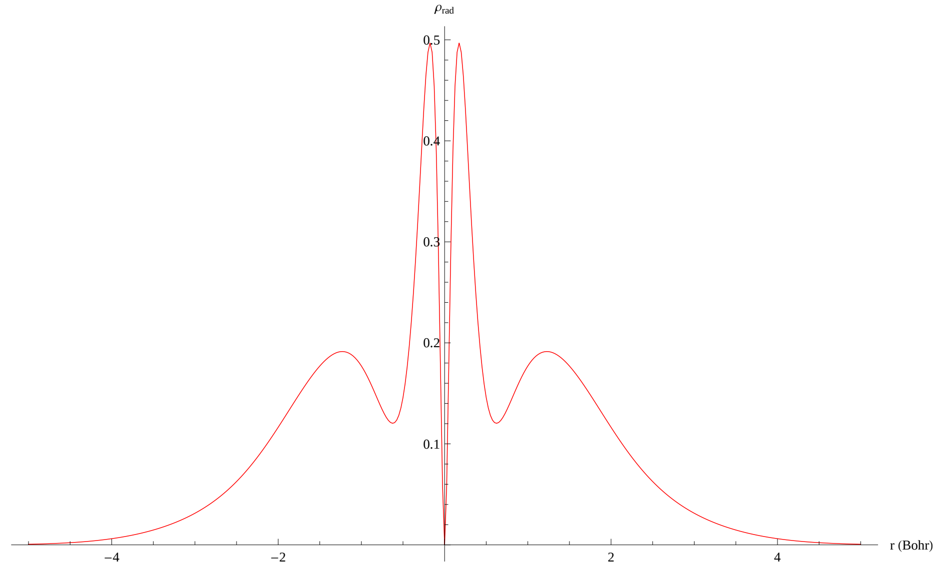
(a) Fitted radial electron densities of F at (0,0,0) (Pink) and H at (0,0,1.7) (Orange) in HF (Green).



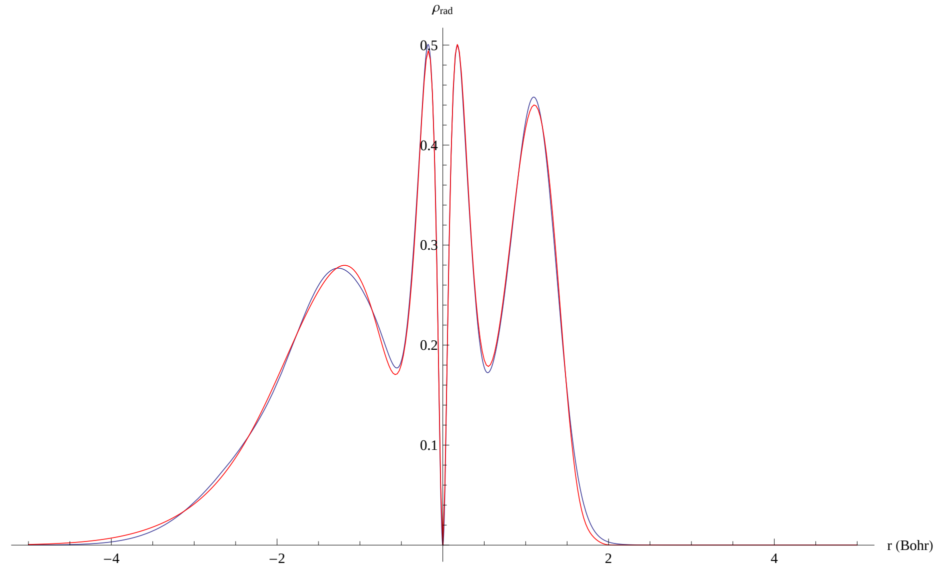
(b) Fitted (Green) vs actual (Red) molecular radial electron density.

Figure B.3.18: A comparison of radial electron density of the fitted AIM and the total molecular radial electron density (a) and a comparison of the fitted and the actual molecular radial electron density (b) of HF (b) vs r in bohr.

B.3.7 CO

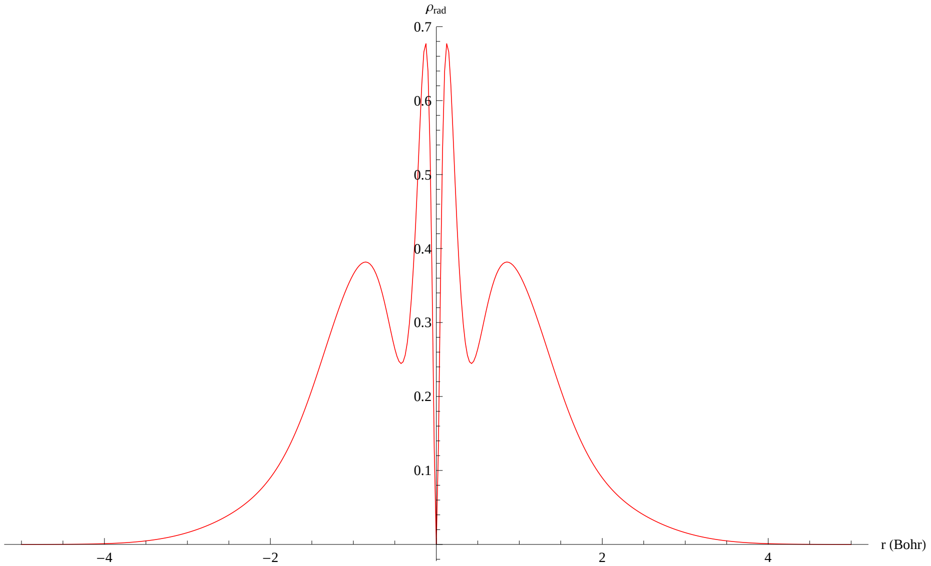


(a) Individual C atom radial electron density.

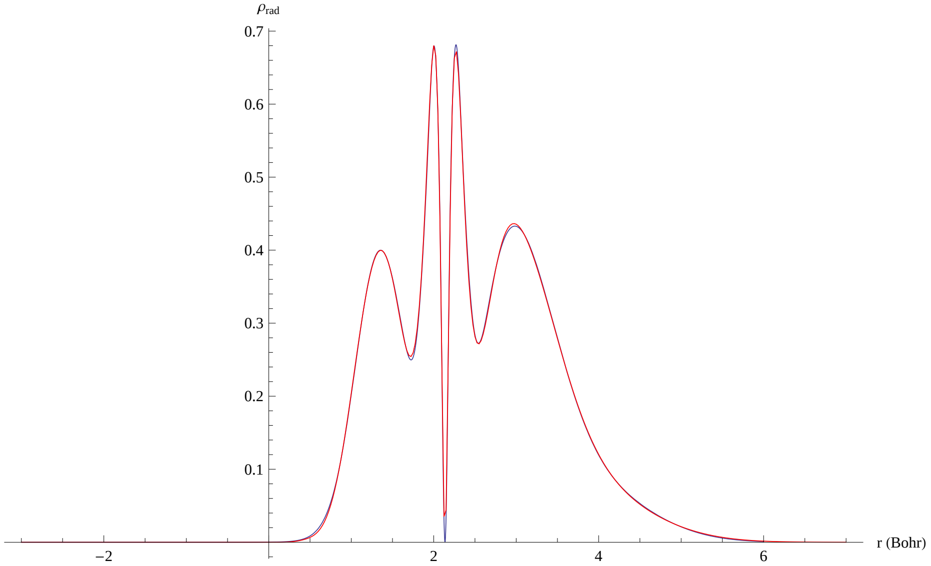


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.3.19: Radial electron density of the C atom (a) and a comparison of the fit and the actual radial electron density of C at (0,0,0) in CO (b) vs r in bohr.

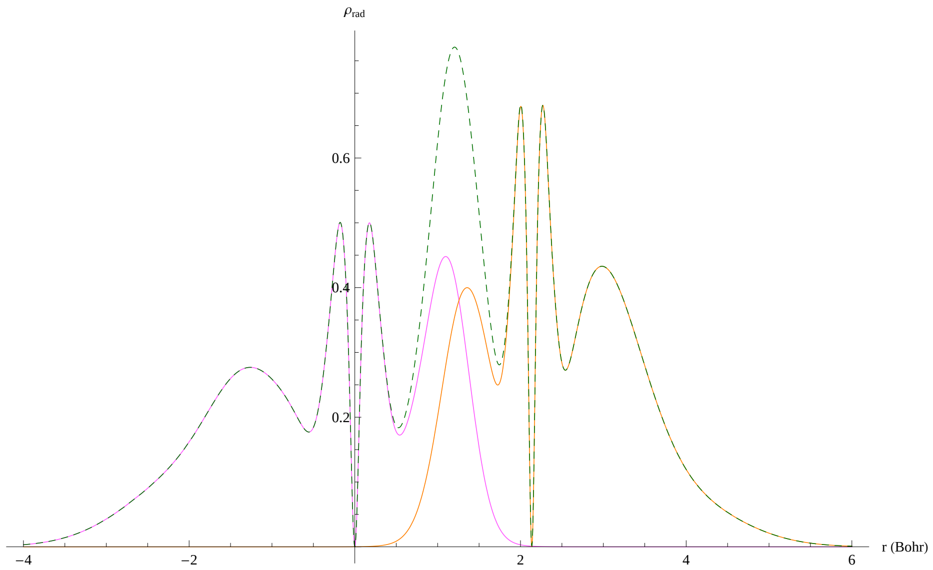


(a) Individual O atom radial electron density.

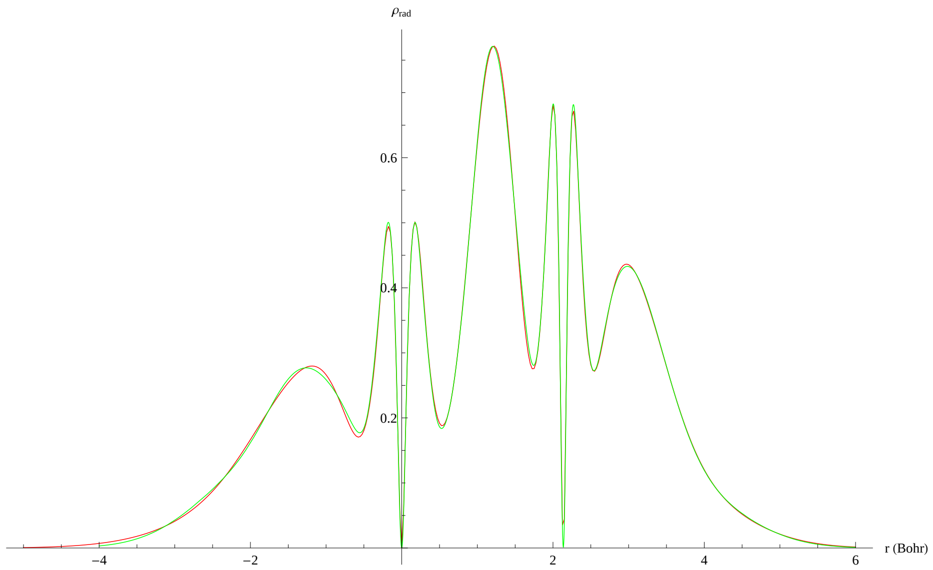


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.3.20: Radial electron density of the O atom (a) and a comparison of the fit and the actual radial electron density of O at (0,0,2.1) in CO (b) vs r in bohr.



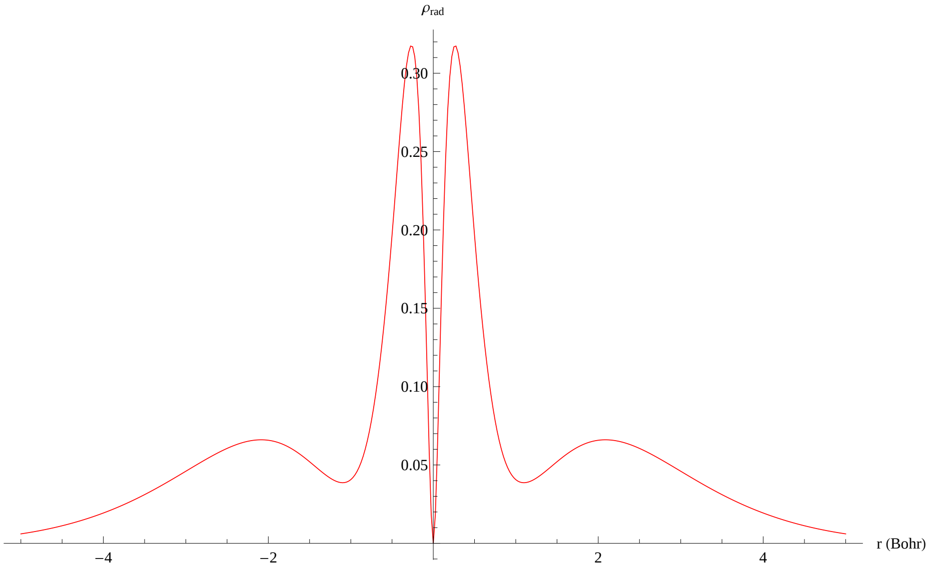
(a) Fitted radial electron densities of C at (0,0,0) (Pink) and O at (0,0,2.1) (Orange) in CO (Green).



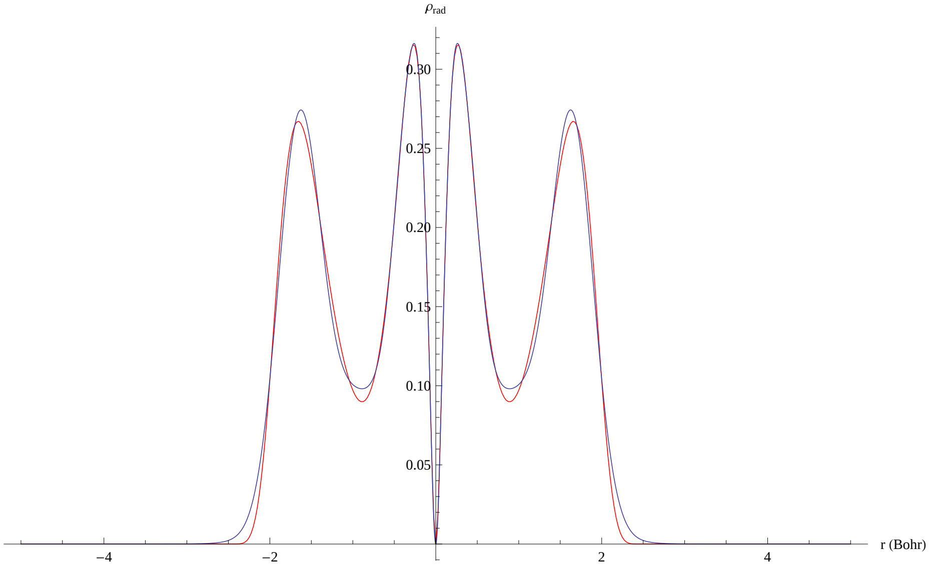
(b) Fitted (Green) vs actual (Red) molecular radial electron density.

Figure B.3.21: A comparison of radial electron density of the fitted AIM and the total molecular radial electron density (a) and a comparison of the fitted and the actual molecular radial electron density (b) of CO (b) vs r in bohr.

B.3.8 BeH_2

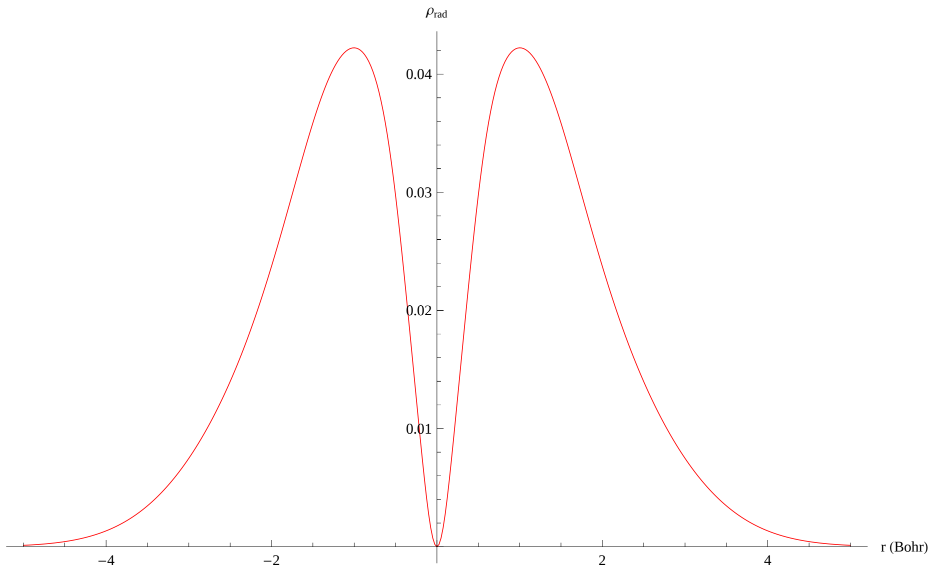


(a) Individual Be atom radial electron density.

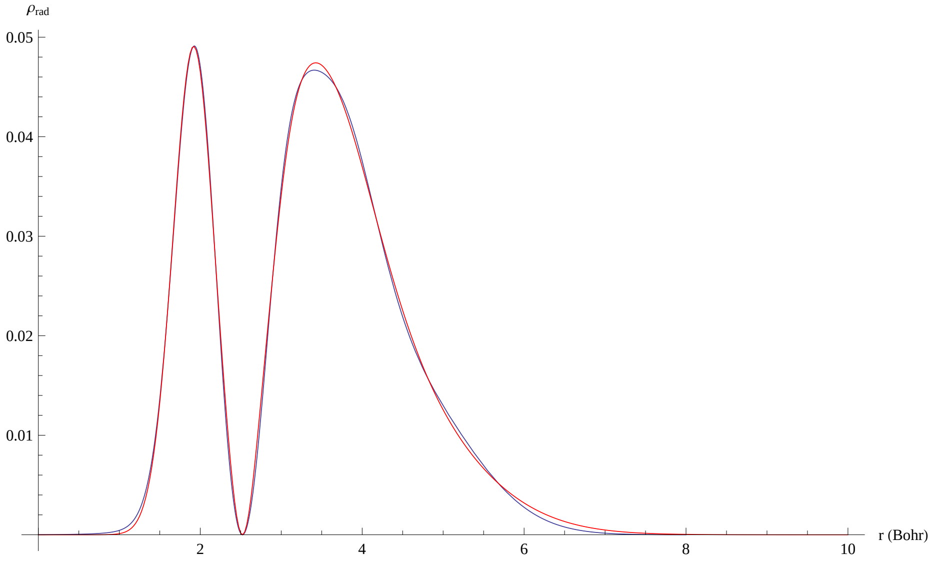


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.3.22: Radial electron density of the Be atom (a) and a comparison of the fit and the actual radial electron density of Be at (0,0,0) in BeH_2 (b) vs r in bohr.

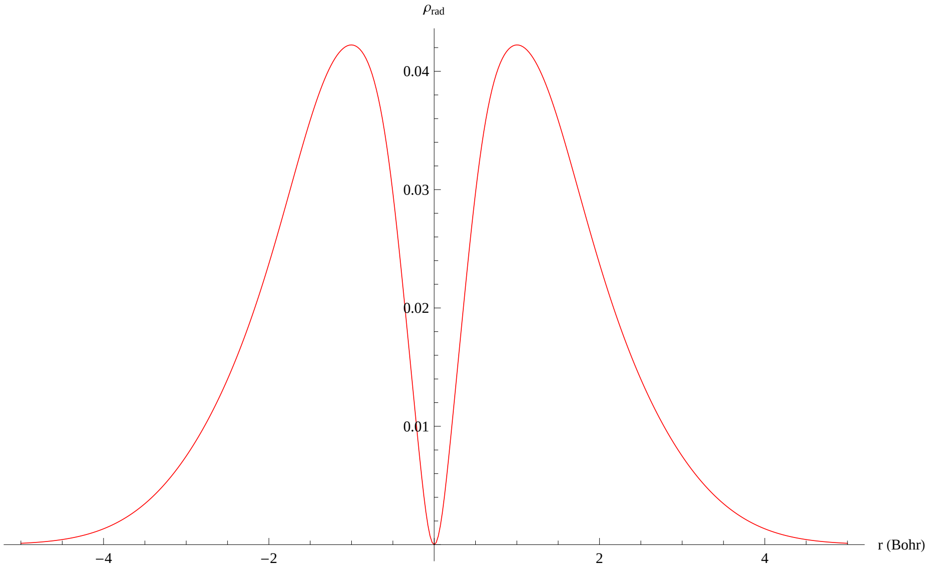


(a) Individual H atom radial electron density.

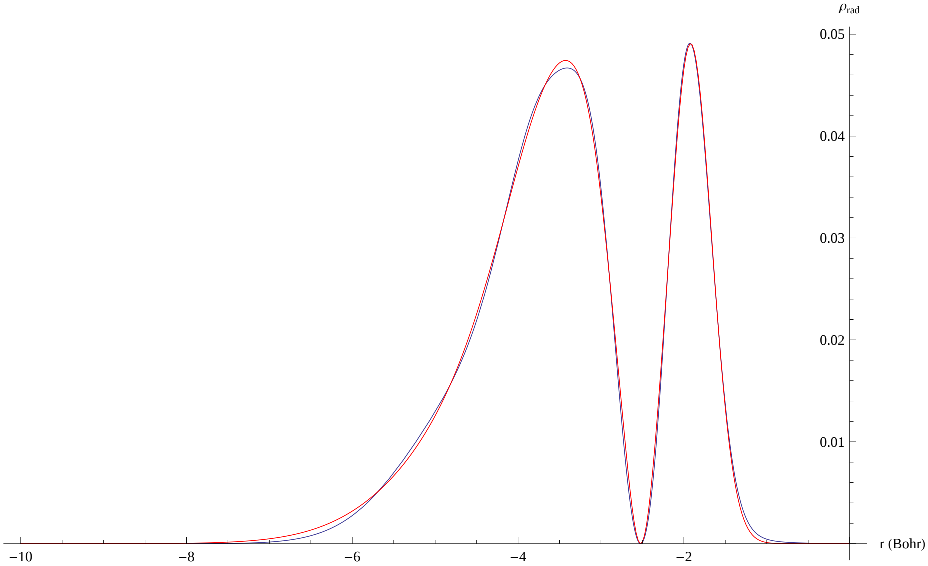


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.3.23: Radial electron density of the H atom (a) and a comparison of the fit and the actual radial electron density of H at (0,0,2.5) in BeH_2 (b) vs r in bohr.

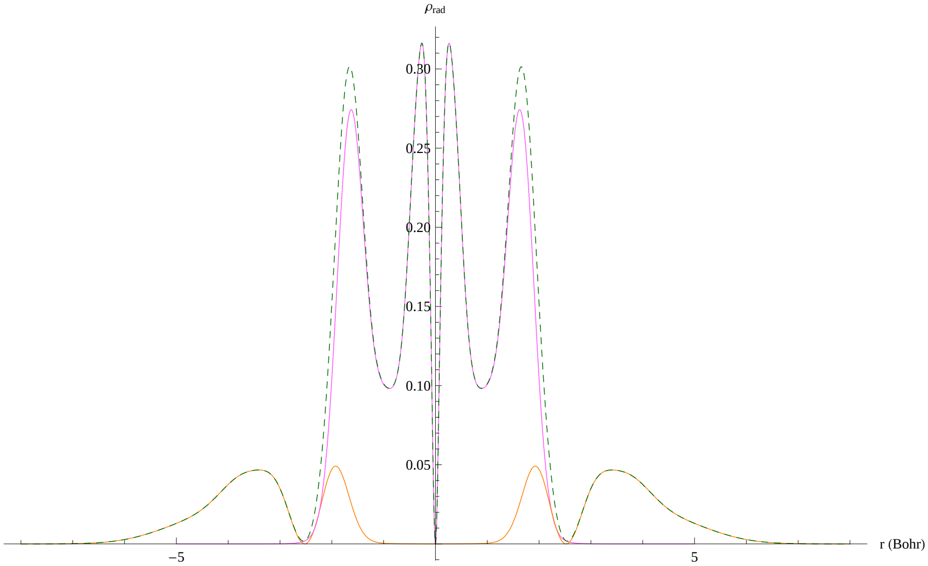


(a) Individual H atom radial electron density.

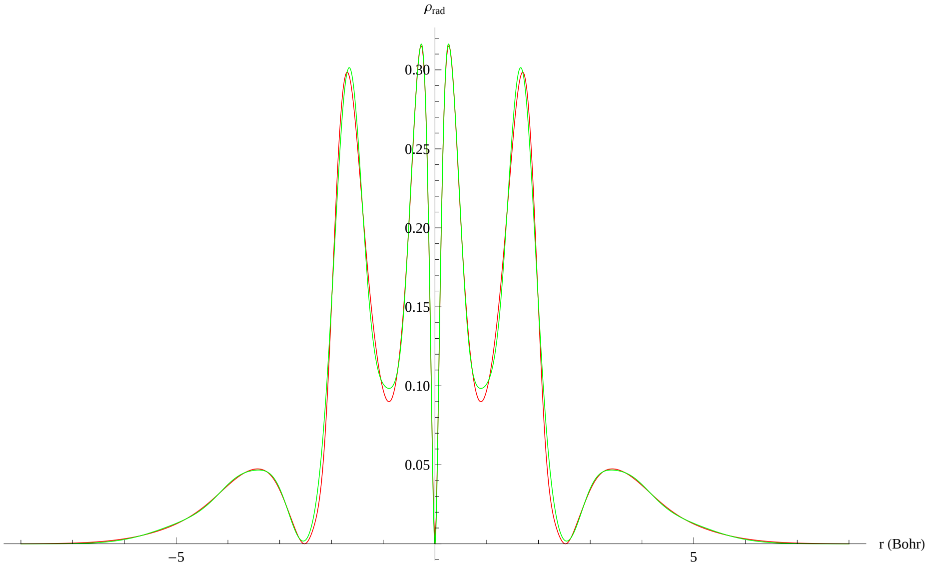


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.3.24: Radial electron density of the H atom (a) and a comparison of the fit and the actual radial electron density of H at (0,0,-2.5) in BeH_2 (b) vs r in bohr.



(a) Fitted radial electron densities of Be at (0,0,0) (Pink), H at (0,0,2.5) (Orange) and H at (0,0,-2.5) (Orange) in BeH_2 (Green).

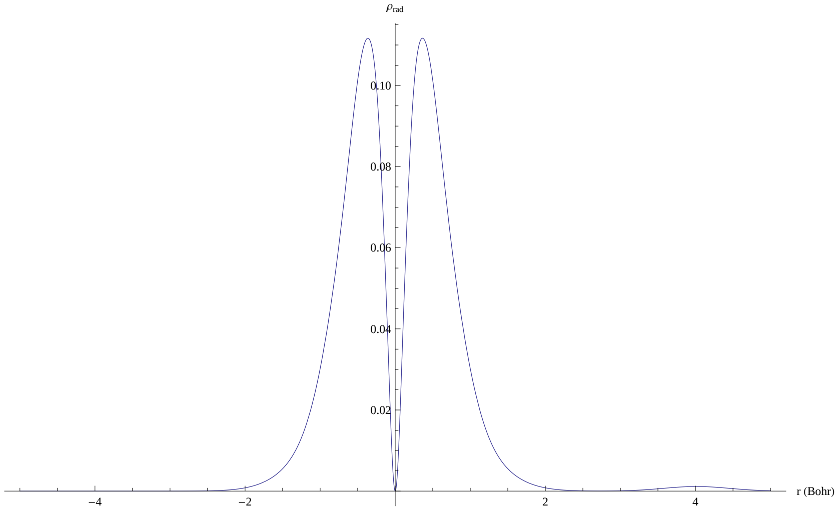


(b) Fitted (Green) vs actual (Red) molecular radial electron density.

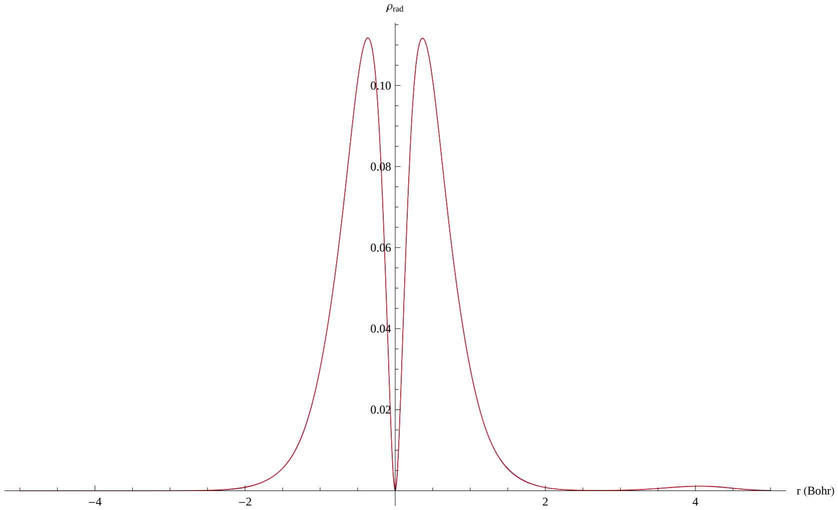
Figure B.3.25: A comparison of radial electron density of the fitted AIM and the total molecular radial electron density (a) and a comparison of the fitted and the actual molecular radial electron density (b) of BeH_2 (b) vs r in bohr.

B.4 Molecular Orbital Densities

B.4.1 Li in Li_2

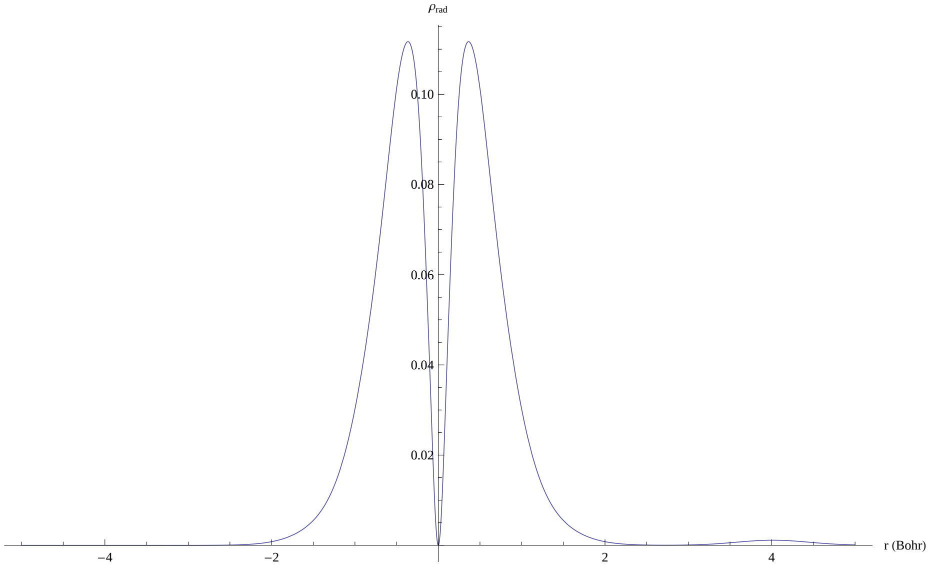


(a) Fitted radial electron density.

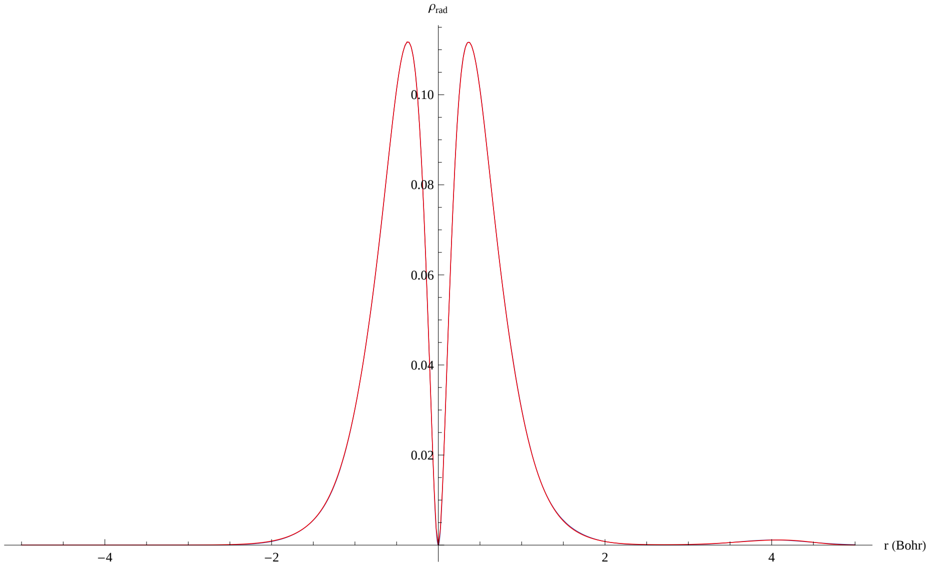


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.4.1: Fitted radial electron density (a) and a comparison of the fitted and the actual radial electron density (b) of molecular orbital one of Li in Li_2 vs r in bohr.

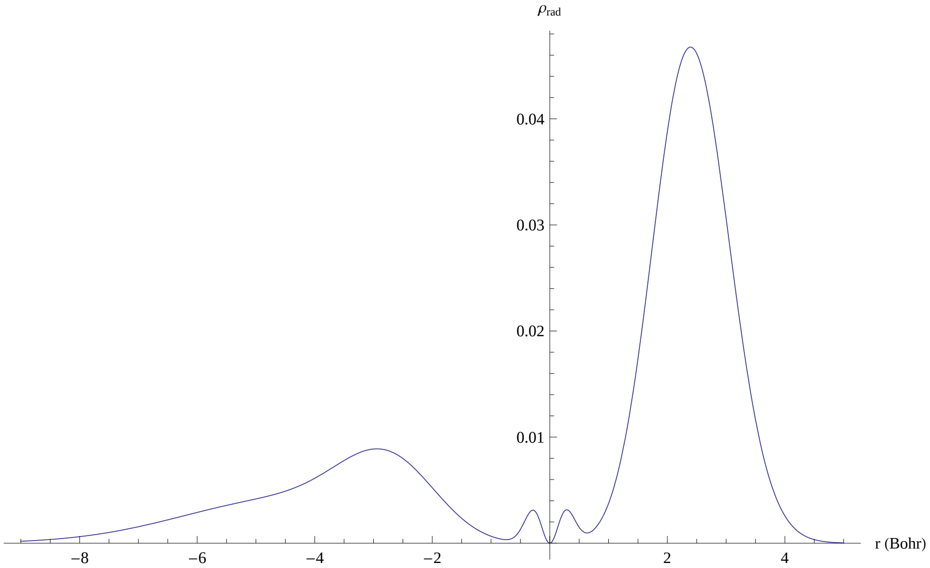


(a) Fitted radial electron density.

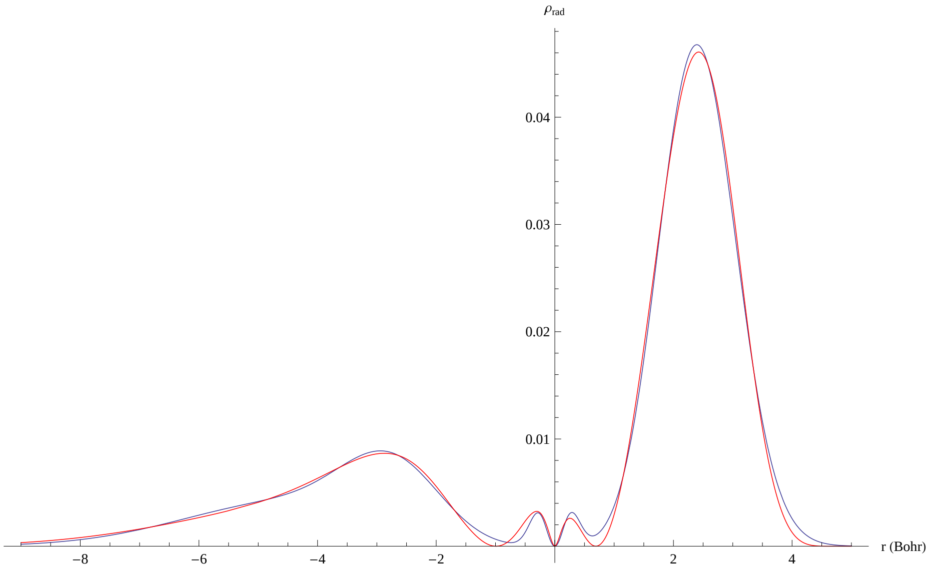


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.4.2: Fitted radial electron density (a) and a comparison of the fitted and the actual radial electron density (b) of molecular orbital two of Li in Li_2 vs r in bohr.

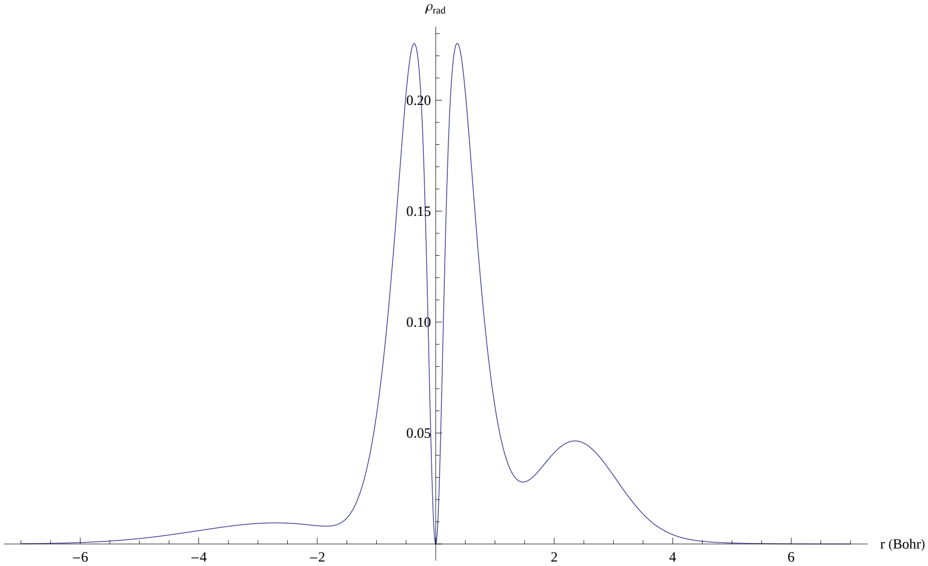


(a) Fitted Density of MO Three

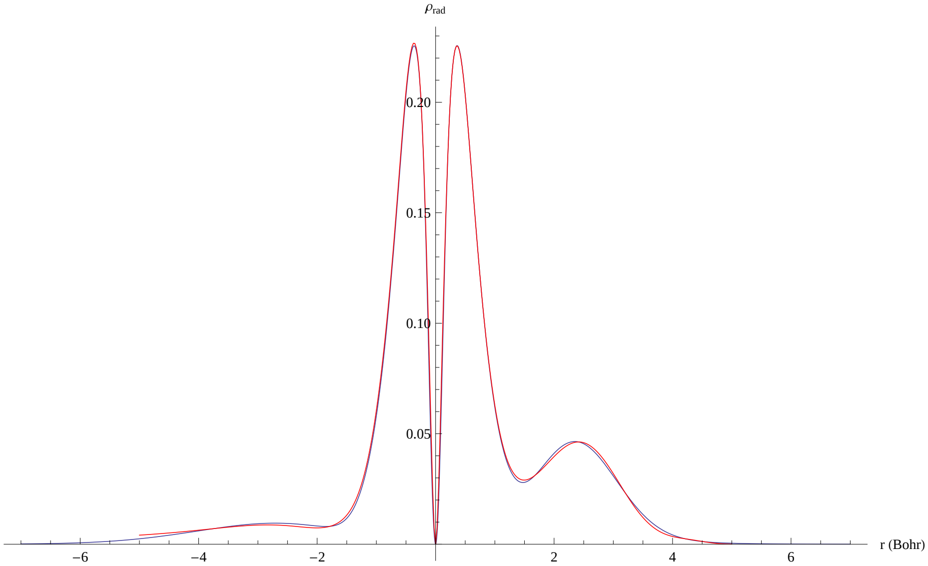


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.4.3: Fitted radial electron density (a) and a comparison of the fitted and the actual radial electron density (b) of molecular orbital three of Li in Li_2 vs r in bohr.



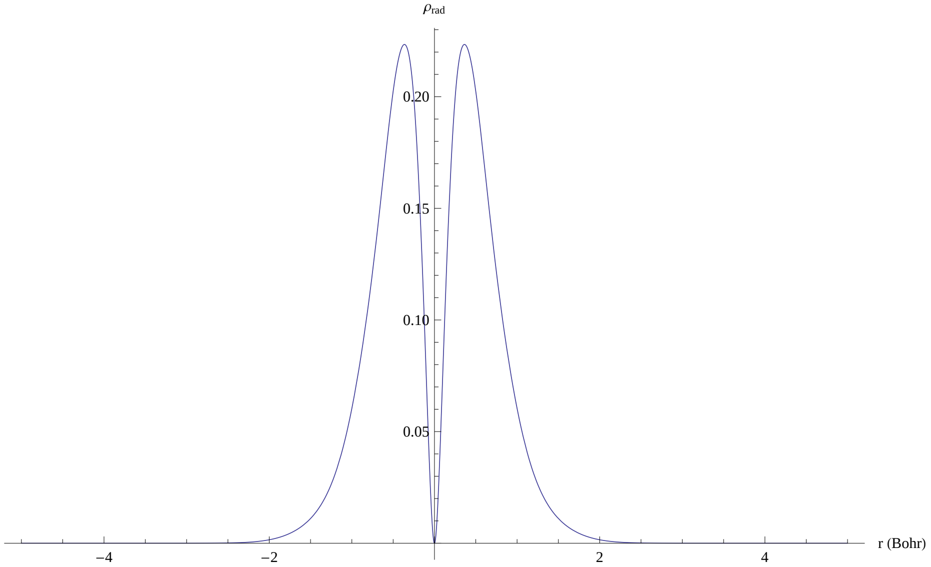
(a) Total fitted radial electron density.



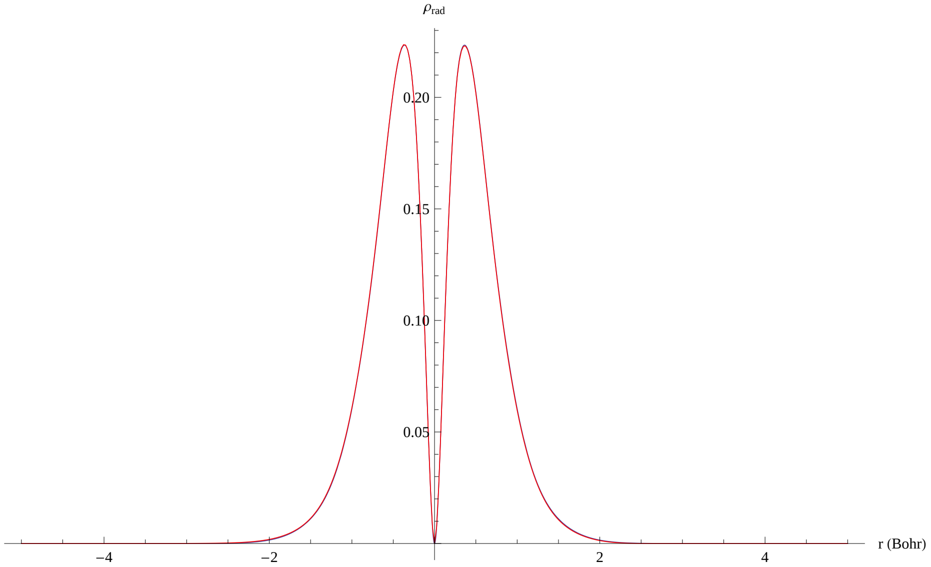
(b) Total fitted (Blue) vs actual (Red) radial electron density.

Figure B.4.4: Total fitted radial electron density (a) and a comparison of the fitted and the actual total radial electron density (b) of Li in Li_2 vs r in bohr.

B.4.2 Li in LiH

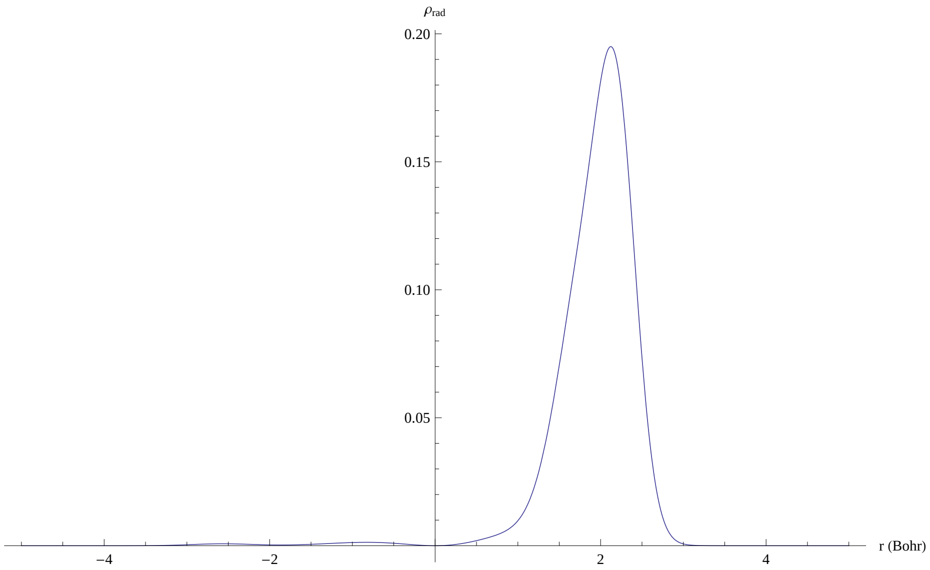


(a) Fitted radial electron density.

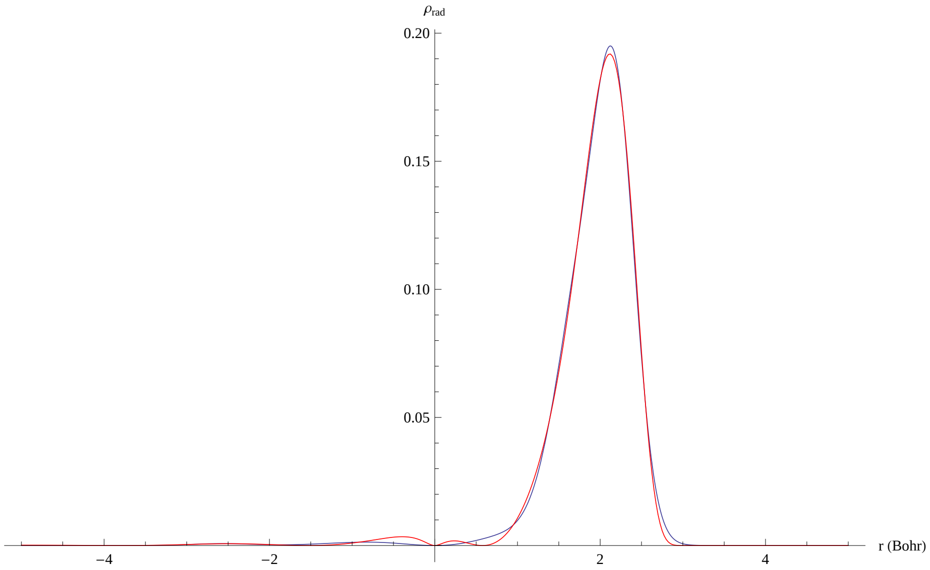


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.4.5: Fitted radial electron density (a) and a comparison of the fitted and the actual(Red) radial electron density (b) of molecular orbital one of Li in LiH vs r in bohr.

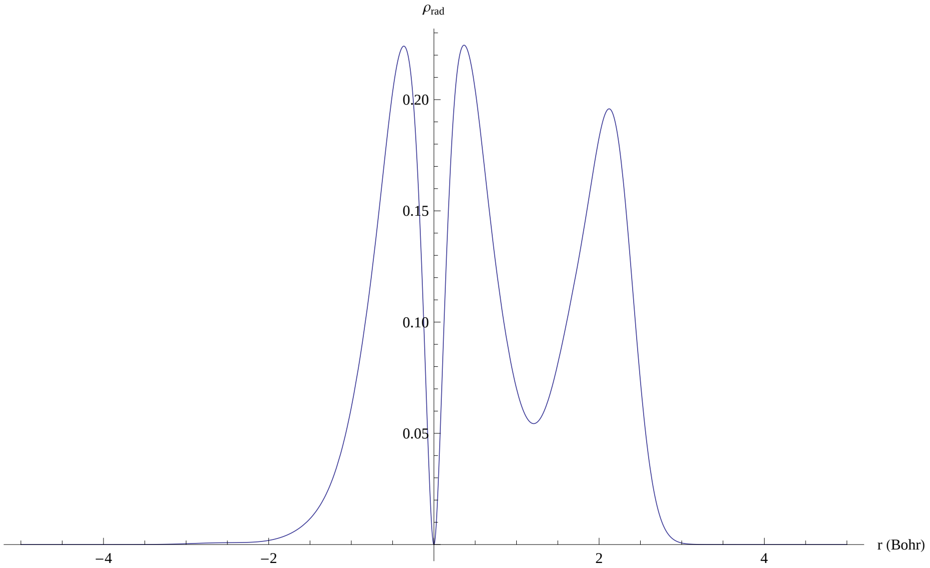


(a) Fitted radial electron density.

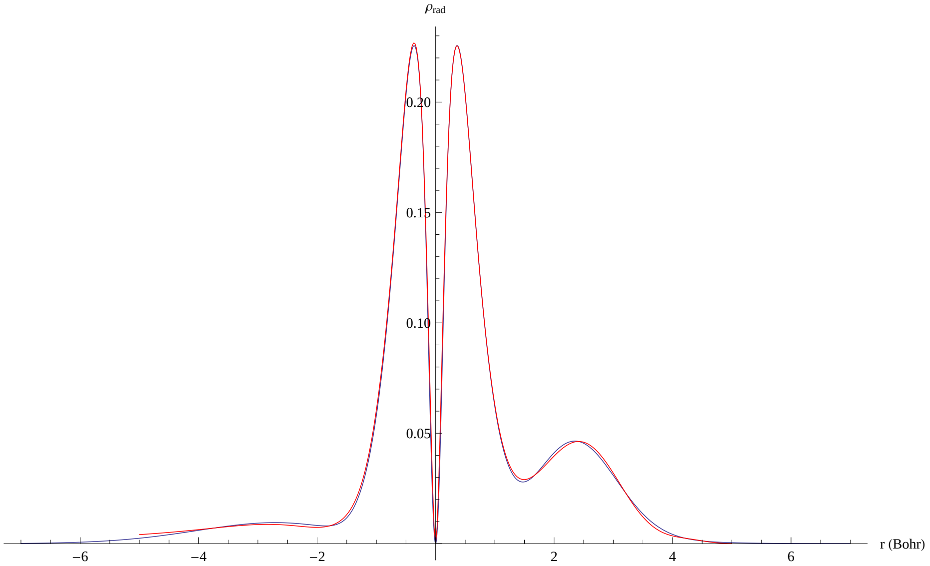


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.4.6: Fitted radial electron density (a) and a comparison of the fitted and the actual radial electron density (b) of molecular orbital two of Li in LiH vs r in bohr.



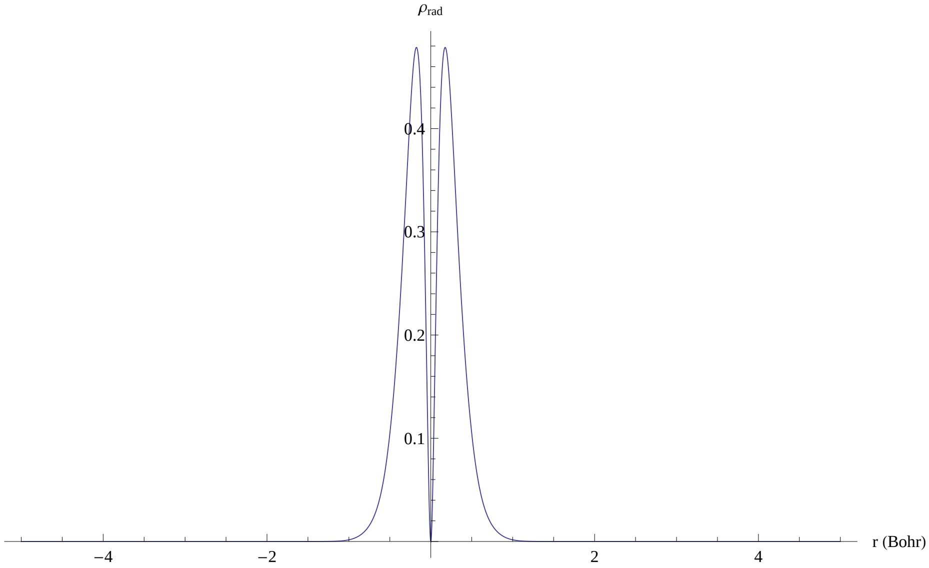
(a) Total fitted radial electron density.



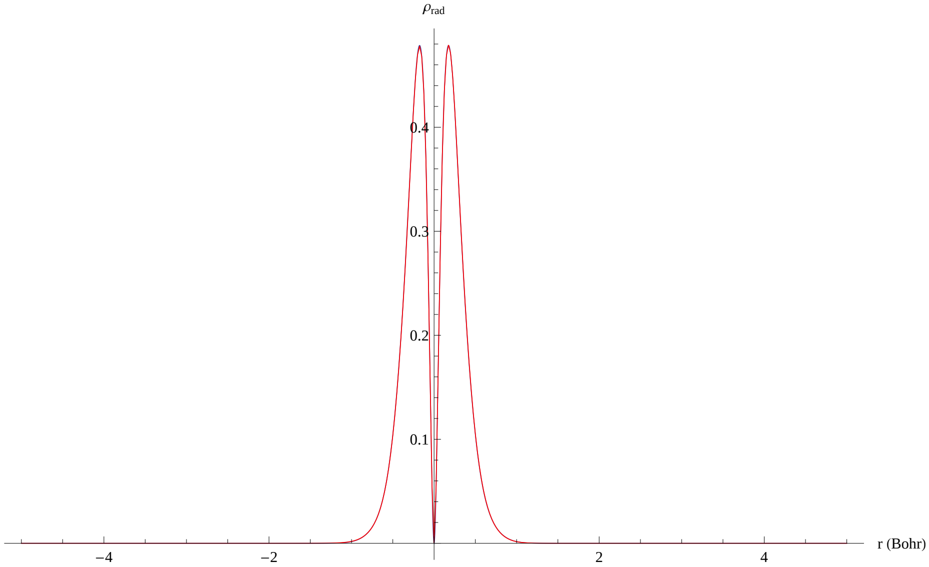
(b) Total fitted (Blue) vs actual (Red) radial electron density.

Figure B.4.7: Total fitted radial electron density (a) and a comparison of the fitted and the actual total radial electron density (b) of Li in LiH vs r in bohr.

B.4.3 C in CO

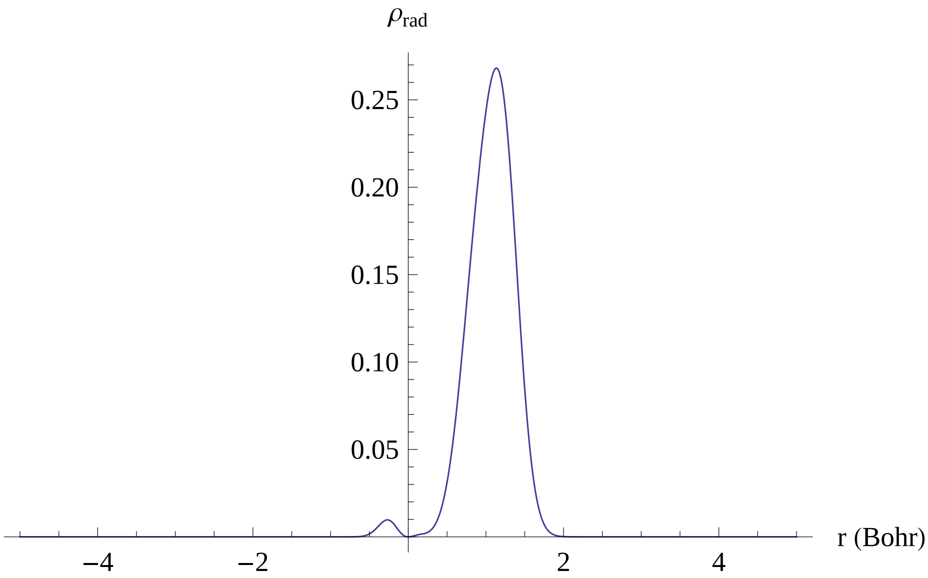


(a) Fitted radial electron density.

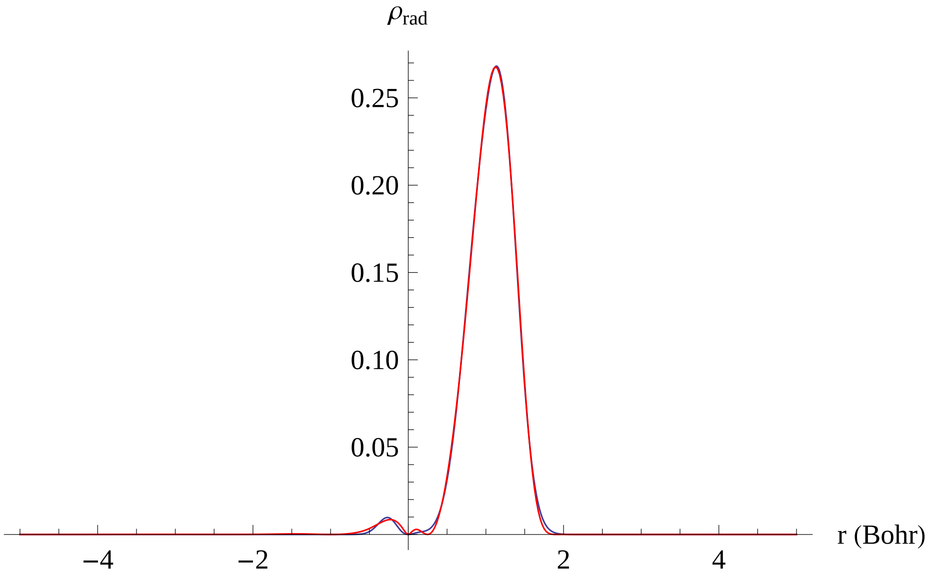


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.4.8: Fitted radial electron density (a) and a comparison of the fitted and the actual radial electron density (b) of molecular orbital two of C in CO vs r in bohr.

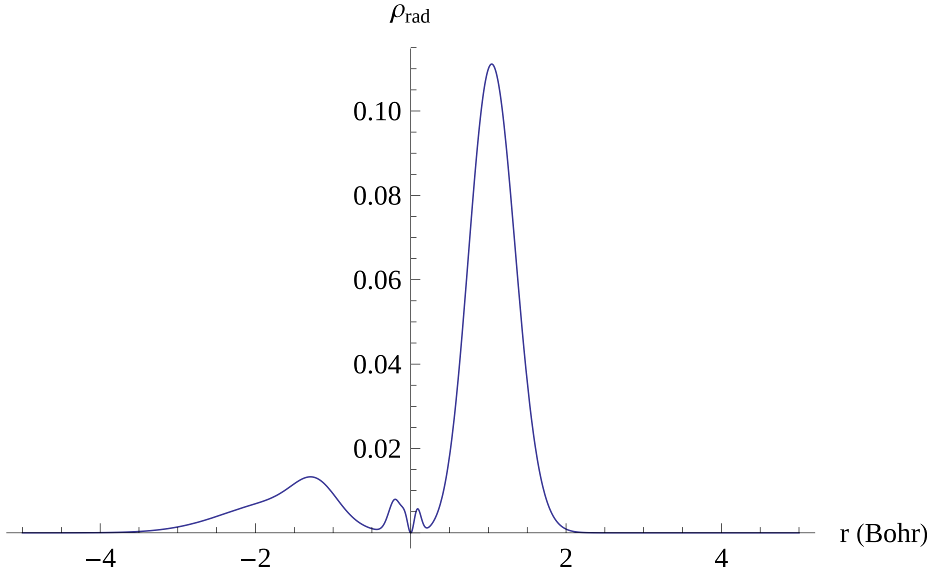


(a) Fitted radial electron density.

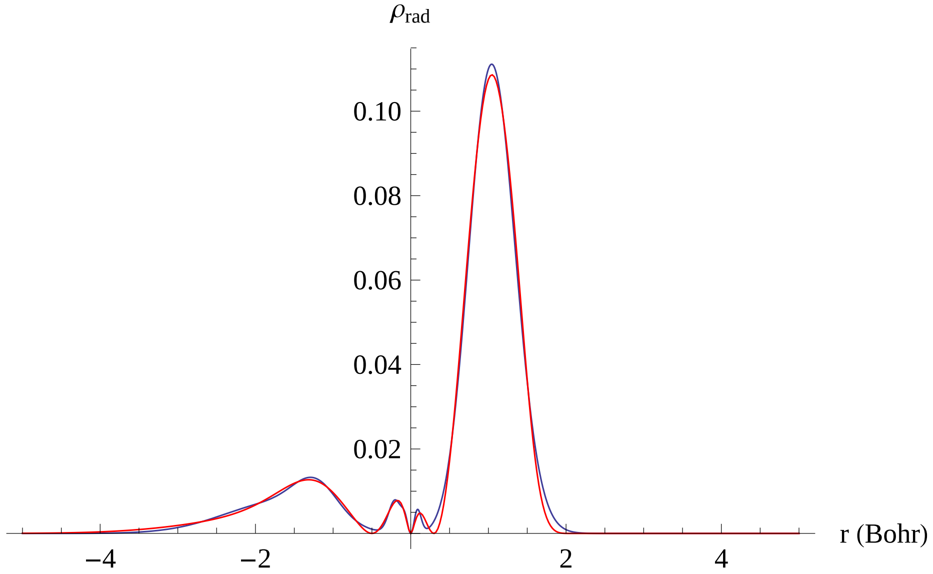


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.4.9: Fitted radial electron density (a) and a comparison of the fitted and the actual radial electron density (b) of molecular orbital three of C in CO vs r in bohr.

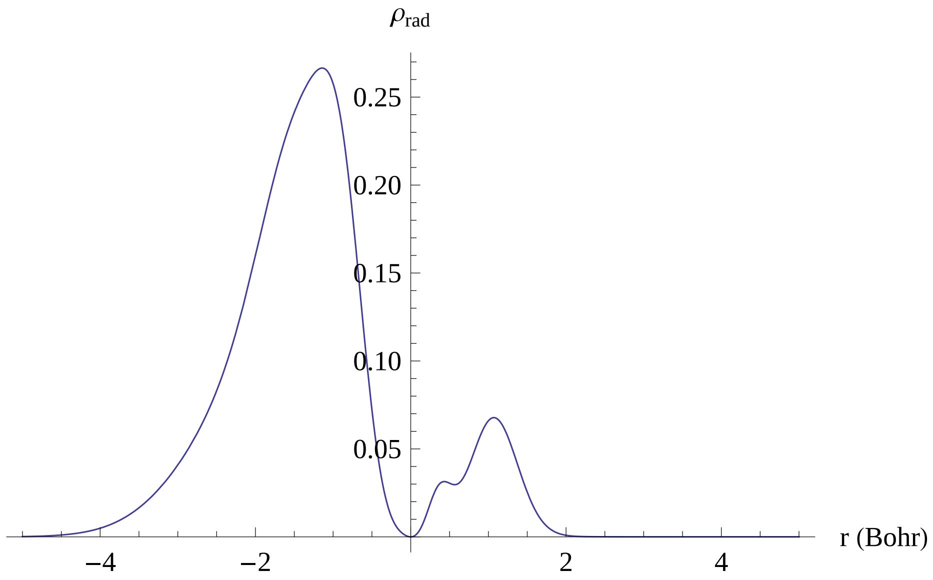


(a) Fitted radial electron density.

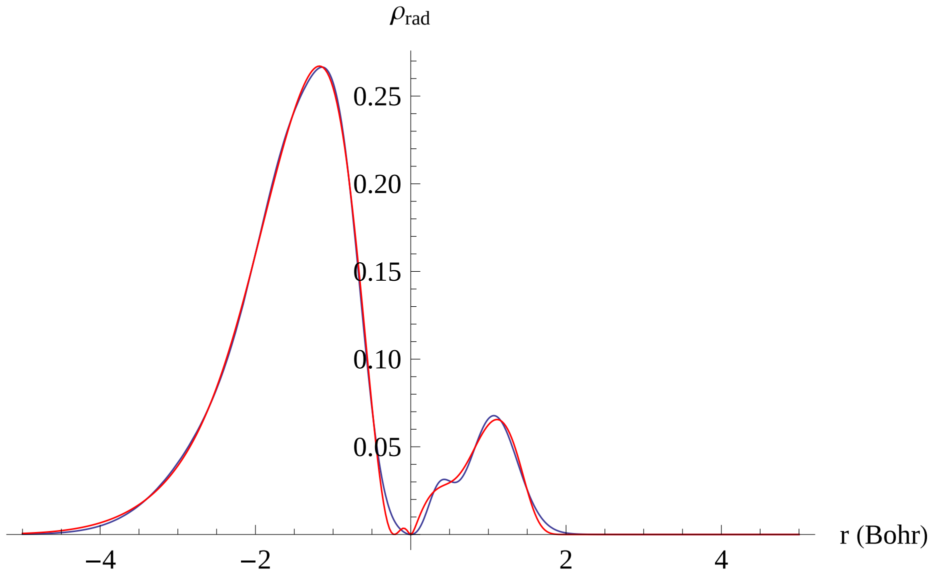


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.4.10: Fitted radial electron density (a) and a comparison of the fitted and the actual radial electron density (b) of molecular orbital four of C in CO vs r in bohr.

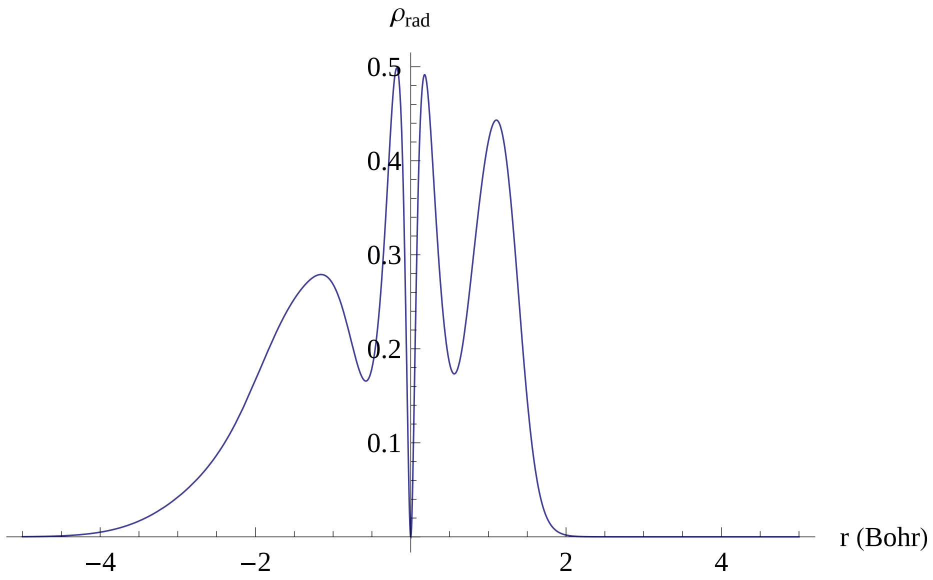


(a) Fitted radial electron density.

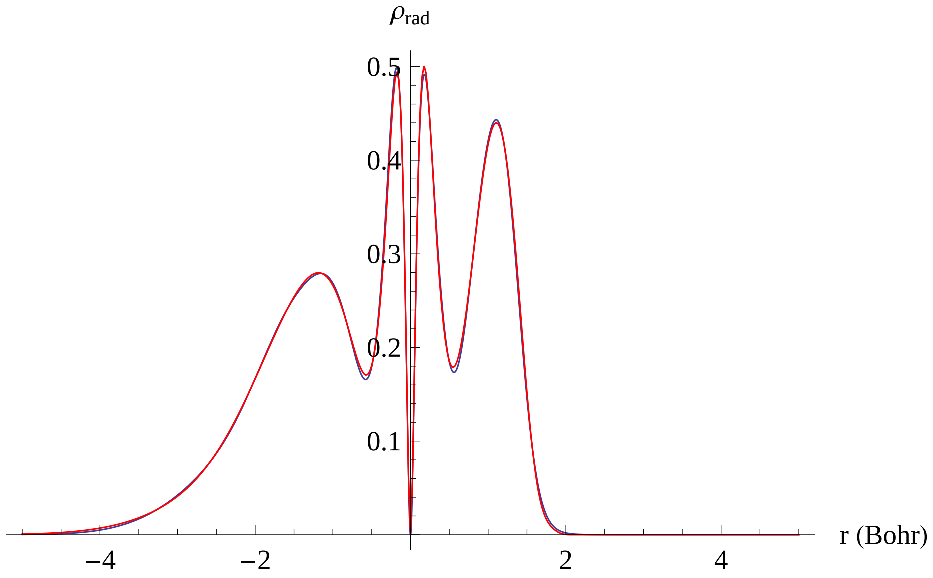


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.4.11: Fitted radial electron density (a) and a comparison of the fitted and the actual radial electron density (b) of molecular orbital seven of C in CO vs r in bohr.



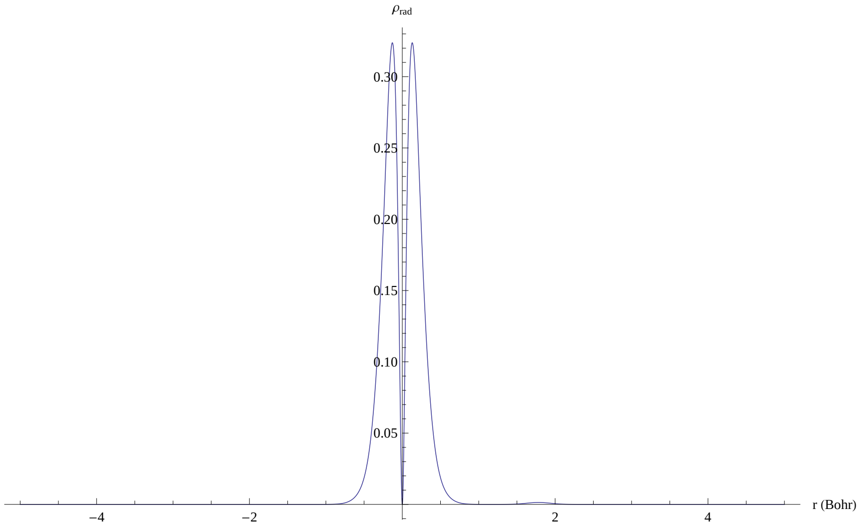
(a) Total fitted radial electron density.



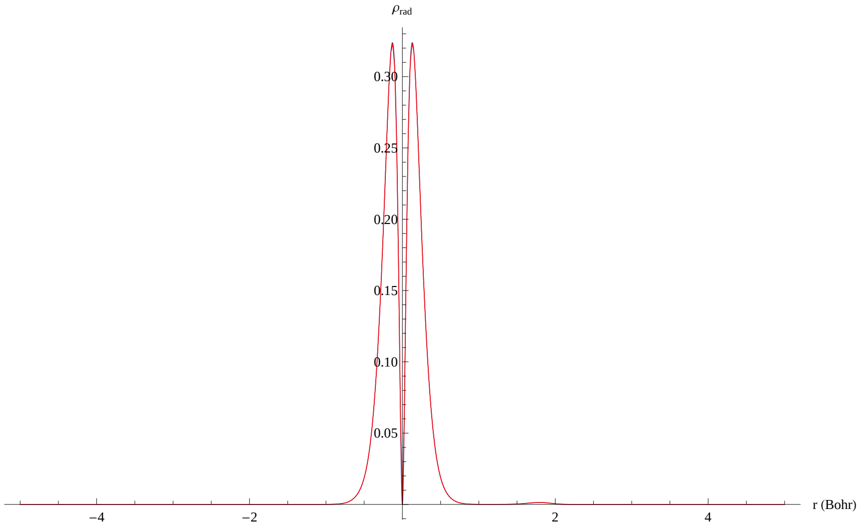
(b) Total fitted (Blue) vs actual (Red) radial electron density.

Figure B.4.12: Total fitted radial electron density (a) and a comparison of the fitted and the actual total radial electron density (b) of C in CO vs r in bohr.

B.4.4 O in O_2

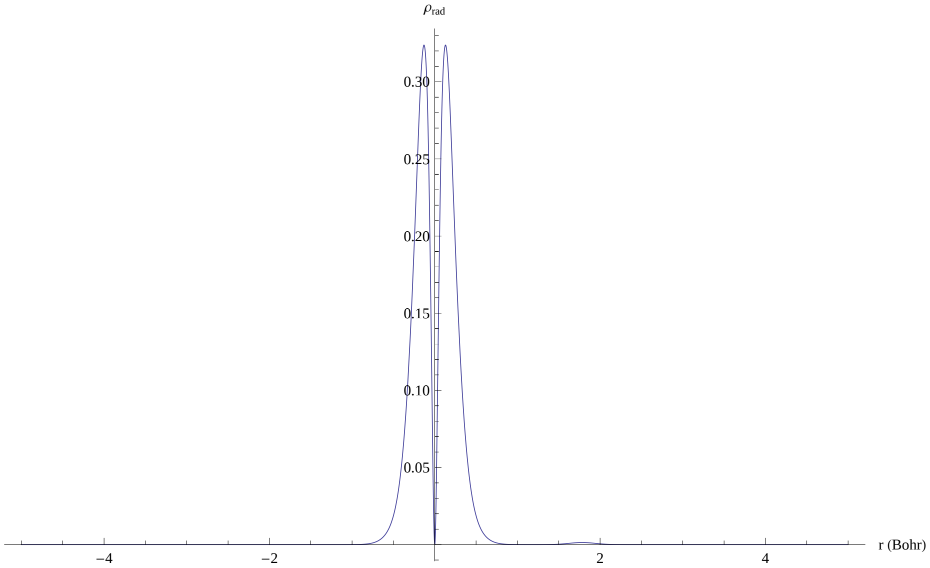


(a) Fitted radial electron density.

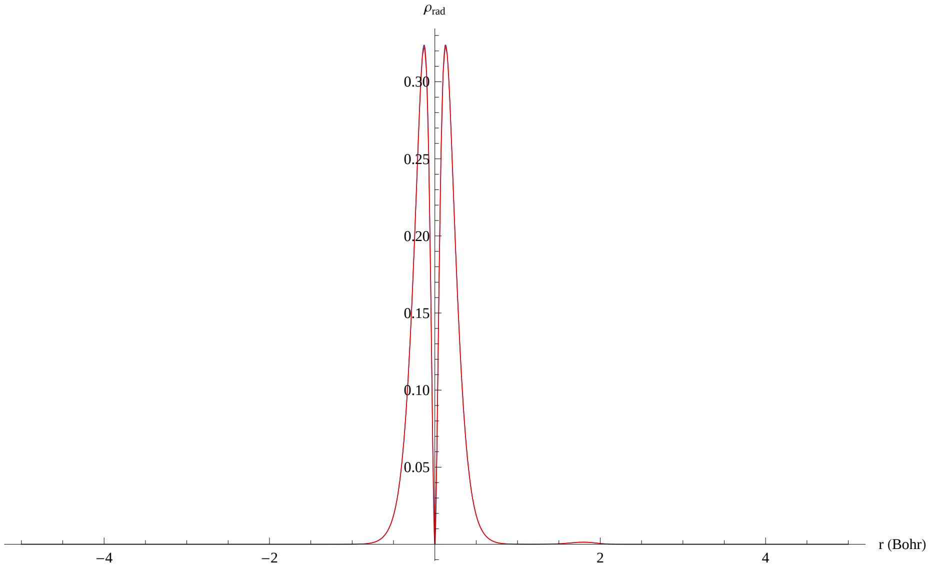


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.4.13: Fitted radial electron density (a) and a comparison of the fitted and the actual radial electron density (b) of molecular orbital one of O in O_2 vs r in bohr.

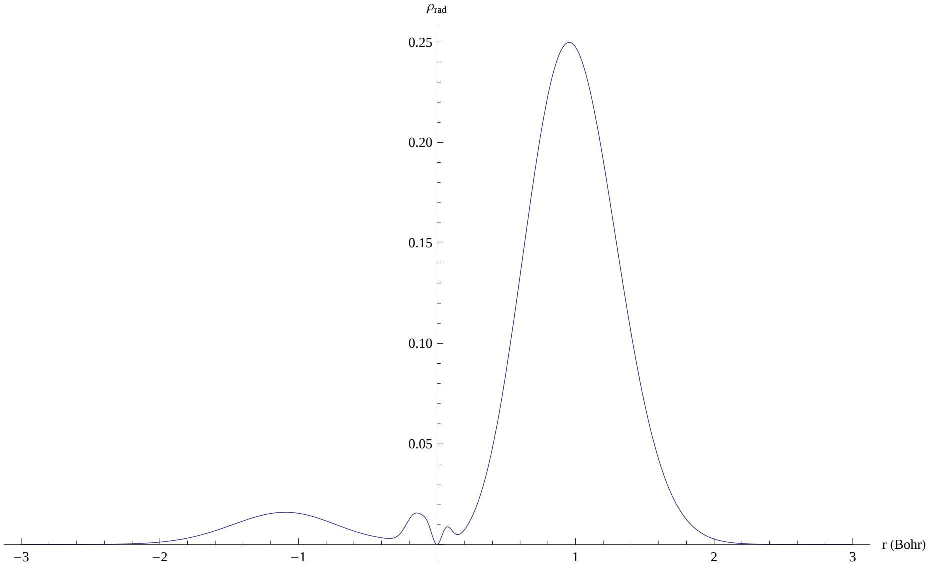


(a) Fitted radial electron density.

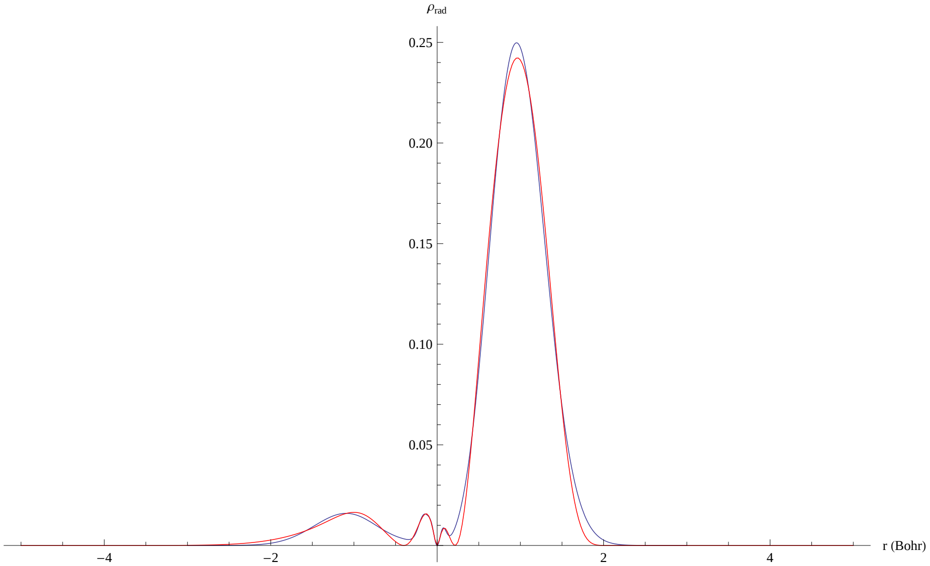


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.4.14: Fitted radial electron density (a) and a comparison of the fitted and the actual radial electron density (b) of molecular orbital two of O in O_2 vs r in bohr.

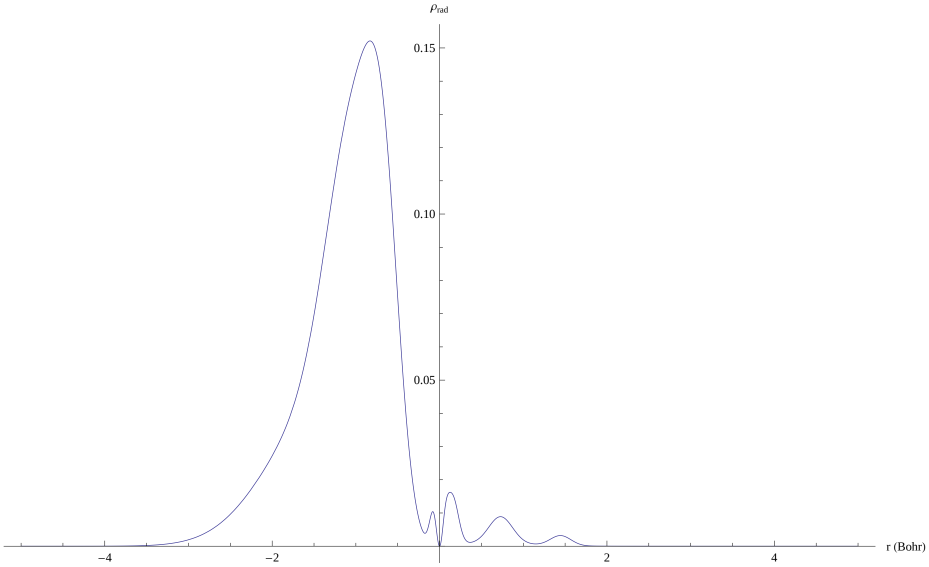


(a) Fitted radial electron density.

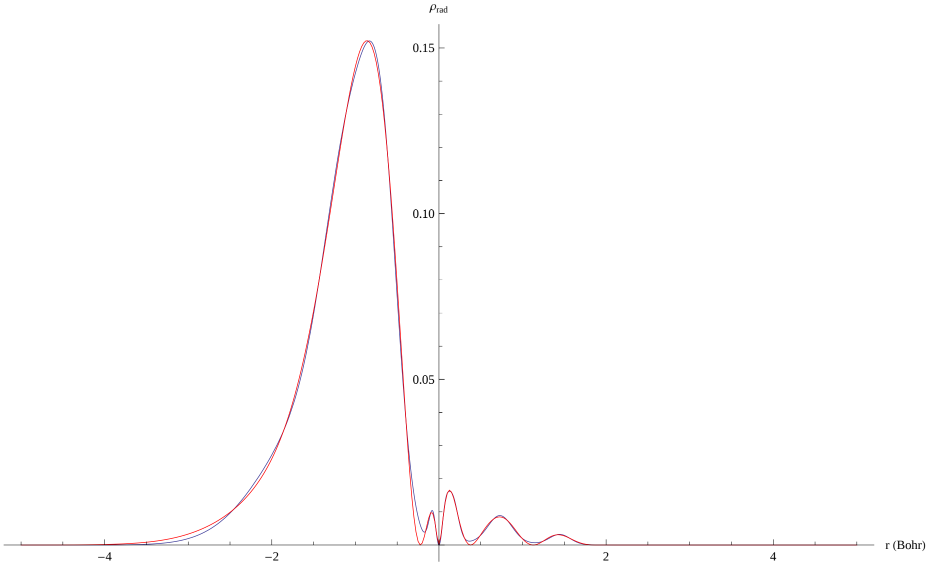


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.4.15: Fitted radial electron density (a) and a comparison of the fitted and the actual radial electron density (b) of molecular orbital three of O in O_2 vs r in bohr.

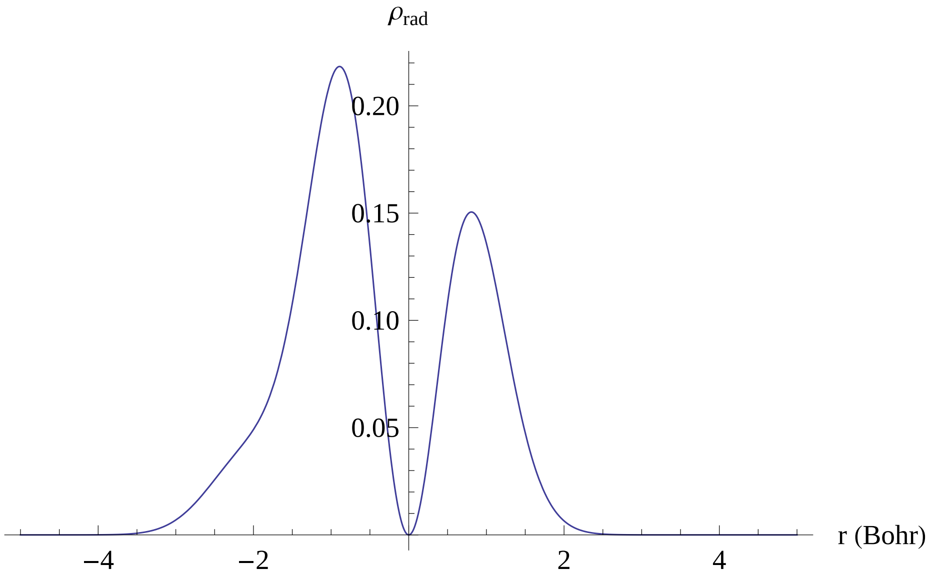


(a) Fitted radial electron density.

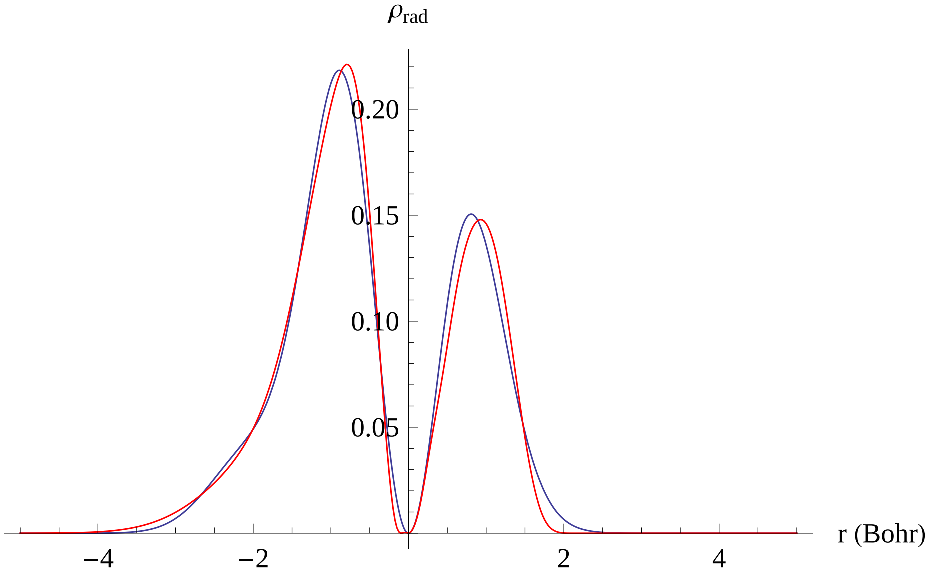


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.4.16: Fitted radial electron density (a) and a comparison of the fitted and the actual radial electron density (b) of molecular orbital four of O in O_2 vs r in bohr.

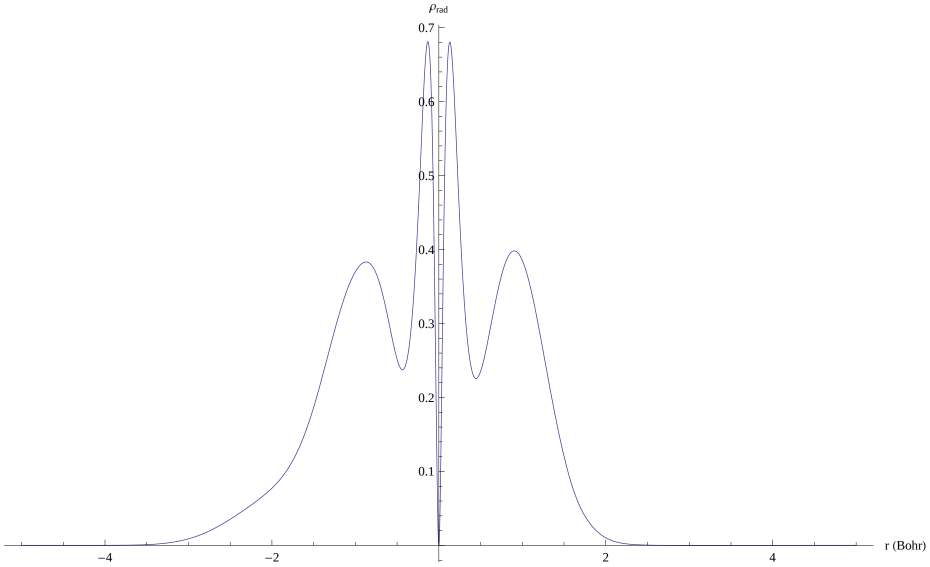


(a) Fitted radial electron density.

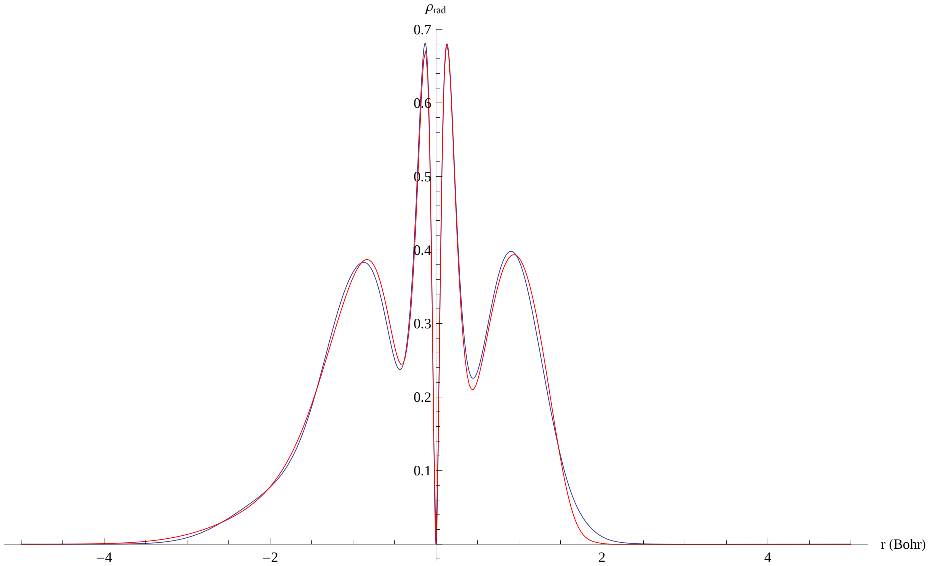


(b) Fitted (Blue) vs actual (Red) radial electron density.

Figure B.4.17: Fitted radial electron density (a) and a comparison of the fitted and the actual radial electron density (b) of molecular orbital six of O in O_2 vs r in bohr.



(a) Total fitted radial electron density.



(b) Total fitted (Blue) vs actual (Red) radial electron density.

Figure B.4.18: Total fitted radial electron density (a) and a comparison of the fitted and the actual total radial electron density (b) of O in O_2 vs r in bohr.