Roothaan-Hartree–Fock wave functions implemented in Python

Computational Quantum Physics

Giorgos Michaildis

ARISTOTLE UNIVERSITY OF THESSALONIKI

School of Physics

June 2025

Chapter 1

Introduction

This project focuses on the study of the Shannon entropy in atoms across the periodic table. The main objective is to analyze the behavior of the Shannon entropy as a function of the atomic number Z. To achieve this, we first computed the electronic wavefunctions of atoms which form the basis for calculating the Shannon entropy. From the wavefunctions, we derived the expression of the electron density, both in the position space $\rho(\mathbf{r})$ and in the momentum space $n(\mathbf{k})$, which closely relate with the Shannon entropy.

For one to obtain the atom wavefunctions, the most common way, is the Roothaan-Hartree-Fock method, (Bunge et al.; 1993). It is an iterative algorithm that approximates atomic orbitals along with system's energy. In the following chapter (Chapter 2), we will discuss our implantation of the Roothaan-Hartree-Fock method in Python, which provided as with the wavefunctions of the atoms in the range of $Z = 2 - 10$. We will present the mathematical framework used to estimate the electron density $\rho(\mathbf{r})$, and perform a Fourier transform of the wavefunctions to obtain $\phi(\mathbf{k})$, from which the electron density in the momentum space, $n(\mathbf{k})$ is derived. With both $\rho(\mathbf{r})$ and $n(\mathbf{k})$ calculated, we computed the Shannon entropy in position and momentum spaces. In Chapter 3, we present the results of our analysis and compare them with the findings of Chatzisavvas et al. (2005), upon which this work is based.

Chapter 2

Methodology

Roothaan-Hartree-Fock algorithm is a Self-Consistent-Field (SCF) procedure that involves linear algebra, making its implementation in Python challenging. Luckily for us, there is a Python module, named PySCF which can help us solve the Roothaan algorithm and obtain the atomic orbitals. Our methodology from this point becomes easier.

According to Chatzisavvas et al. (2005) the wavefunction of an atom is decomposed in the form:

$$\phi(\mathbf{r}) = \phi_{nlm}(\mathbf{r}) = R_{nl}(r)Y_{lm}(\Omega_r) \tag{2.1}$$

where the radial atomic orbitals R_{nl} are expressed as a finite superposition of primitive radial functions:

$$R_{nl}(r) = \sum_j C_{jnl}S_{jl}(r) \tag{2.2}$$

C_{jnl} are the coefficients of the atomic orbital and S_{jl} are the basis functions, which normally are taken as a Slater-type orbital set.

In the present work, for simplicity reasons, we will consider that the wavefunctions $\phi(\mathbf{r})$ correspond the radial atomic orbitals $R_{nl}(r)$, ignoring the angular dependence.

Using PySCF, we can build the atomic system of interest as a Python object. This object allow is to access the basis functions in equation 2.2, which, in our case are not Slayter-type orbitals (STOs) but Gaussian-type orbitals (GTOs) -spherical Gaussian approximations that represent STOs in quantum chemistry. The GTOs can be evaluated on a grid of radial points, r, giving us an array representing the basis function values at each point in space. We can also use the PySCF object to perform a Roothaan-Hartree-Fock (RHF) calculation, implemented already in the module, which can provide us with the *molecular orbital (MO)* coefficients -in our case atomic coefficients- in equation 2.2.

As a result, we obtain:

- **A basis matrix** of shape (N_r, N_b), where N_r is the number of radial grid points (e.g.,500) and N_b is the number of basis functions (e.g., 5 for orbitals such as $1s,\ 2s,\ 2p_x,\ 2p_y,\ 2p_z$).
- **A coefficient matrix** of shape (N_b, N_b) where each column correspond to an orbital and each row is the weight that express each orbital as a linear compilation of the basis functions.

By computing the matrix product:

$$orb = \mathbf{np}.\mathbf{dot}(\ basis,\ C\ ) \tag{2.3}$$

we obtain an array *orb* with a shape of (N_r, N_b) where each column is an atomic orbital $\phi(\mathbf{r})$ evaluated over the grid of radial points. Mathematically, we calculate equation 2.2 where we have assumed that the radial orbitals R_{nl}, in our case, are the general atomic orbitals, $\phi(\mathbf{r})$.

Let us assume, for example, that we want to obtain the radial wavefunctions of an atom with orbitals like $1s,\ 2s,\ 2p_x,\ 2p_y,\ 2p_z$. This atom has five orbitals, so the number of basis functions is $N_b = 5$. By defining a radial grid with $N_r = 500$, PySCF can provide us with the numerical values of each basis function evaluated at each grid point. As a result, the basis matrix, is a Numpy array with a shape of $(500, 5)$, where each row corresponds to a radial point and each column to a basis function. The coefficient matrix, on the other hand, has a shape of $(5, 5)$ where each column represents the expansion coefficients of one orbital in terms of the basis functions. Using *np.dot*, we construct the orbitals by combining the basis function values at each radial point with the corresponding coefficients. For example the first radial point of the $1s$ wavefunction is a weighted sum of the five basis functions at that point, using the coefficients in the first column of the matrix. Each wavefunction at each radial point is therefore computed using:

$$\phi_j = C_{1,i} \cdot b_1(j) + C_{2,i} \cdot b_2(j)\ + ... +\ C_{n,i} \cdot b_5(j) \tag{2.4}$$

where $b_k(j)$ is the value of the k^{th} basis function at the j^{th} radial point, ϕ_j is the j^{th} value of the wavefunction corresponding to orbital i, $C_{k,i}$ is the coefficient for the k^{th} basis function in the i^{th} orbital.

Having obtain the radial wavefunctions, we can compute the electron density from equation:

$$\rho(\mathbf{r}) = \sum_i \phi_i^*(\mathbf{r})\phi_i(\mathbf{r}) = \sum_i |\phi_i(\mathbf{r})|^2 \tag{2.5}$$

where ϕ_i is the value of the wavefunction in the i^{th} orbital. This means that the first radial value of the density is the sum of the first radial values of the wavefunctions.

As a result, we obtain the electron density evaluated over the radial grid. Then using equation (Chatzisavvas et al.; 2005):

$$\tilde{R}_{nl}(k) = 4\pi \int_0^\infty r^2 R_{nl}(r) j_l(kr) dr \tag{2.6}$$

we can obtain the wavefunctions in the momentum space k. $j_l(kr)$ is a spherical Bessel functions, which we can obtain using the Python library Scipy. Using the values of the wavefunctions in the momentum space, we can compute the electron density $n(\mathbf{k})$ as:

$$n(\mathbf{k}) = \sum_i \phi_i^*(\mathbf{k})\phi_i(\mathbf{k}) = \sum_i |\phi_i(\mathbf{k})|^2 \tag{2.7}$$

where again $\phi(\mathbf{k})$ would correspond to $R_{nl}(\mathbf{k})$. Now, we have everything we need to compute the Shannon entropy in both spaces. The Shannon entropy is calculated as:

$$S_r = -4\pi r^2 \cdot \int \rho(\mathbf{r}) \cdot ln\rho(\mathbf{r})\, dr \tag{2.8}$$

$$S_k = -4\pi k^2 \cdot \int \rho(\mathbf{k}) \cdot ln\rho(\mathbf{k})\, dr \tag{2.9}$$

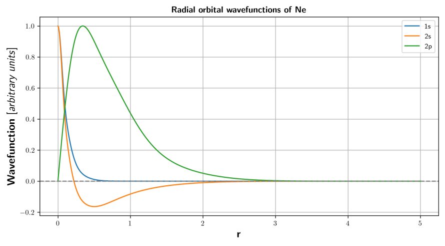
We computed these integrals numerically, using the trapezoidal rule and as a result, for each atom of interest, we obtained its Shannon entropy value.

Finally, we adopted the total entropy ($S_r + S_k$) values from Chatzisavvas et al. (2005) and performed a fitting to obtain the equation that describes the relation between S and z. From a visual inspection of the plot, presented in Chapter 3, it was clear that a logarithmic scale in the data, would reveal a linear dependence. Thus, the curve fitting would became a linear fitting of the form: $logy = logA + \beta \cdot logx$ and we only needed to obtain the slope and intercept to fit an equation of the form: $S = A \cdot e^{\beta \cdot logZ}$.

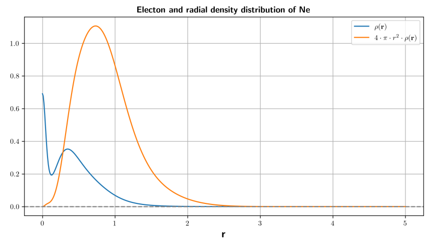
Chapter 3

Results

Following the methodology described in Chapter 2, we computed the radial wavefunctions for atoms with atomic numbers in the range $Z = 2 - 10$, from helium (He) to neon (Ne). Due to space constraints in this report, we present in Figure 3.1 a representative example of the results: the radial wavefunctions of the $1s, 2s$ and $2p$ orbitals for the neon atom along with the electron density and radial probability density.



(a)



(b)

Figure 3.1: (a) The radial orbitals of Ne. While the $1s$ and $2s$ orbitals are spherical and thus lack angular dependence, the $2p$ orbital has an angular momentum quantum number $l = 1$, which introduces angular dependence. However, in the present analysis, only the radial part of the $2p$ orbital is considered. (b) The electron density as described in equation 2.4, and the radial probability density given by $4\pi r^2 \cdot \rho$.

It must be mentioned here that the values of the electron density, both in position and momentum space, have been normalized so the integrals $4\pi \int r^2 \cdot \rho \, dr$ and $4\pi \int k^2 \cdot n \, dk$ are unity.

Now that we have computed the electron densities of the atoms, we can evaluate their Shannon entropies as functions of the atomic number Z. In Figure 3.2, we present the entropy values in both position and momentum spaces, along with the corresponding results reported by Chatzisavvas et al. (2005). Our analysis reproduces the general trend observed in the literature but with a noticeable deviation from the reference values. This discrepancy is most likely due to our choice of basis set as a Gaussian-type orbital (GTO) which is an approximation of the Slater-type orbital (STOs), which represents better the true atomic orbitals. So, the limited accuracy of GTOs may lead to deviations in the computed wavefunctions and, therefore, in the resulting entropy values.

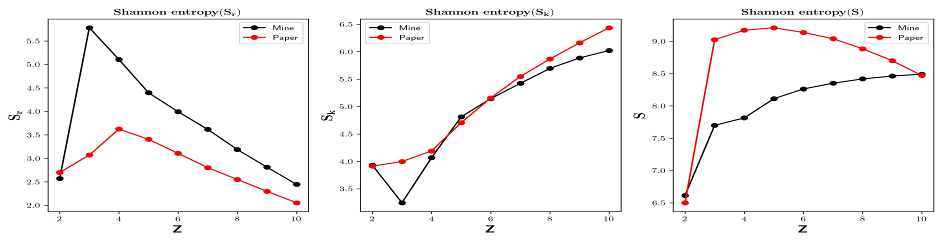


Figure 3.2: The Shannon entropies as functions of Z. The third plot corresponds to the total entropy, defined as $S_r + S_k$.

Adopting the values of the total entropy S from Chatzisavvas et al. (2005) in the atomic number range of $Z = 1 - 54$, we performed a fitting in the data, Figure 3.3. As a result, we obtained the equation that describes their relation: $S = A \cdot e^{\beta \cdot logxZ}$, where $A = 1.884 \pm 0.007$ and $\beta = 0.119 \pm 0.002$. The uncertainties were accessed using the Scipy fitting library, **linregress**.

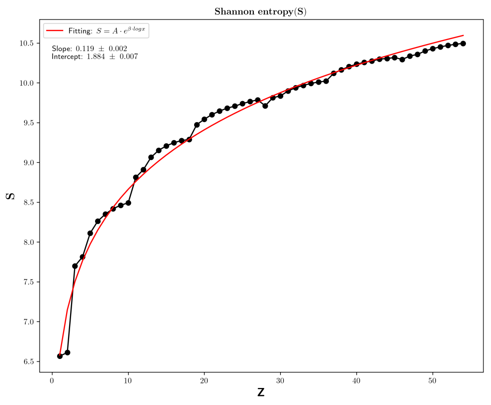


Figure 3.3: The total entropy as a function of Z and the fitted function that was obtained using Scipy in Python.

Bibliography

Bunge, C., Barrientos, J. and Bunge, A. (1993). Roothaan-hartree-fock ground-state atomic wave functions: Slater-type orbital expansions and expectation values for z = 2-54, *Atomic Data and Nuclear Data Tables* **53**(1): 113–162.
URL: *https://www.sciencedirect.com/science/article/pii/S0092640X8371003X*

Chatzisavvas, K. C., Moustakidis, C. C. and Panos, C. P. (2005). Information entropy, information distances, and complexity in atoms, *The Journal of Chemical Physics* **123**(17): 174111.
URL: *https://doi.org/10.1063/1.2121610*