Hydrogen_Atom_Ground_State (2)

January 4, 2024

#beria ayşenur can 090200705

a) The probability density for finding an electron in a hydrogen atom in the ground state, which involves plotting a graph of the probability density as a function of radial distance (r/a0).

b) Determining the most probable location of an electron in the ground state, which would likely involve finding the maximum of the probability density function.

c) Finding the expected value of the position for the ground state wave function, and indicating this on the graph.

[51]:
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.constants import hbar, m_e ,pi
from scipy.optimize import minimize_scalar
from scipy.integrate import quad
```

#a)

$$\psi_{100}(r) = \frac{1}{\sqrt{\pi \cdot a_0^3}} \cdot e^{-\frac{r}{a_0}} \tag{1}$$

$$dP = |\psi|^2 \cdot 4\pi \cdot r^2 \cdot dr \tag{2}$$

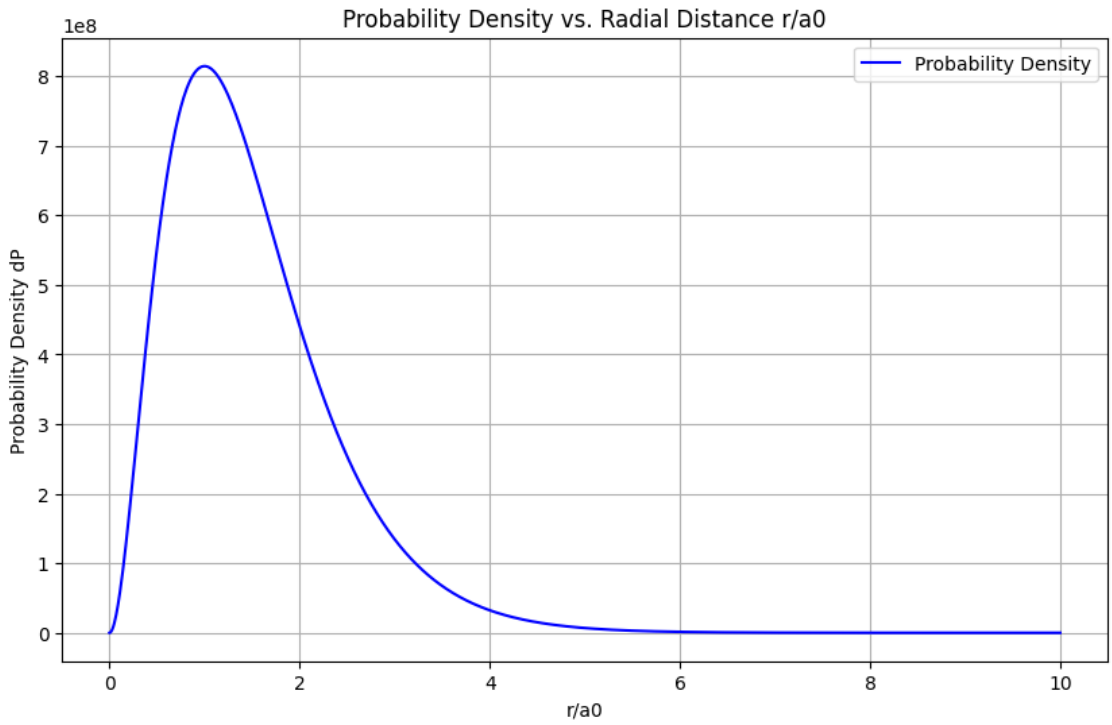
$$dP/dr = |\frac{1}{\sqrt{\pi \cdot a_0^3}} \cdot e^{-\frac{r}{a_0}}|^2 \cdot 4\pi \cdot r^2 \tag{3}$$

$$dP/dr = |\frac{1}{\sqrt{\pi \cdot a_0^3}} \cdot e^{-\frac{r}{a_0}}|^2 \cdot 4\pi \cdot r^2 \tag{4}$$

[48]:
```
a0 = 0.0529e-9 # bohr radius


# wave function
def psi_100(r):
    return (1 / np.sqrt(np.pi * a0**3)) * np.exp(-r / a0)
```

```
# dP
def probability_density(r):
    return np.abs(psi_100(r))**2 * r**2

# from 0 to 10 times the Bohr radius
r = np.linspace(0, 10*a0, 1000)

dP = probability_density(r)
```

[49]:
```
plt.figure(figsize=(10, 6))
plt.plot(r/a0, dP, label='Probability Density', color='blue')
plt.xlabel('r/a0')
plt.ylabel('Probability Density dP')
plt.title('Probability Density vs. Radial Distance r/a0')
plt.legend()
plt.grid(True)
plt.show()
```



#b)

$$\frac{d}{dr}\left(\frac{dP}{dr}\right) = 0 \tag{5}$$

$$\frac{d}{dr}\left(\frac{dP}{dr}\right) = \frac{4}{a_0^3} \cdot (2r \cdot e^{-\frac{2r}{a_0}} \cdot (1 - \frac{r}{a_0})) \tag{6}$$

$$\psi = \frac{1}{\sqrt{\pi a_0^3}} e^{-r/a_0} \tag{7}$$

$$P = |\psi|^2 4\pi r^2 dr = \frac{4}{a_0^3} e^{-2r/a_0} r^2 dr = p(r)dr \tag{8}$$

$$\frac{dp}{dr} = \frac{4}{a_0^3}\left[2re^{-2r/a_0} + r^2\left(-\frac{2}{a_0}\right)e^{-2r/a_0}\right] = \frac{8}{a_0^3} e^{-2r/a_0}\left(1 - \frac{r}{a_0}\right) = 0 \Rightarrow r = a_0. \tag{9}$$

Denklem:

$$\frac{dp}{dr} = \frac{8}{a_0^4} e^{-2r/a_0}\left(r - \frac{r^2}{a_0}\right) = 0 \tag{10}$$

Denklemin köklerini bulmak için, terimleri sıfıra eşitleyip (r) için çözeriz:

$$r - \frac{r^2}{a_0} = 0 \tag{11}$$

Bu denklemi çözersek:

$$r(1 - \frac{r}{a_0}) = 0 \tag{12}$$

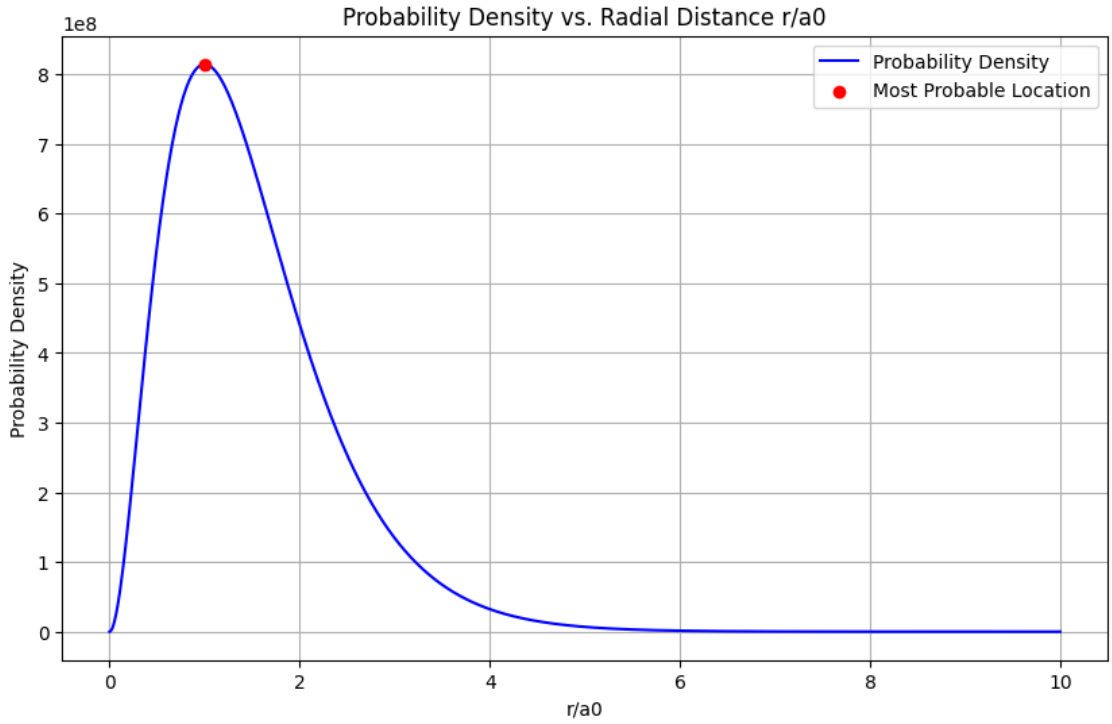
Bu durumda, ya $(r = 0)$ ya da $(r = a_0)$olmalıdır. $(r = 0)$, fiziksel bir anlam ifade etmeyen bir çözümdür çünkü bu noktada olasılık yoğunluğu sıfırdır. Geriye kalan fiziksel anlamı olan tek çözüm $(r = a_0)$ olmalıdır. Yani, en olası konum (a_0)'ya eşittir. Bu, teorik olarak beklenen sonuçtur.

[61]:
```
r_max = a0
r_max_normalized = r_max / a0  # Normalizing the most probable location by Bohr␣
↪radius

r_max, r_max_normalized
```

[61]:
```
(5.29e-11, 1.0)
```

[62]:
```
# Normalized radial values for plotting
r_values = np.linspace(0, 10*a0, 1000)
dP_values = probability_density(r_values)

# Plotting the probability density and marking the most probable location
plt.figure(figsize=(10, 6))
plt.plot(r_values / a0, dP_values, label='Probability Density', color='blue')
plt.scatter(r_max / a0, probability_density(r_max), color='red', zorder=5,␣
↪label='Most Probable Location')
```

```
plt.xlabel('r/a0')
plt.ylabel('Probability Density')
plt.title('Probability Density vs. Radial Distance r/a0')
plt.legend()
plt.grid(True)
plt.show()
```

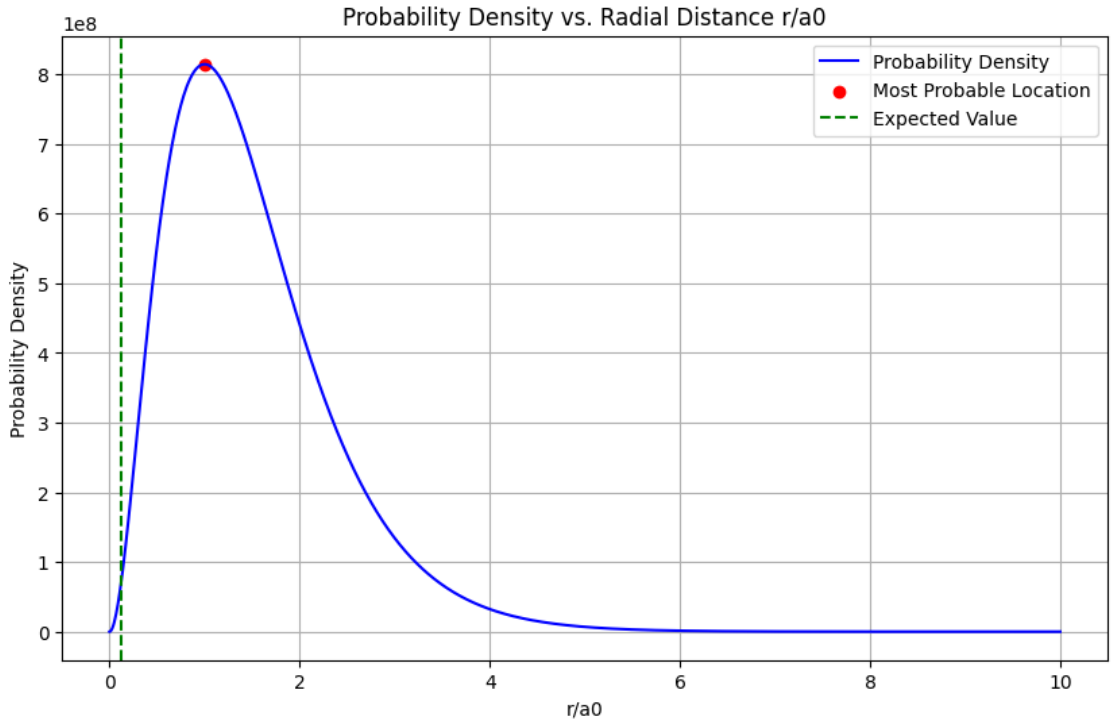


#c)

$$\langle r \rangle = \int_0^\infty r \cdot dP\, dr \tag{13}$$

[64]:
```
# Define the integrand for the expected value of r
def integrand_for_expected_r(r):
    return r * probability_density(r)


# Compute the expected value of r (mean distance) using numerical integration
# We'll integrate from 0 to a point where the wave function is effectively zero
upper_limit = 20 * a0  # Integration upper limit
expected_r, _ = quad(integrand_for_expected_r, 0, upper_limit)

# Normalizing the expected value by the Bohr radius
```

```
expected_r_normalized = expected_r / a0

# Add the expected value to the plot
plt.figure(figsize=(10, 6))
plt.plot(r_values / a0, dP_values, label='Probability Density', color='blue')
plt.scatter(r_max / a0, probability_density(r_max), color='red', label='Most␣
 ↪Probable Location')
plt.axvline(x=expected_r_normalized, color='green', linestyle='--',␣
 ↪label='Expected Value')
plt.xlabel('r/a0')
plt.ylabel('Probability Density')
plt.title('Probability Density vs. Radial Distance r/a0')
plt.legend()
plt.grid(True)
plt.show()

expected_r_normalized, expected_r
```



[64]: (0.11936620731891535, 6.314472367170622e-12)