CMP Assignment-1

Karmukilan Somasundaram

January 2026

1 Density of states plots

1.1 Linear chain of atoms

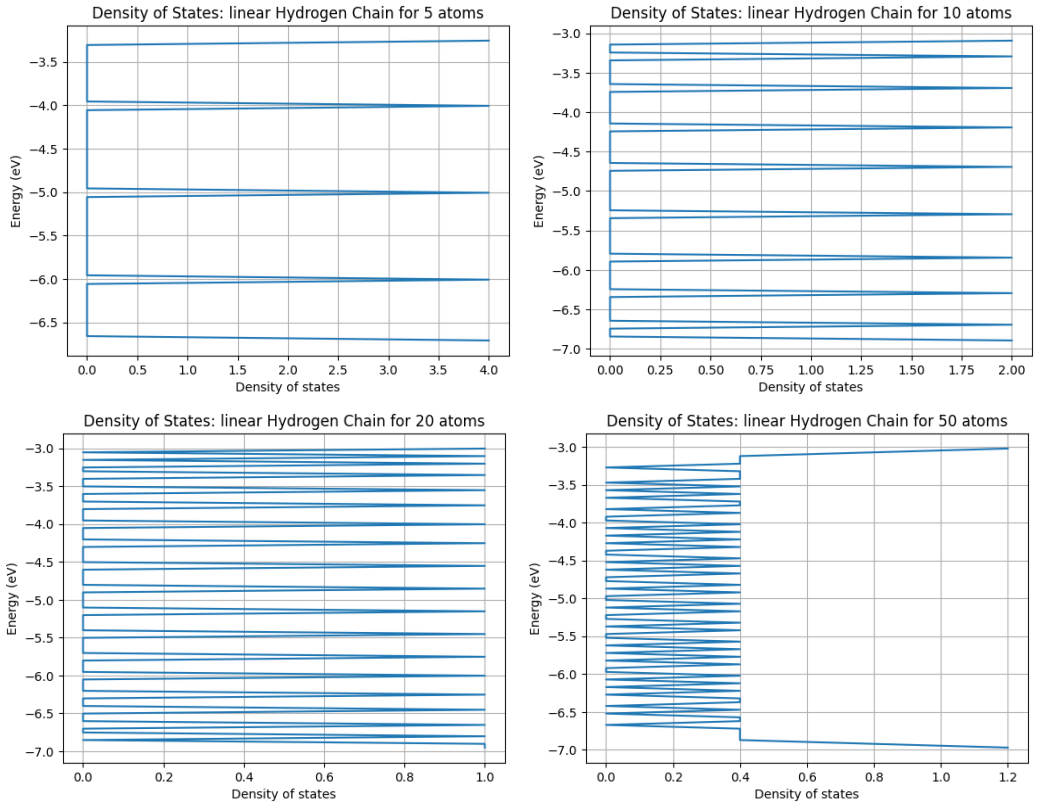

Figure 1: Density of states for linear chain, small N

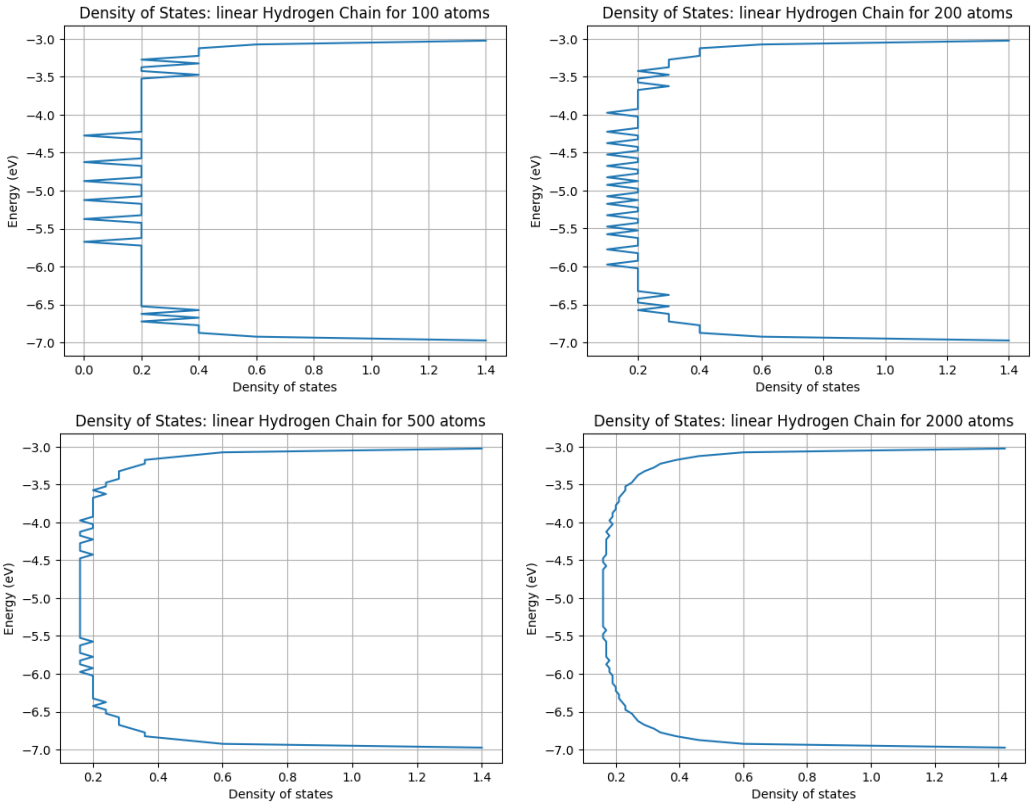

Figure 2: Density of states for linear chain, large N

1.2 Periodic boundary condition

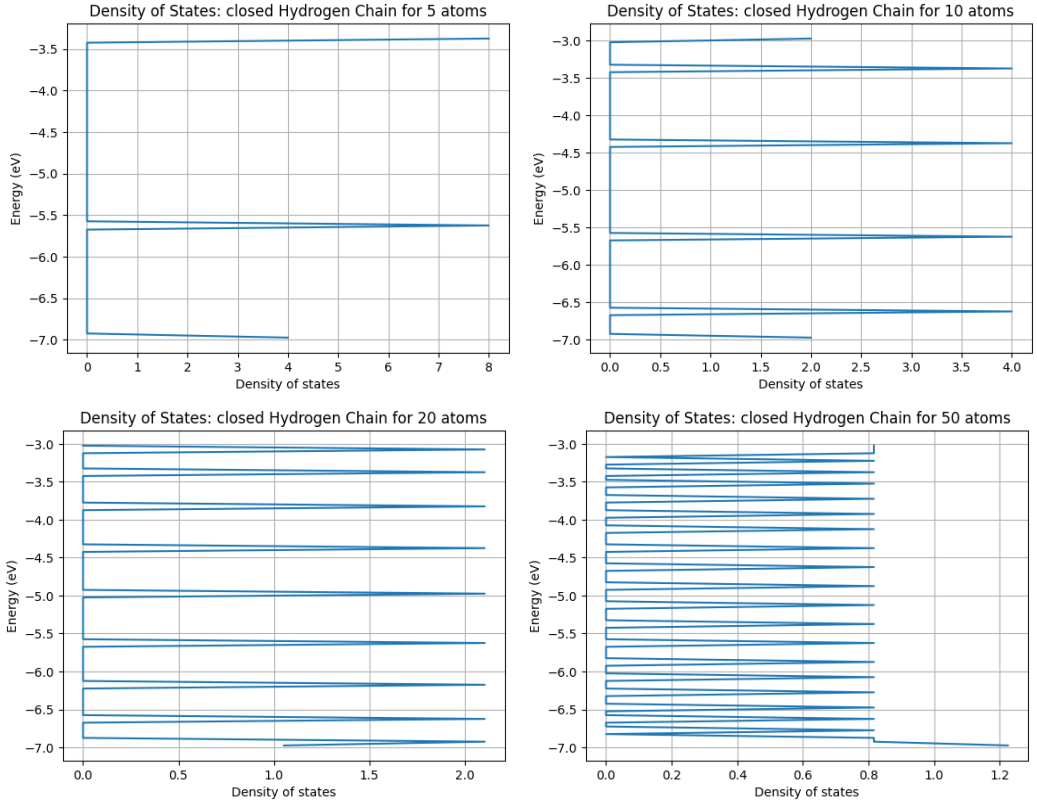

Figure 3: Density of states for closed chain, small N

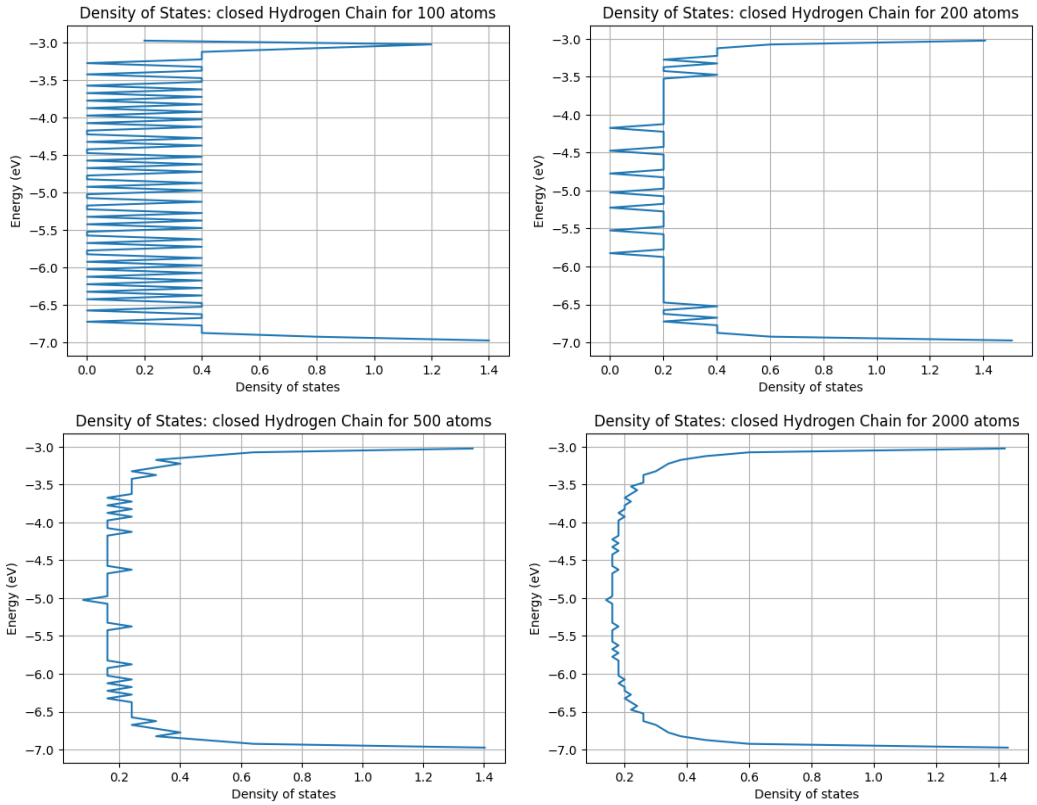

Figure 4: Density of states for closed chain, large N

1.3 Dimerization plots

For a chain of dimers, we have two hopping parameters defined as follows

$$\beta_1 = \beta(1+\delta)$$

$$\beta_2 = \beta(1-\delta)$$

where δ is the strength of dimerization

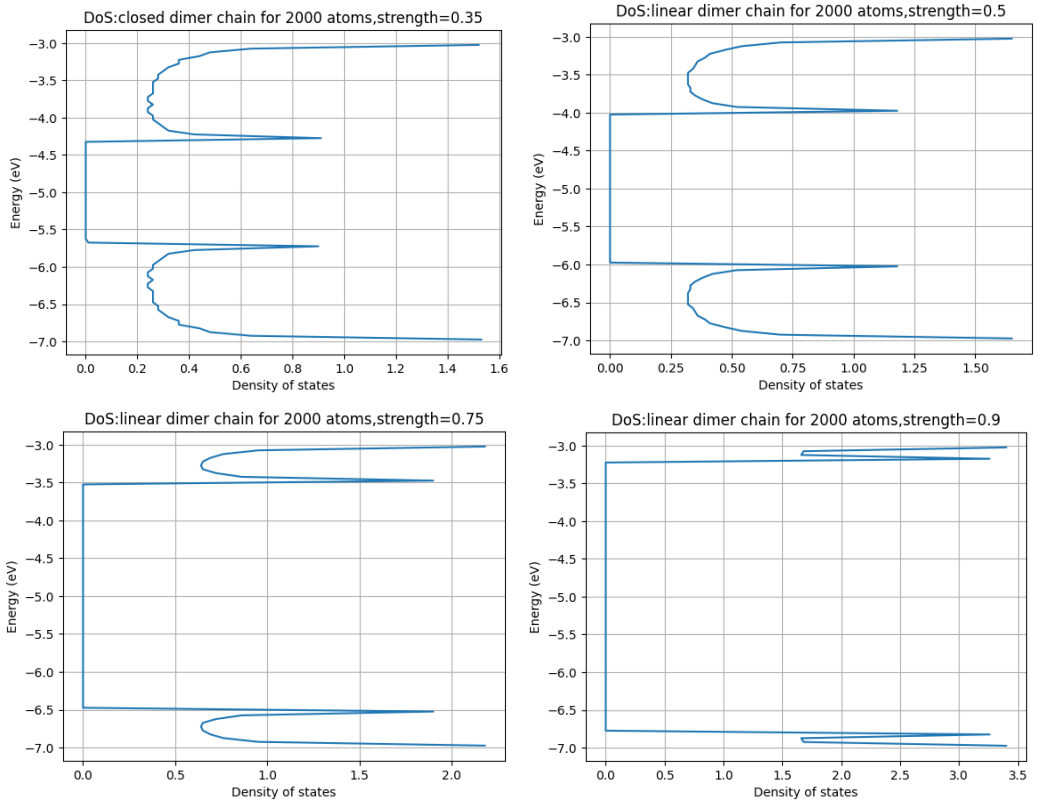

Figure 5: Density of states for Different strength of Dimerization δ

1.4 Eigenstate and fermi-energy plots

The ground state wave-function resembles a gaussian and the subsequent energy eigenstates are oscillatory in nature this implies the first ground state is essentially just the ground state wave function of the central atom and the subsequent energy eigen-states result from a complicated linear combination of the ground state gaussians.

We can observe that the fermi-level is same as E. One can repeat the same process for different E and verify the previously mentioned claim. This is because the Distribution of Eigen-values is

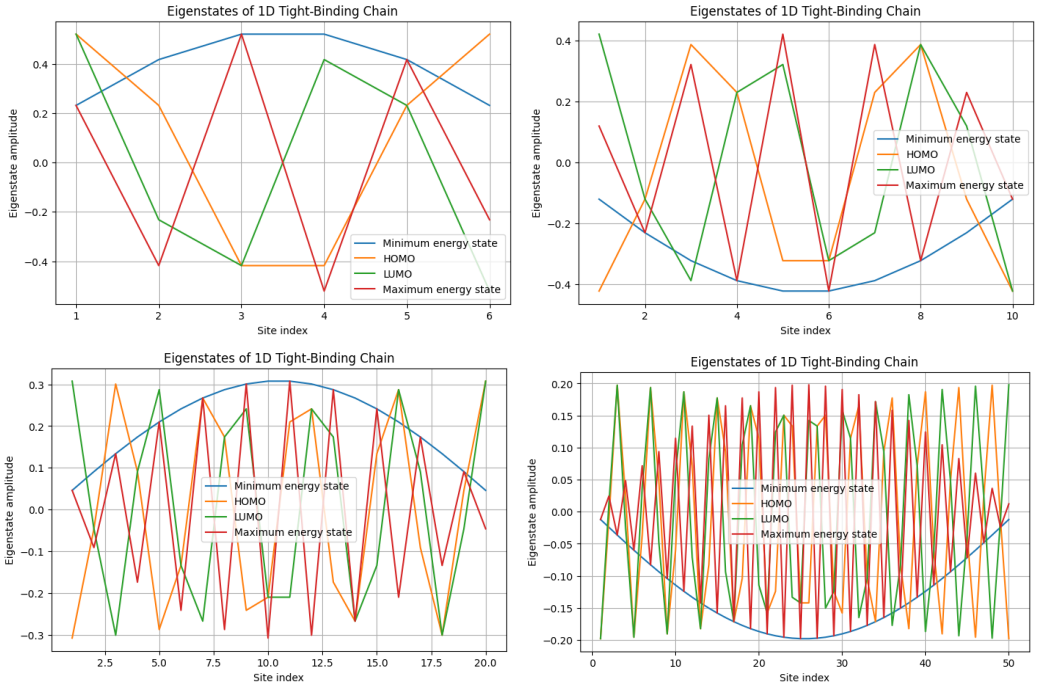

Figure 6: Eigen-state plot for small number of sites(N=5,10,20,50)

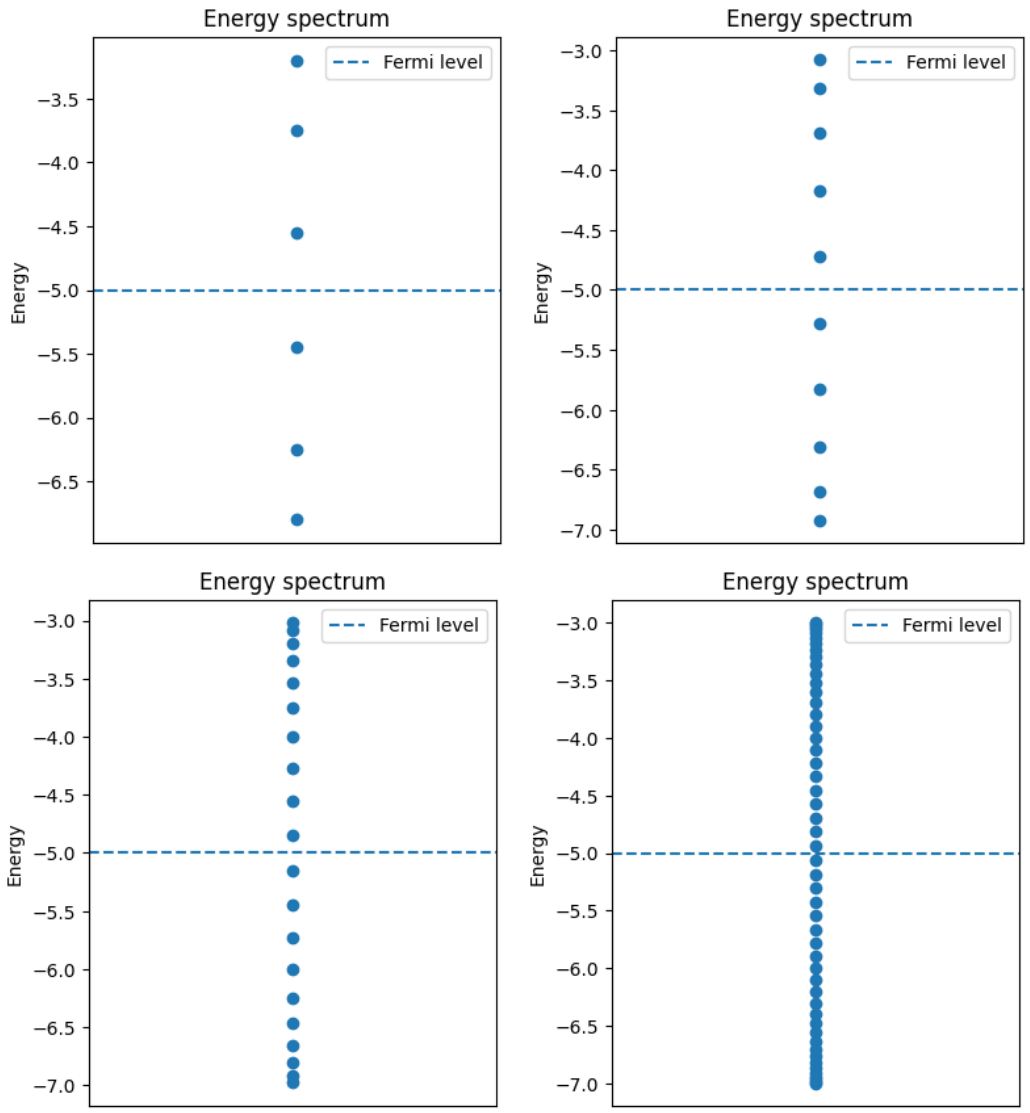

Figure 7: Fermi level for small number of sites(N=5,10,20,50)

symmetric about E. Also the range of eigen-values is directly proportional to the hopping parameter beta for a linear chain of equidistant atoms.To be more precise $E_{max} - E_{min} = -4\beta$.

2 Python Codes

Listing 1: Density of states

```
import numpy as np
import matplotlib.pyplot as plt

# Number of atoms
n =2000

# Parameters
E0 = -5.0
beta = -1.0
delta = 0.35   # dimerization strength (play with this)

beta_1 = beta * (1 + delta)
beta_2 = beta * (1 - delta)

# Build Hamiltonian
H = np.zeros((n, n))
np.fill_diagonal(H, E0)

for i in range(n - 1):
    if i % 2 == 0:
        H[i, i + 1] = beta_1
        H[i + 1, i] = beta_1
    else:
        H[i, i + 1] = beta_2
        H[i + 1, i] = beta_2
H[0, n - 1] = beta_2
H[n - 1, 0] = beta_2

# Eigenvalues
eigenvalues = np.linalg.eigvalsh(H)
# ---- BIN WIDTH CONTROL ----
bin_width = 0.05  # eV (change this freely)

E_min = eigenvalues.min()
E_max = eigenvalues.max()

bins = np.arange(E_min, E_max + bin_width, bin_width)

# Histogram (density=True gives DOS)
dos, bin_edges = np.histogram(eigenvalues, bins=bins, density=True)

# Bin centers
```

```
energy_centers = 0.5 * (bin_edges[:-1] + bin_edges[1:])

# Plot DOS
plt.figure()
plt.plot(dos,energy_centers)
plt.ylabel("Energy (eV)")
plt.xlabel("Density of states")
plt.title(f"DoS: closed dimer chain for {n} atoms, strength={delta}")
plt.grid(True)
plt.show()
```

Listing 2: Tight-binding chain with Fermi level and eigenstates

```
import numpy as np
import matplotlib.pyplot as plt

# Parameters
N = 40
epsilon = -5.0
beta = -1.0

# Hamiltonian
H = np.zeros((N, N))
for i in range(N):
    H[i, i] = epsilon
    if i < N - 1:
        H[i, i+1] = beta
        H[i+1, i] = beta

# Diagonalization
eigvals, eigvecs = np.linalg.eigh(H)

# Spin-degenerate filling (half filling)
n_occ = N // 2
E_F = 0.5 * (eigvals[n_occ - 1] + eigvals[n_occ])

print("Fermi energy =", E_F)

# Plot eigenstates
sites = np.arange(1, N + 1)

plt.plot(sites, eigvecs[:, 0], label="Lowest energy")
plt.plot(sites, eigvecs[:, n_occ - 1], label="HOMO")
plt.plot(sites, eigvecs[:, n_occ], label="LUMO")
plt.plot(sites, eigvecs[:, -1], label="Highest energy")

plt.xlabel("Site index")
plt.ylabel("Eigenstate amplitude")
plt.legend()
plt.show()
```