CMP project 3

Karmukilan Somasundaram

February 2026

1 Introduction

Using the Central equation we obtained from Bloch's Theorem. We will try to obtain the plots for small perturbations of the periodic potential.Note to efficiently solve for the simultaneous system of linear equations, we truncate the fourier series expansion of ψ_k. $V_G = V_{-G}$ is assumed for convenience. We will in

2 Plots

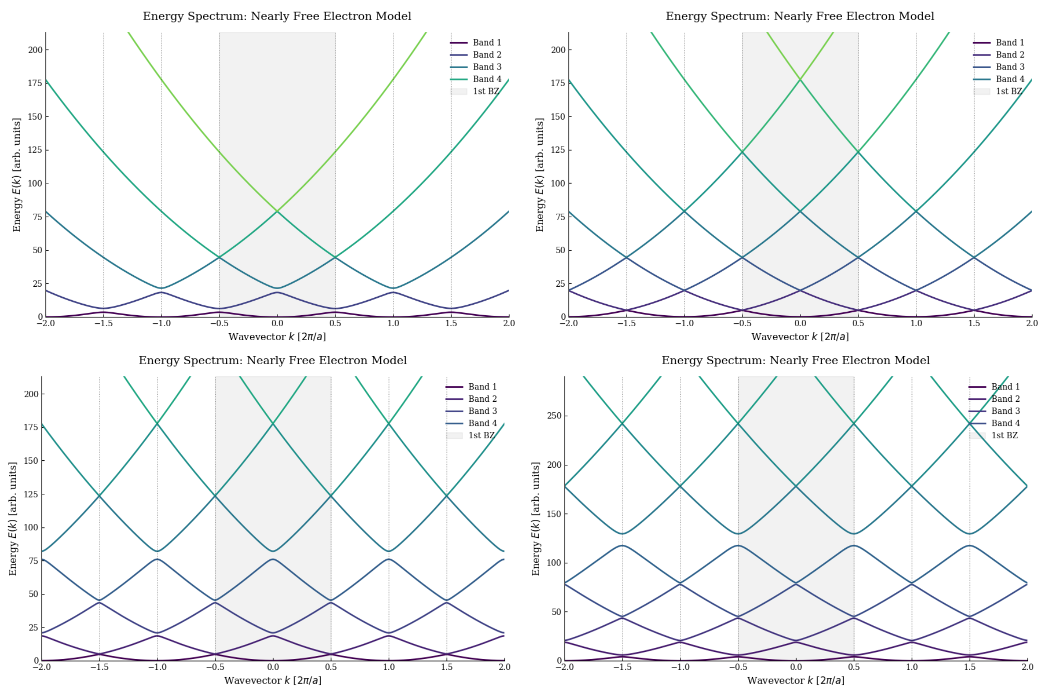

Figure 1: a.(1.5,1.5), b.(0.1,0.1,0.1), c.(0.2,1.2,1,3), d.(1,1,1,1,6)

From the Plot we can see that at the edges and the middle of the brillioun Zones the slopes of the spectrum is zero. This could be analytically derived by considering mixing of plane-waves in the junction of two bands.

3 Python Code

Listing 1: Nearly Free Electron Band structure

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

# Set plotting style for a "scientific" look
plt.rcParams.update({
    "text.usetex": False, # Set to True if you have LaTeX installed
    "font.family": "serif",
    "axes.labelsize": 12,
    "xtick.direction": "in",
    "ytick.direction": "in",
})

def plot_aesthetic_nfe():
    a = 1.0
    G0 = 2 * np.pi / a
    k_min, k_max = -2, 2
    num_k_points = 1000 # Smoother curves

    n = int(input("Truncation␣parameter␣n:␣"))
    v_input = input(f"Enter␣{n}␣values␣for␣V_G␣(commas):␣")
    v_list = [float(x.strip()) for x in v_input.split(',')]
    while len(v_list) < 2*n: v_list.append(0.0) # Pad for safety

    g_indices = np.arange(-n, n + 1)
    num_basis = len(g_indices)
    k_vals = np.linspace(k_min * G0, k_max * G0, num_k_points)

    energies_grid = []

    for k in k_vals:
        H = np.zeros((num_basis, num_basis))
        for i in range(num_basis):
            for j in range(num_basis):
                if i == j:
                    H[i, j] = 0.5 * (k - g_indices[i] * G0)**2
                else:
                    m = abs(g_indices[i] - g_indices[j])
                    H[i, j] = v_list[m-1] if m <= len(v_list) else 0

        energies_grid.append(np.linalg.eigvalsh(H))

```

```python
    energies_grid = np.array(energies_grid)

    # --- Plotting ---
    fig, ax = plt.subplots(figsize=(9, 6), dpi=100)

    # Use a colormap for the bands
    colors = plt.cm.viridis(np.linspace(0, 0.8, num_basis))

    for i in range(num_basis):
        ax.plot(k_vals / G0, energies_grid[:, i], color=colors[i], lw=2, label=f'Band␣{i+1}' if i < 4 else "")

    # Shading the First Brillouin Zone
    ax.axvspan(-0.5, 0.5, color='gray', alpha=0.1, label='1st␣BZ')

    # Vertical lines at BZ boundaries
    for boundary in [-1.5, -1.0, -0.5  , 0.5, 1.0, 1.5]:
        ax.axvline(x=boundary, color='black', linestyle=':', lw=0.8, alpha=0.5)

    ax.set_title(r"Energy␣Spectrum:␣Nearly␣Free␣Electron␣Model", fontsize=14, pad=15)
    ax.set_xlabel(r"Wavevector␣$k$␣[$2\pi/a$]", fontsize=12)
    ax.set_ylabel(r"Energy␣$E(k)$␣[arb.␣units]", fontsize=12)

    ax.set_xlim(k_min, k_max)
    ax.set_ylim(0, energies_grid[:, n+1].max() * 1.2) # Focus on the first few gaps

    ax.legend(loc='upper␣right', frameon=False, fontsize=10)
    ax.spines['top'].set_visible(False)
    ax.spines['right'].set_visible(False)

    plt.tight_layout()
    plt.show()

plot_aesthetic_nfe()
```