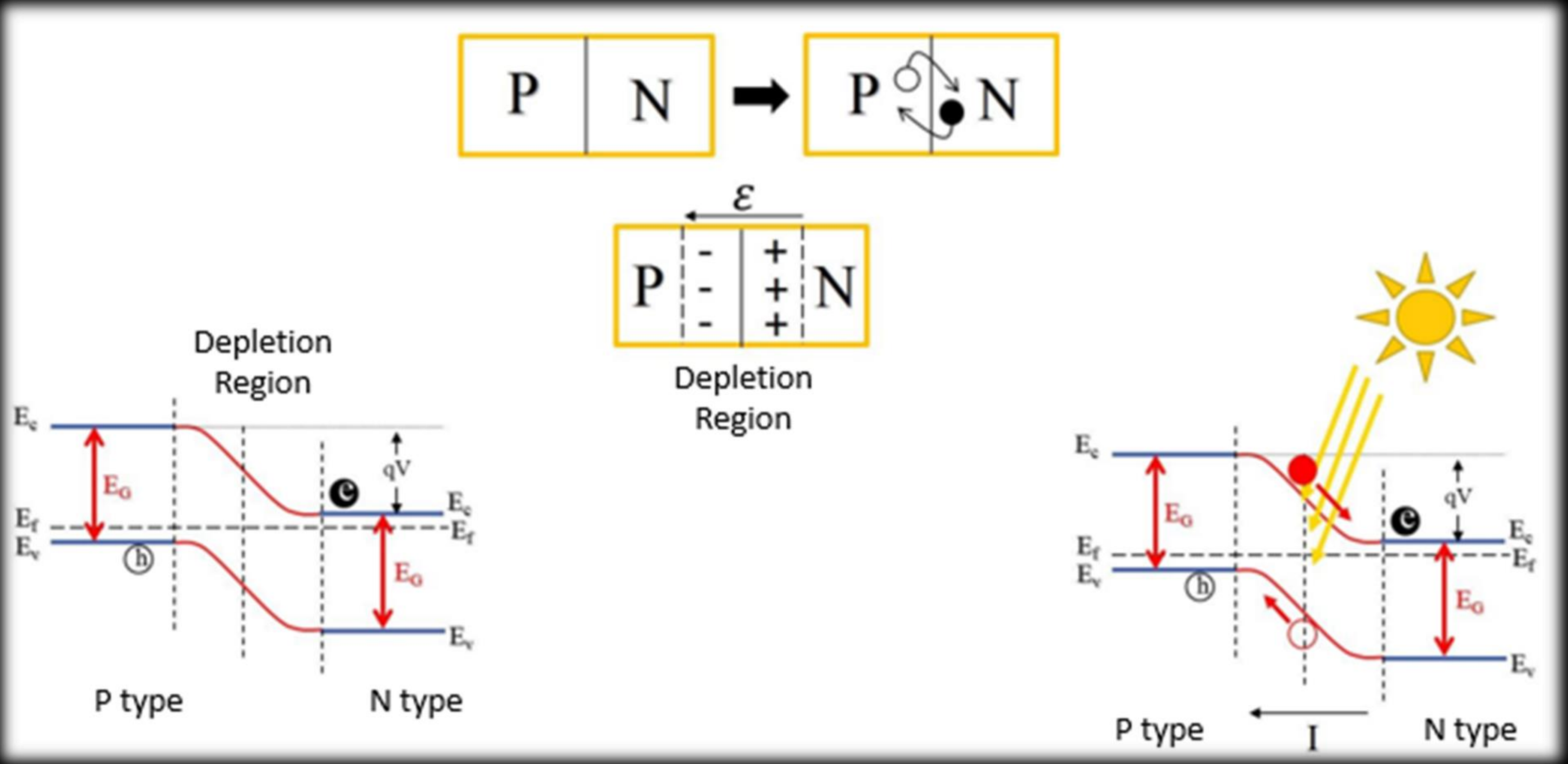
QUANTUM DOT & MORE EFFICIENT SOLAR CELL

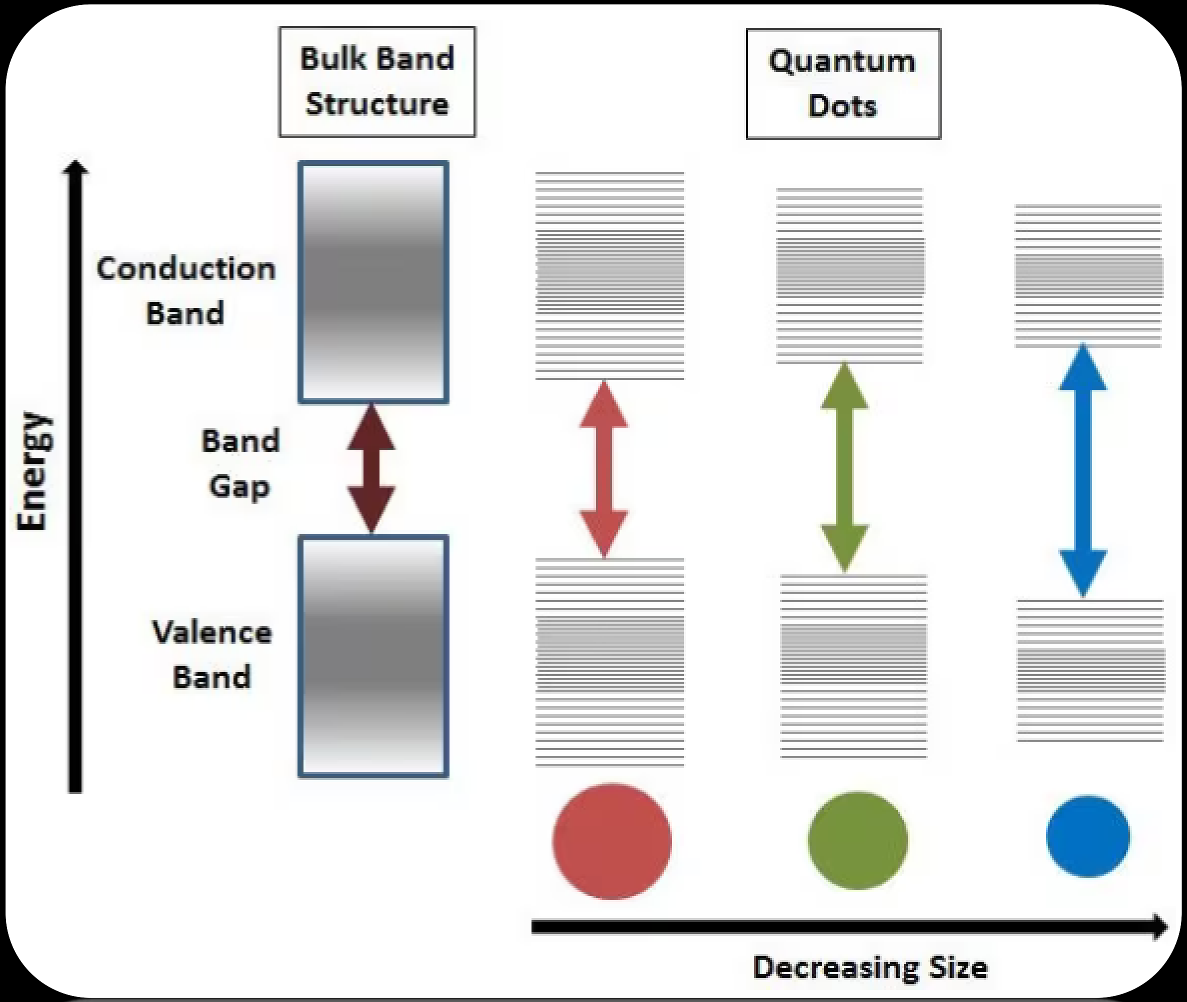
Fateme Yazdani , Marzie Nadi , Anisa Sadeghi, Nima Molatefi
Supervisor: Dr. Sahebsara
Teaching Assistant: Mohamad Amini

SOLAR CELL

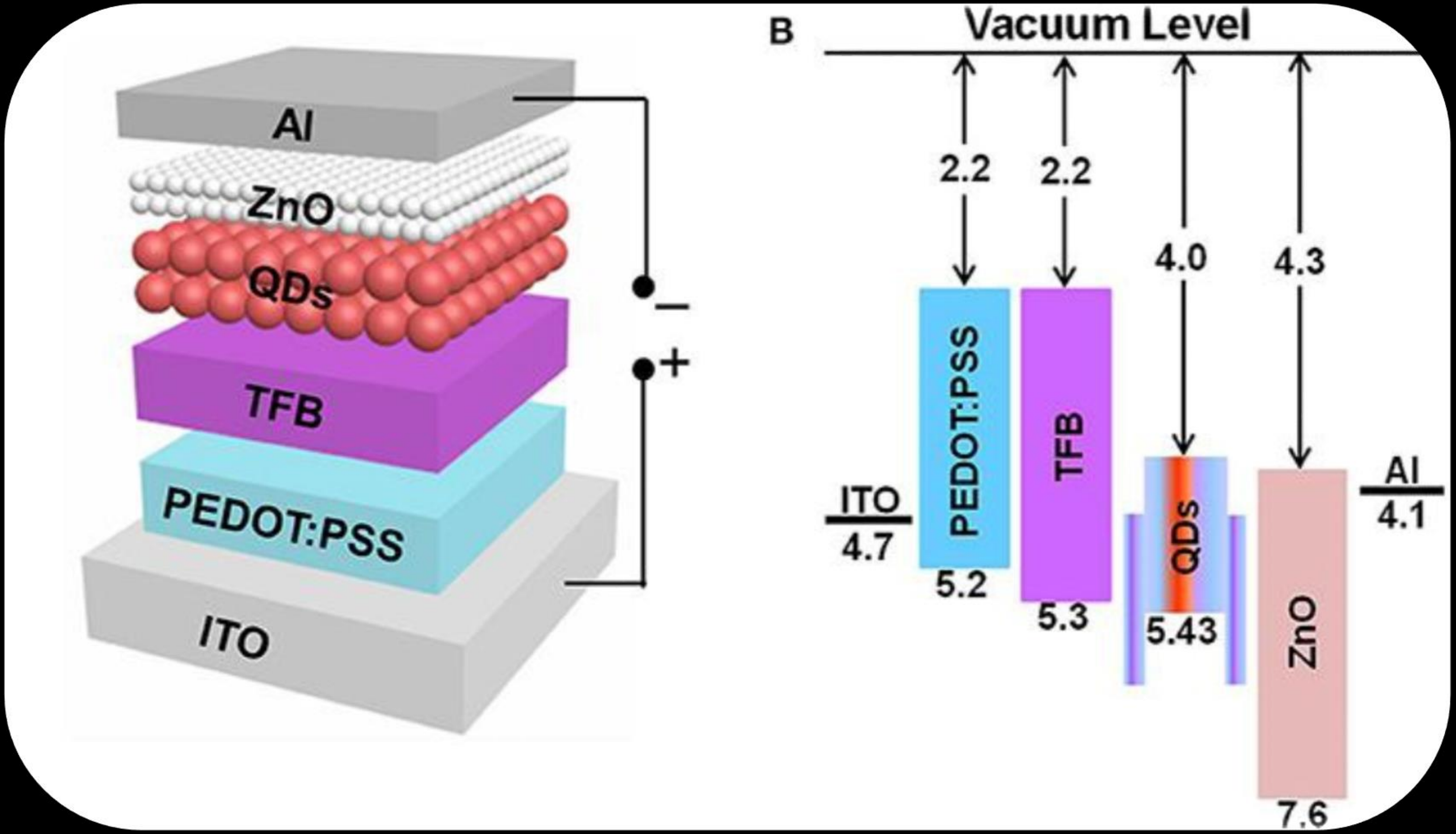


P
N
P
N
ε
P
-
-
-
+
+
+
N
Depletion
Region
Depletion
Region
E_c
E_G
E_f
E_v
h
e
qV
E_c
E_f
E_G
E_v
P type
N type
E_c
E_G
E_f
E_v
h
e
qV
E_c
E_f
E_G
E_v
P type
I
N type

WHAT IS QUANTUM DOT?



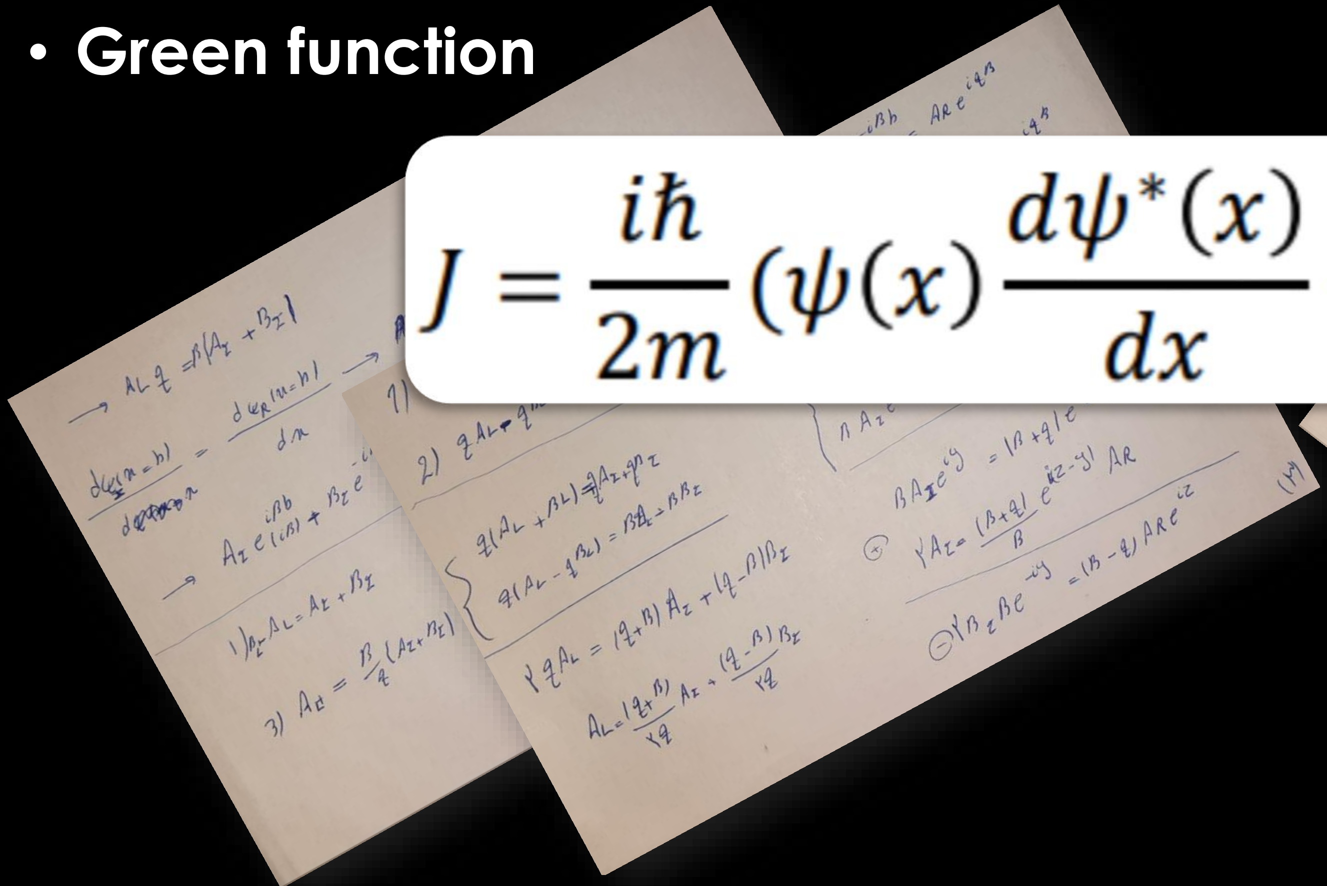
WHY DO WE NEED QUANTUM DOTS?

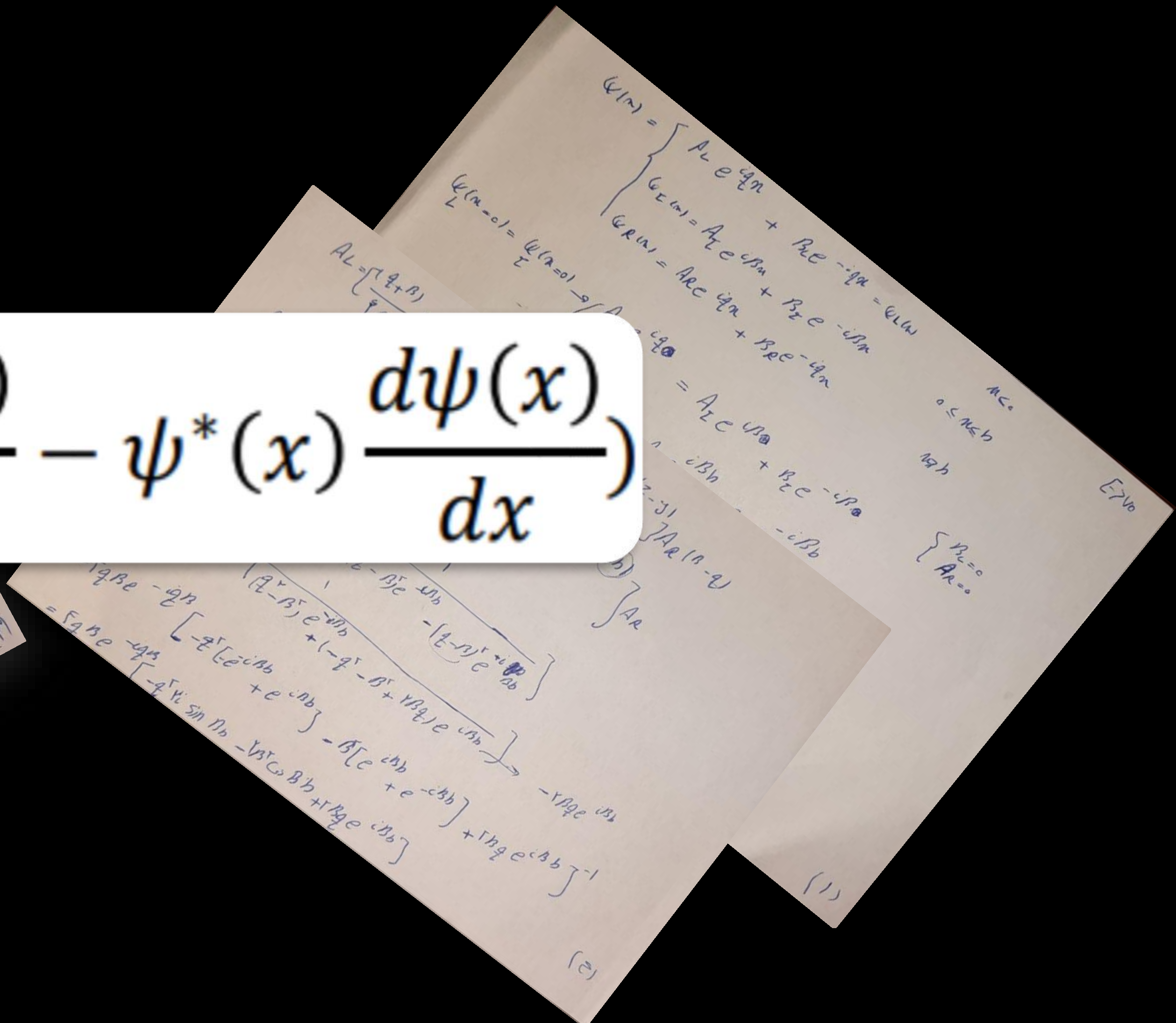


CALCULATING TRANSPORT

- **Current Density**
- **Transmission Matrix**
- **Green function**

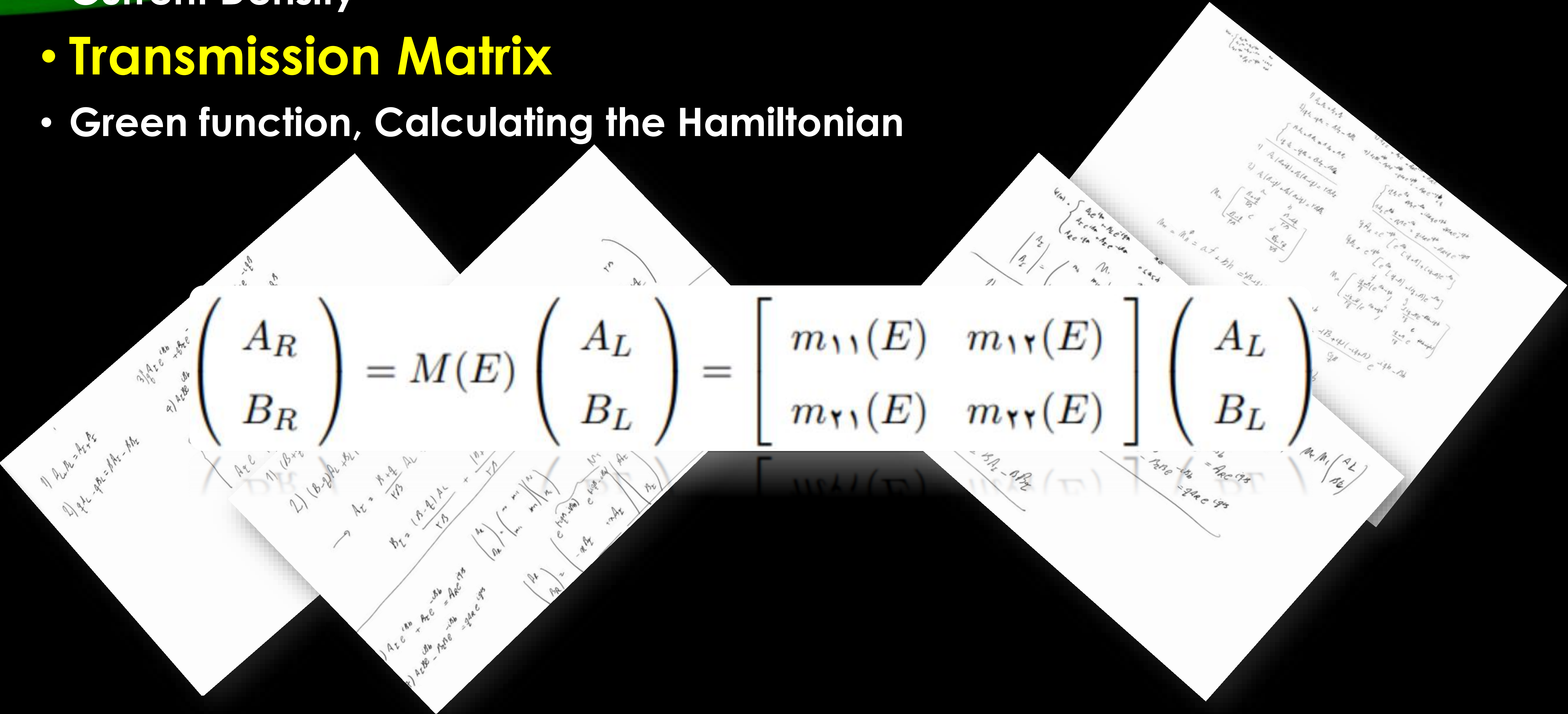
$$J = \frac{i\hbar}{2m}\left(\psi(x)\frac{d\psi^*(x)}{dx} - \psi^*(x)\frac{d\psi(x)}{dx}\right)$$





CALCULATING TRANSPORT

- **Current Density**
- **Transmission Matrix**
- **Green function, Calculating the Hamiltonian**

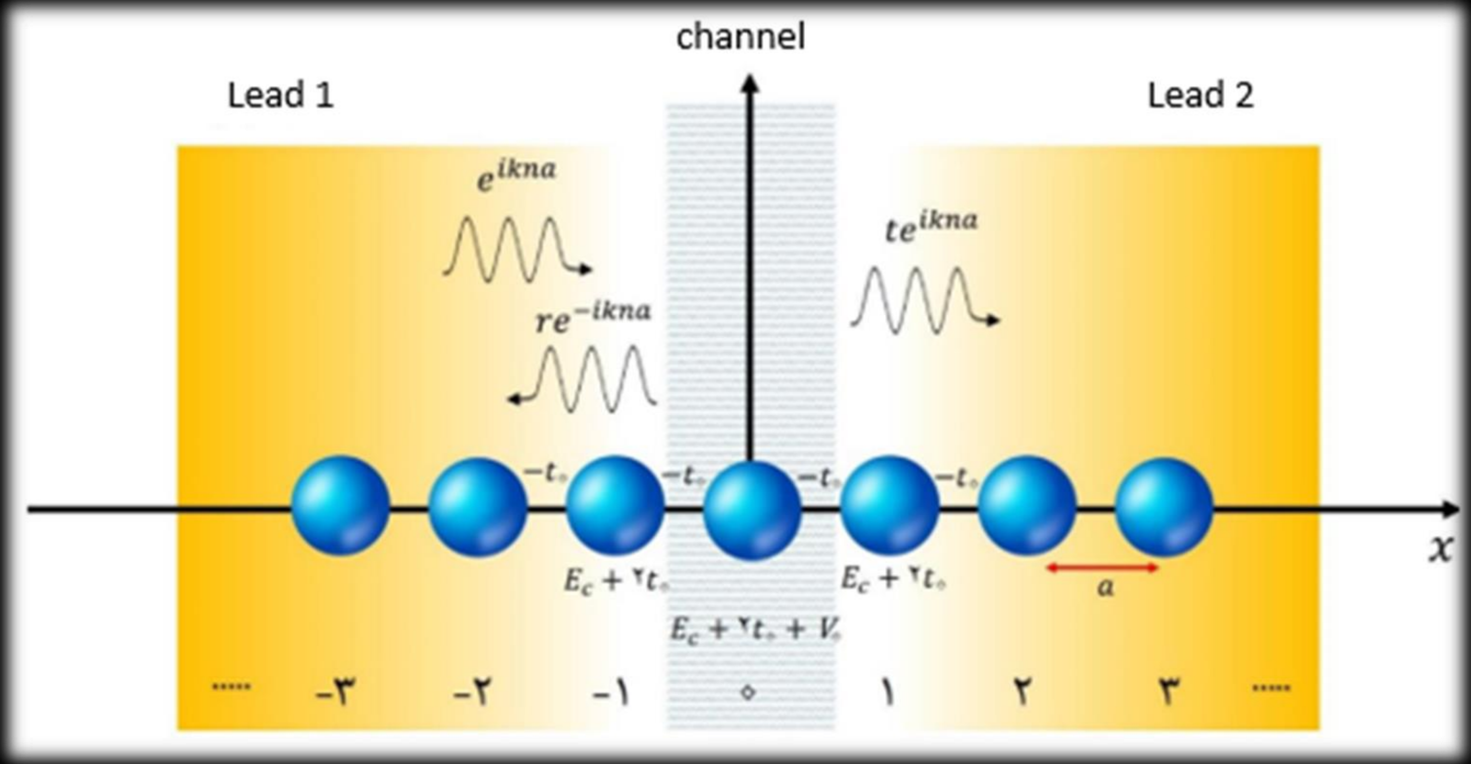


$$\begin{pmatrix} A_R \\ B_R \end{pmatrix} = M(E) \begin{pmatrix} A_L \\ B_L \end{pmatrix} = \begin{bmatrix} m_{11}(E) & m_{12}(E) \\ m_{21}(E) & m_{22}(E) \end{bmatrix} \begin{pmatrix} A_L \\ B_L \end{pmatrix}$$

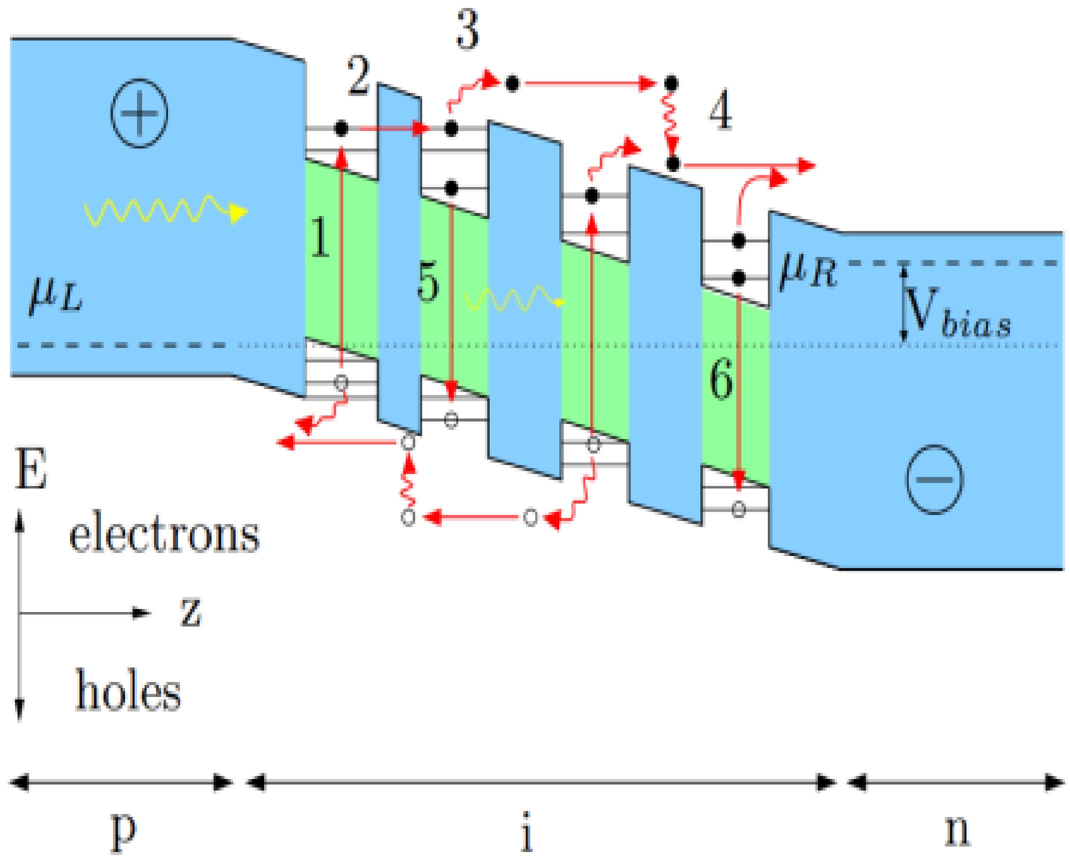
CALCULATING TRANSPORT

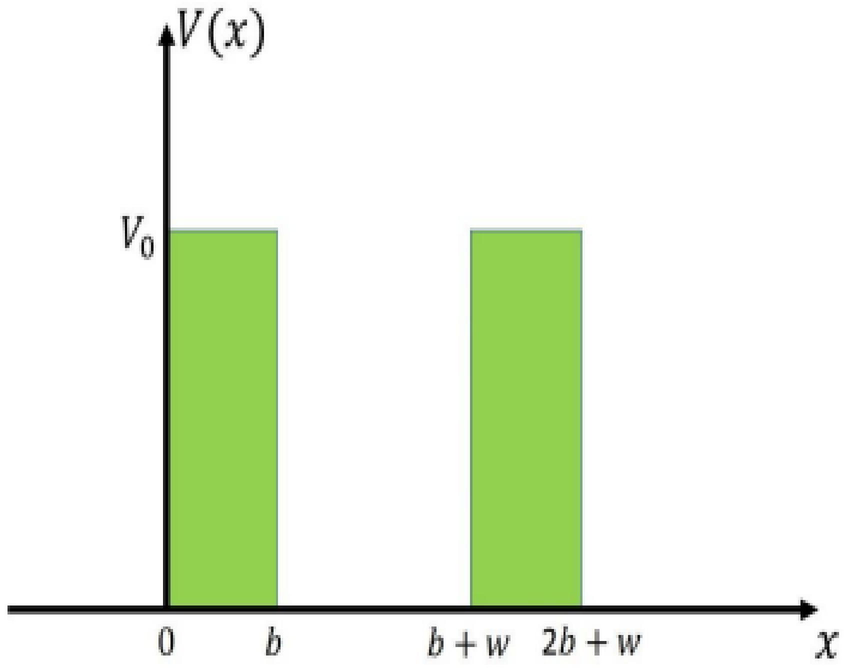
- **Current Density**
- **Transmission Matrix**
- **Green function, Calculating the Hamiltonian: Hubbard**

$$G = \frac{1}{[E - H + i\eta - \Sigma_1 - \Sigma_2]}$$



More steps…

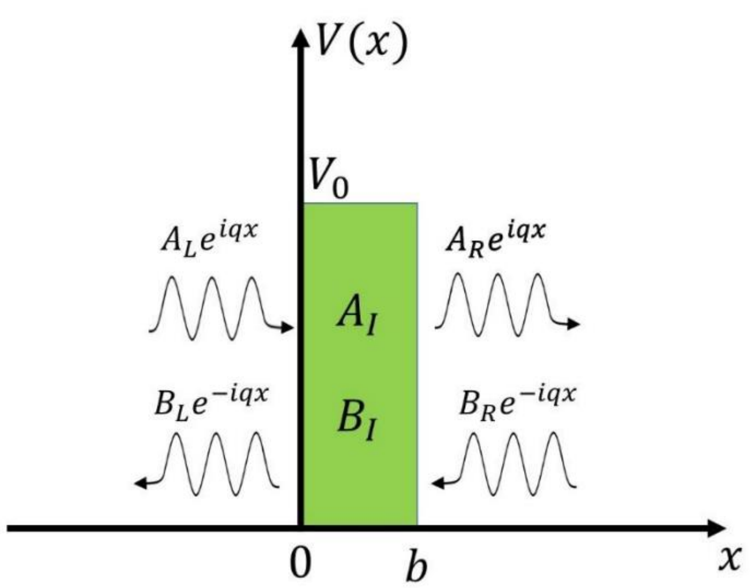


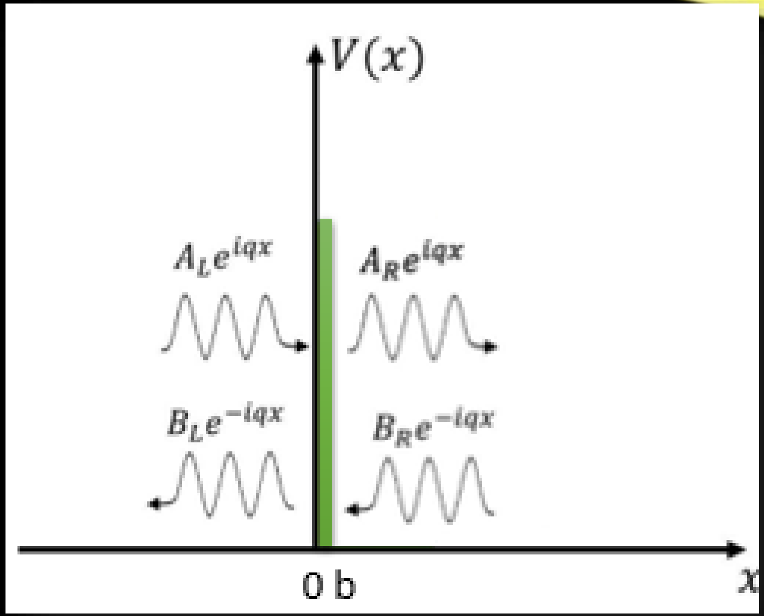


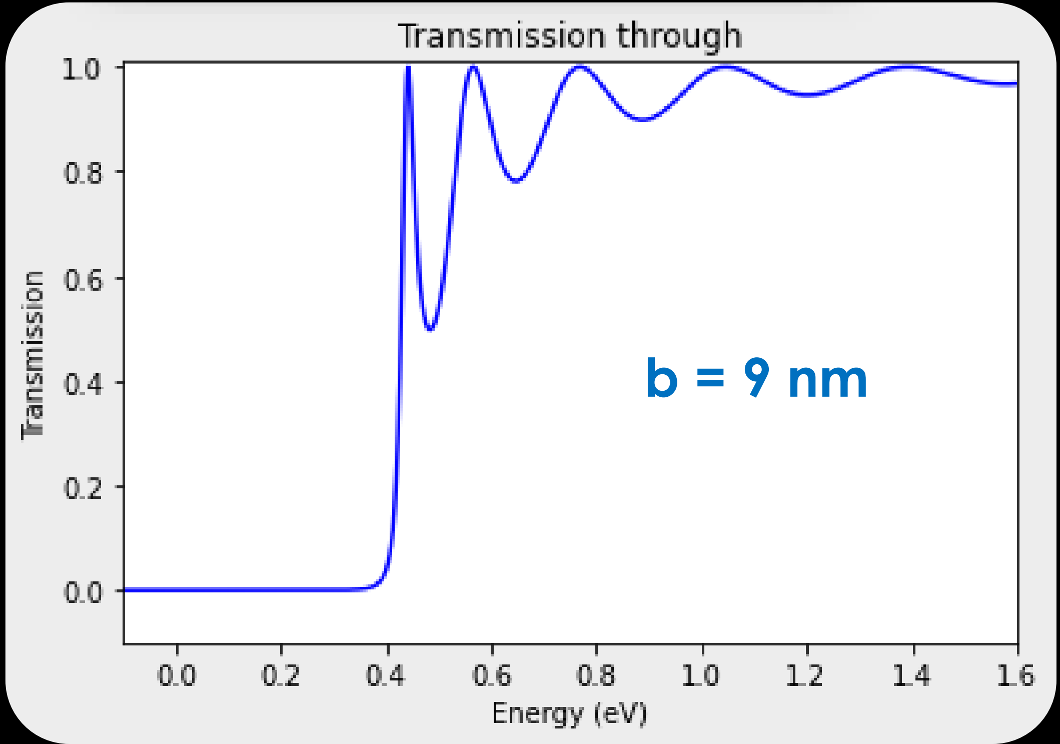
RESULTS

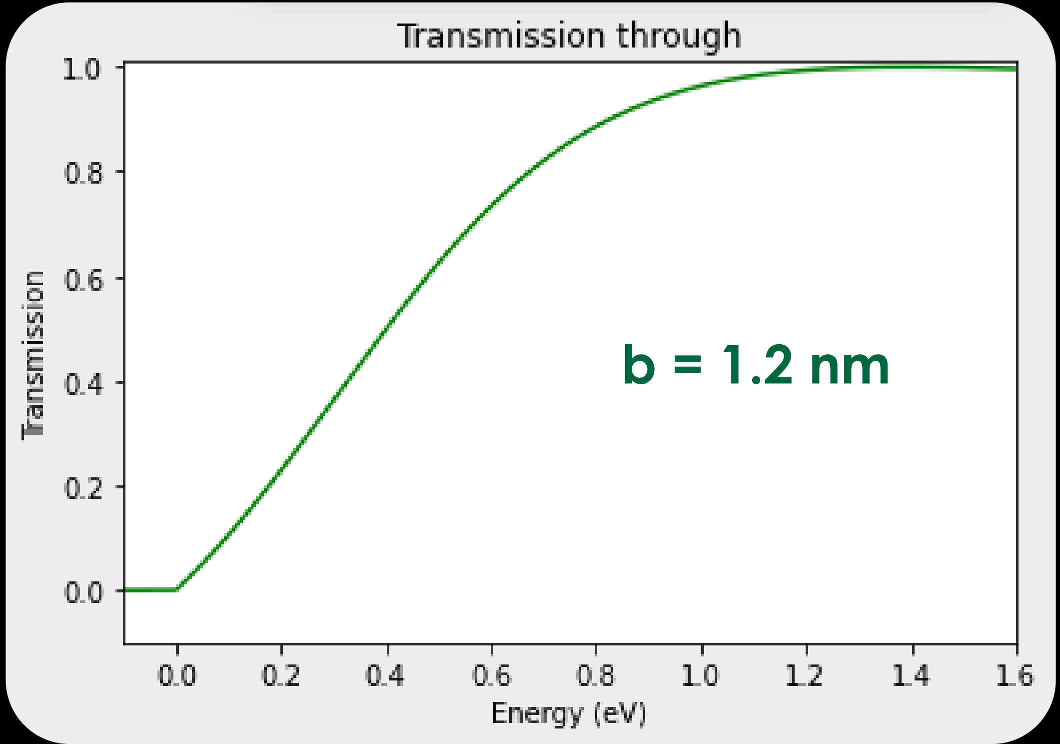
METHOD : GREEN FUNCTION

V = 0.4 eV



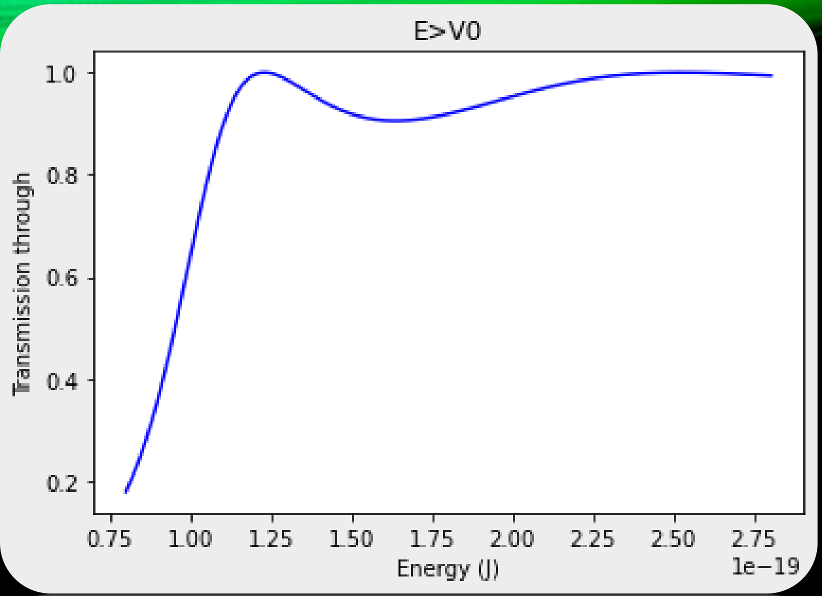




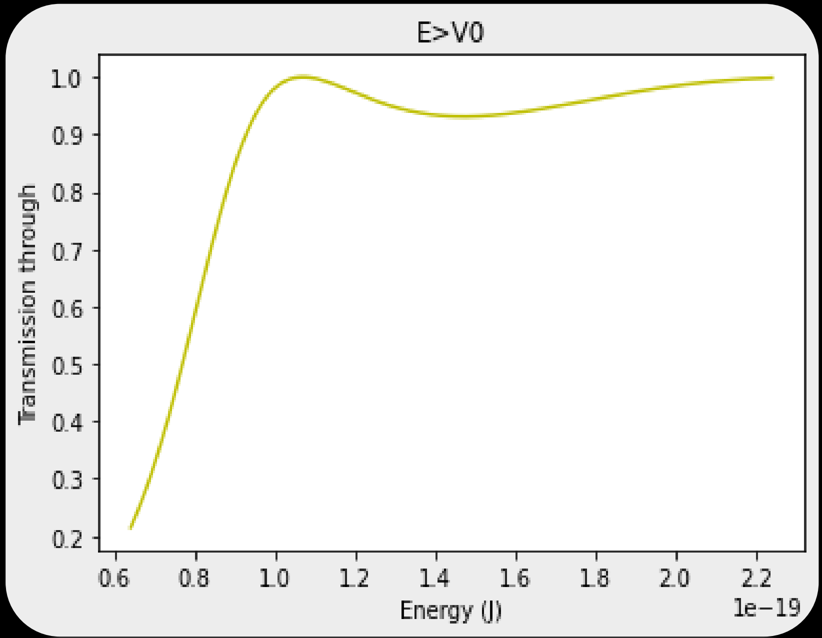


Results

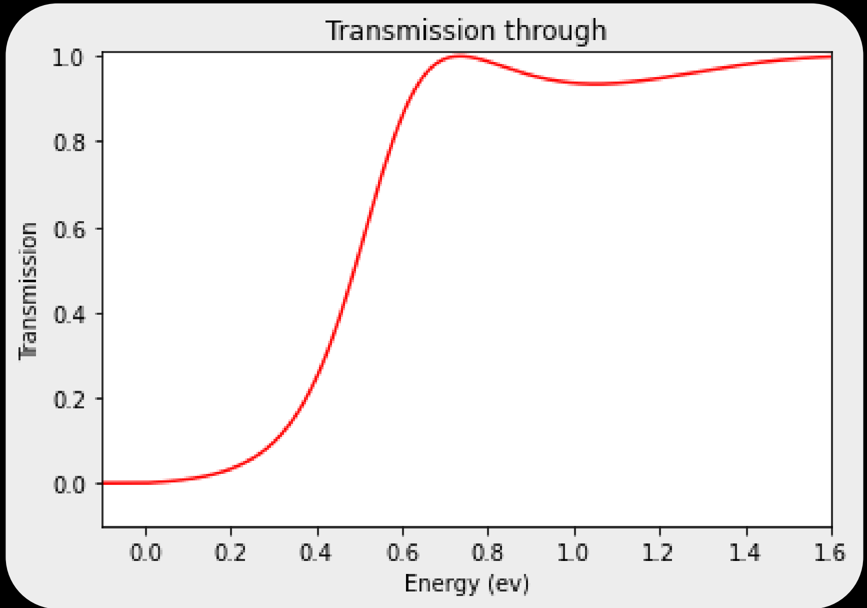
method : Current Density



method : Transmission Matrix



method : Green function




```
mport numpy as np
mport matplotlib.pyplot as plt
rom scipy import linalg
rom ipywidgets import interact , interactive , fixed , interact_manual
mport ipywidgets as widgets
matplotlib inline
a = 3*10**(-10)
= np.linspace(-0.2 , 1.6 , 5001)
bar = 1.054*10**(-34)
0 = 0.4
harge = 1.602*10**(-19)
= 0.25*9.1*10**(-31)
0 = (hbar**2)/(2*m*a**2)/charge
= int(input('N:'))
1 = (-t0)*np.eye(N , k=1)
2 = 2*t0*np.eye(N)
3 = (-t0)*np.eye(N , k=-1)
= T1+T2+T3
tta0 = 1e-12*1j
tta = Etta0*np.eye(50)
1 = int(input('n1:'))
2 = int(input('n2:'))
3 = N - n1 - n2
1 = np.zeros((1,n1), dtype = complex)
2 = u0*np.ones((1,n2), dtype = complex)
3 = np.zeros((1,n3), dtype = complex)
= np.concatenate((U1 , U2 , U3), axis = 1)
= U.transpose()
= U*np.eye(U.shape[0])
```

```
sigma1 = np.zeros((N,N), dtype = complex)
sigma2 = np.zeros((N,N), dtype = complex)
TM = []
for i in E:
    sigma1[0][0] = -t0*np.exp(1j*(np.arccos(1-((i+Etta0)/(2*t0)))))
    sigma2[49][49] = -t0*np.exp(1j*(np.arccos(1-((i+Etta0)/(2*t0)))))
    EE = i*np.eye(N)
    g = EE - H + Etta - sigma1 - sigma2
    Gamma1 = -2*np.imag(sigma1)
    Gamma2 = -2*np.imag(sigma2)
    G = linalg.inv(EE - H + Etta - sigma1 - sigma2)
    G_dag = G.conjugate().transpose()
    TM1 = np.dot(Gamma1,G)
    TM2 = np.dot(TM1,Gamma2)
    TM3 = np.dot(TM2,G_dag)
    TM.append(np.real(TM3.trace()))
plt.plot(E,TM , 'r')
plt.title('Transmission through')
plt.xlabel('Energy (ev)')
plt.ylabel('Transmission')
plt.xlim(-0.1, u0*4)
plt.ylim(-0.1 , 1.01)
plt.show()

N:50
n1:22
```

REFERENCES:

1. J. Grosso Vaj. Pastori Paravicini. Solid state physics. Translated by P. Sahib-Sara Volume 1, Isfahan University of Technology" 2018(0)2. N. Zettili, "Quantum mechanics: concepts and applications," (2003).

3. S. Datta., "Quantum Transport:Atom to Transistor', Cambridge University Press (2005).

4. H. Bruus, and E. Karsten., “Many-body quantum theory in condensed matter physics: an introduction"', OUP Oxford (2004).