Lecture11

Sung-Min Hong (smhong@gist.ac.kr)

Semiconductor Device Simulation Lab.
School of Electrical Engineering and Computer Science
Gwangju Institute of Science and Technology

GIST Lecture on October 17, 2018

Notice

- No lecture on October 29
  - Currently, no make-up session will be provided.

- No lecture on December 3 and 5
  - One make-up session on October 31
  - We need one more!

How to find $n(x, y, z)$

- The total number is known. But, how can we find $n(x, y, z)$?

    – Each state, $\psi_{k_x, k_y, n}(x, y, z)$, contributes $\left| \psi_{k_x, k_y, n}(x, y, z) \right|^2$.

    – Recall that the wavefunction can be written as
    $$\psi_{k_x, k_y, n}(x, y, z) = A_{k_x, k_y, n} e^{+ik_x x} e^{+ik_y y} \psi_{z,n}(z)$$

    – Then, $\left| \psi_{k_x, k_y, n}(x, y, z) \right|^2 = \left| A_{k_x, k_y, n} \right|^2 \left| \psi_{z,n}(z) \right|^2$

    – Integration of $\left| \psi_{k_x, k_y, n}(x, y, z) \right|^2$ over the box should give unity.
    $$L_x L_y \left| A_{k_x, k_y, n} \right|^2 \int_0^{L_z} dz \left| \psi_{z,n}(z) \right|^2 = 1$$

Normalization of $\psi_{z,n}(z)$

- If $\psi_{z,n}(z)$ is normalized in the 1D structure,

  - We have the following condition:

  $$\left|A_{k_x,k_y,n}\right|^2 = \frac{1}{L_x L_y}$$

  - Therefore, each state contributes

  $$\left|\psi_{k_x,k_y,n}(x,y,z)\right|^2 = \frac{1}{L_x L_y}\left|\psi_{z,n}(z)\right|^2$$

  - Note that every state in a subband has the same electron density. (In general, it does not hold.)

1D infinite potential well

- When $\psi_{z,n}(z) = A_{z,n} \sin\left(\frac{n\pi}{L_z} z\right)$,

  - The value of $A_{z,n}$ is $\sqrt{\frac{2}{L_z}}$.

  - Therefore, when fully occupied, a state in the n-th subband contributes an electron density of (per spin)
  $$\frac{2}{L_x L_y L_z} \sin^2\left(\frac{n\pi}{L_z} z\right)$$

  - Finally, the electron density can be obtained by considering all subbands.

MATLAB example (1)

- Preparing some constants (the same as before)

```
h = 6.626176e-34; % Planck constant, J s

hbar = h / (2*pi); % Reduced Planck constant, J s

q = 1.602192e-19; % Elementary charge, C

m0 = 9.109534e-31; % Electron rest mass, kg

k_B = 1.380662e-23; % Boltzmann constant, J/K

T = 300.0; % Temperature, K

Lx = 100e-9; Ly = 100e-9; Lz = 5e-9; % Lenghs, m

mxx = 0.19; myy = 0.19; mzz = 0.91; % Masses, m0

nmax = 10;

coef = 2*Lx*Ly/(2*pi)*sqrt(mxx*myy)*m0/(hbar^2)*(k_B*T);
```

MATLAB example (2)

- Calculation of `elec`

```
totalNumber = 0;
Nz = 51;
z = transpose([0:Nz-1])*Lz/(Nz-1);
elec = zeros(Nz,1); % Electron density, /m^3
for n=1:nmax
   Ez = (hbar^2)/(2*mzz*m0)*(pi*n/Lz)^2;
   subbandNumber = coef*log(1+exp(-Ez/(k_B*T)));
   totalNumber = totalNumber + subbandNumber;
   elec = elec + 2/(Lx*Ly*Lz)*(sin(n*pi*z/Lz).^2)*subbandNumber;
end
plot(z/1e-9,elec/1e6)
```
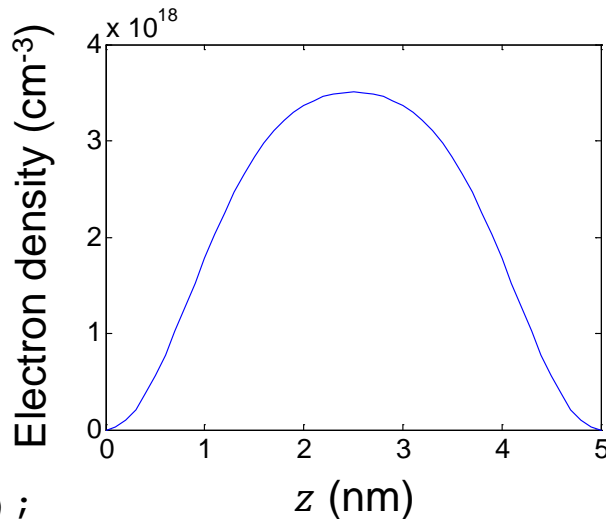
Homework#7

- Due: AM08:00, <u>October 22</u>
- Problem#1
  - Up to now, we have assumed that the Fermi energy is 0 eV.
  - In this problem, a 5-nm-thick potential well is considered again.
  - You can set it to any value. (-1 eV to +1 eV)
  - Calculate the integrated electron density (/cm^2) as a function of the gate voltage.
  - Show snapshots of the electron density for different Fermi energies.

Potential energy

- In the case of the infinite potential well, the potential energy is 0 eV.
  - Of course, in the realistic case, the potential energy is provided from the Poisson equation.
  - When the electro potential, $\phi(z)$, is given, the potential energy is written as

$$V(z) = -q\phi(z) + (E_c - E_i)$$

$E_c - E_i$: Constant for a given material

Schrödinger equation

- For the z-directional Schrodinger equation,
  - We have the following form:

  $$-\frac{\hbar^2}{2m_{zz}}\frac{d^2}{dz^2}\psi_{z,n}(z) + V(z)\,\psi_{z,n}(z) = E_{z,n}\psi_{z,n}(z)$$
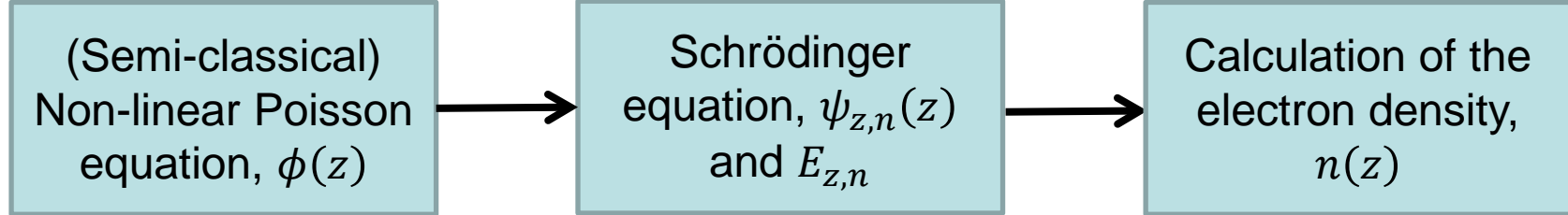
  - After a simple manipulation,

  $$\frac{d^2}{dz^2}\psi_{z,n}(z) - \frac{2m_{zz}}{\hbar^2}V(z)\,\psi_{z,n}(z) = -\frac{2m_{zz}}{\hbar^2}E_{z,n}\psi_{z,n}(z)$$

  - Discretized version

  $$\psi_{z,n,i+1} - 2\psi_{z,n,i} + \psi_{z,n,i-1} - \frac{2m_{zz}}{\hbar^2}V(z_i)(\Delta z)^2\psi_{z,n,i} = -\frac{2m_{zz}}{\hbar^2}E_{z,n}(\Delta z)^2\psi_{z,n,i}$$

Simulation flow

- Anyway, we need the initial solution.
  - It can be obtained from the semi-classical simulation.
  - Then, under the given potential energy, we can calculate the electron density.

```
┌─────────────────────┐      ┌─────────────────────┐      ┌─────────────────────┐
│   (Semi-classical)  │      │     Schrödinger     │      │  Calculation of the │
│  Non-linear Poisson │ ───▶ │ equation, ψ_{z,n}(z)│ ───▶ │   electron density, │
│   equation, φ(z)    │      │     and E_{z,n}     │      │        n(z)         │
└─────────────────────┘      └─────────────────────┘      └─────────────────────┘
```

- Of course, the electron density is not fully consistent with the Poisson equation. It should be improved later.

MATLAB example (1)

- Defining variables

```
h = 6.626176e-34; % Planck constant, J s
hbar = h / (2*pi); % Reduced Planck constant, J s
q = 1.602192e-19; % Elementary charge, C
m0 = 9.109534e-31; % Electron rest mass, kg
k_B = 1.380662e-23; % Boltzmann constant, J/K
eps0 = 8.854187817e-12; % Vacuum permittivity, F/m
T = 300.0; % Temperature, K
thermal = k_B*T/q; % Thermal voltage, V
mxx = 0.19; myy = 0.19; mzz = 0.91; % Masses, m0
Deltaz = 0.1e-9; % 0.1 nm spacing
Nz = 61; % 6 nm thick
z = Deltaz*transpose([0:Nz-1]); % real space, m
interface1 = 6; % At z=0.5 nm
interface2 = 56; % At z=5.5 nm
eps_si = 11.7; eps_ox = 3.9; % Relative permittivity
Nacc = 1e24; % 1e18 /cm^3
ni = 1.075e16; % 1.075e10 /cm^3
coef = Deltaz*Deltaz*q/eps0;
Ec_Ei = 0.561004; % E_c – E_i, eV
```

MATLAB example (2)

- Semi-classical nonlinear Poisson solver

```
phi = zeros(Nz,1);
phi(:,1) = 0.33374;
for newton=1:10
    res = zeros(Nz,1);
    Jaco = sparse(Nz,Nz);
    res(1,1) = phi(1,1) - 0.33374;
    Jaco(1,1) = 1.0;
    res(Nz,1) = phi(Nz,1) - 0.33374;
    Jaco(Nz,Nz) = 1.0;
    for ii=2:Nz-1
        if      (ii< interface1 || ii> interface2)
            res(ii,1) = eps_ox*phi(ii+1,1) - 2*eps_ox*phi(ii,1) + eps_ox*phi(ii-1,1);
            Jaco(ii,ii-1) = eps_ox; Jaco(ii,ii) = -2*eps_ox;        Jaco(ii,ii+1) = eps_ox;
        elseif (ii==interface1)
            res(ii,1) = eps_si*phi(ii+1,1) - (eps_si+eps_ox)*phi(ii,1) + eps_ox*phi(ii-1,1);
            Jaco(ii,ii-1) = eps_ox; Jaco(ii,ii) = -(eps_si+eps_ox); Jaco(ii,ii+1) = eps_si;
        elseif (ii==interface2)
            res(ii,1) = eps_ox*phi(ii+1,1) - (eps_ox+eps_si)*phi(ii,1) + eps_si*phi(ii-1,1);
            Jaco(ii,ii-1) = eps_si; Jaco(ii,ii) = -(eps_ox+eps_si); Jaco(ii,ii+1) = eps_ox;
        else
            res(ii,1) = eps_si*phi(ii+1,1) - 2*eps_si*phi(ii,1) + eps_si*phi(ii-1,1);
            Jaco(ii,ii-1) = eps_si; Jaco(ii,ii) = -2*eps_si;        Jaco(ii,ii+1) = eps_si;
        end
    end
end
```

MATLAB example (3)

- Semi-classical nonlinear Poisson solver (continued)

```
for ii=interface1:interface2
   if      (ii==interface1)
      res(ii,1) = res(ii,1) - coef*(Nacc+ni*exp(phi(ii,1)/thermal))*0.5;
      Jaco(ii,ii) = Jaco(ii,ii) - coef*ni*exp(phi(ii,1)/thermal)/thermal*0.5;
   elseif (ii==interface2)
      res(ii,1) = res(ii,1) - coef*(Nacc+ni*exp(phi(ii,1)/thermal))*0.5;
      Jaco(ii,ii) = Jaco(ii,ii) - coef*ni*exp(phi(ii,1)/thermal)/thermal*0.5;
   else
      res(ii,1) = res(ii,1) - coef*(Nacc+ni*exp(phi(ii,1)/thermal));
      Jaco(ii,ii) = Jaco(ii,ii) - coef*ni*exp(phi(ii,1)/thermal)/thermal;
   end
   end
   update = Jaco \ (-res);
   phi = phi + update;
 end
```

MATLAB example (3)

- Schrödinger solver
  - Now we have the electrostatic potential. The potenti energy is

```
V = q*Ec_Ei - q*phi; % Potential energy, J
```

  - Only for bulk silicon nodes, the Hamiltonian is constructured.

```
Nbulk = interface2-interface1-1; % Number of bulk silicon nodes
Hamil = zeros(Nbulk,Nbulk);
Hamil(1,1) = -2; Hamil(1,2) = 1;
for ii=2:Nbulk-1
    Hamil(ii,ii+1) =  1;
    Hamil(ii,ii  ) = -2;
    Hamil(ii,ii-1) =  1;
end
Hamil(Nbulk,Nbulk) = -2; Hamil(Nbulk,Nbulk-1) = 1;
```

MATLAB example (4)

- Schrödinger solver (continued)
    - The potential energy is added.

```
for ii=1:Nbulk
    Hamil(ii,ii) = Hamil(ii,ii) -2*mzz*m0*(Deltaz/hbar)^2*V(ii+interface1,1);
end
```

    - Get the solution.

```
[eigenvectors,eigenvalues] = eig(Hamil);
```

    - Scale the eigenenergies.

```
Ez = diag(eigenvalues)/(-2*mzz*m0*(Deltaz/hbar)^2); % Eigenenergy, J
```

    - The wavefunction should be scaled.

Normalization of $\psi_{z,n}(z)$

- Its normalization condition
  - Intergration form

$$\int_0^{L_z} dz \left|\psi_{z,n}(z)\right|^2 = 1$$

  - Discretized version

$$\sum_i \Delta z \left|\psi_{z,n,i}\right|^2 = 1$$

  - Eigenfunctions should be normalized accordingly.

Modified MATLAB code (1)

- Defining variables (1)

```
h = 6.626176e-34; % Planck constant, J s

hbar = h / (2*pi); % Reduced Planck constant, J s

q = 1.602192e-19; % Elementary charge, C

m0 = 9.109534e-31; % Electron rest mass, kg

k_B = 1.380662e-23; % Boltzmann constant, J/K

eps0 = 8.854187817e-12; % Vacuum permittivity, F/m

T = 300.0; % Temperature, K

thermal = k_B*T/q; % Thermal voltage, V
```

Modified MATLAB code (2)

- Defining variables (2)

```
Lx = 100e-9; Ly = 100e-9; % Lenghs, m
mxx = 0.19; myy = 0.19; mzz = 0.91; % Masses, m0
Deltaz = 0.1e-9; % 0.1 nm spacing
Nz = 61; % 6 nm thick
z = Deltaz*transpose([0:Nz-1]); % real space, m
interface1 = 6; % At z=0.5 nm
interface2 = 56; % At z=5.5 nm
eps_si = 11.7; eps_ox = 3.9; % Relative permittivity
Nacc = 1e24; % 1e18 /cm^3
ni = 1.075e16; % 1.075e10 /cm^3
coef_Poi = Deltaz*Deltaz*q/eps0;
coef_Sch = 2*Lx*Ly/(2*pi)*sqrt(mxx*myy)*m0/(hbar^2)*(k_B*T);
Ec_Ei = 0.561004; % E_c - E_i, eV
```

Modified MATLAB code (3)

- Semi-classical nonlinear Poisson solver

```
phi = zeros(Nz,1);
phi(:,1) = 0.33374;
for newton=1:10
    res = zeros(Nz,1);
    Jaco = sparse(Nz,Nz);
    res(1,1) = phi(1,1) - 0.33374;
    Jaco(1,1) = 1.0;
    res(Nz,1) = phi(Nz,1) - 0.33374;
    Jaco(Nz,Nz) = 1.0;
```

Modified MATLAB code (4)

- Semi-classical nonlinear Poisson solver

```
for ii=2:Nz-1
    if      (ii< interface1 || ii> interface2)
        res(ii,1) = eps_ox*phi(ii+1,1) - 2*eps_ox*phi(ii,1) + eps_ox*phi(ii-1,1);
        Jaco(ii,ii-1) = eps_ox; Jaco(ii,ii) = -2*eps_ox;         Jaco(ii,ii+1) = eps_ox;
    elseif (ii==interface1)
        res(ii,1) = eps_si*phi(ii+1,1) - (eps_si+eps_ox)*phi(ii,1) + eps_ox*phi(ii-1,1);
        Jaco(ii,ii-1) = eps_ox; Jaco(ii,ii) = -(eps_si+eps_ox); Jaco(ii,ii+1) = eps_si;
    elseif (ii==interface2)
        res(ii,1) = eps_ox*phi(ii+1,1) - (eps_ox+eps_si)*phi(ii,1) + eps_si*phi(ii-1,1);
        Jaco(ii,ii-1) = eps_si; Jaco(ii,ii) = -(eps_ox+eps_si); Jaco(ii,ii+1) = eps_ox;
    else
        res(ii,1) = eps_si*phi(ii+1,1) - 2*eps_si*phi(ii,1) + eps_si*phi(ii-1,1);
        Jaco(ii,ii-1) = eps_si; Jaco(ii,ii) = -2*eps_si;         Jaco(ii,ii+1) = eps_si;
    end
  end
```

Modified MATLAB code (5)

- Semi-classical nonlinear Poisson solver (continued)

```
for ii=interface1:interface2
    if      (ii==interface1)
        res(ii,1) = res(ii,1) - coef_Poi*(Nacc+ni*exp(phi(ii,1)/thermal))*0.5;
        Jaco(ii,ii) = Jaco(ii,ii) - coef_Poi*ni*exp(phi(ii,1)/thermal)/thermal*0.5;
    elseif (ii==interface2)
        res(ii,1) = res(ii,1) - coef_Poi*(Nacc+ni*exp(phi(ii,1)/thermal))*0.5;
        Jaco(ii,ii) = Jaco(ii,ii) - coef_Poi*ni*exp(phi(ii,1)/thermal)/thermal*0.5;
    else
        res(ii,1) = res(ii,1) - coef_Poi*(Nacc+ni*exp(phi(ii,1)/thermal));
        Jaco(ii,ii) = Jaco(ii,ii) - coef_Poi*ni*exp(phi(ii,1)/thermal)/thermal;
    end
end
update = Jaco \ (-res);
phi = phi + update;
end
```

Modified MATLAB code (6)

- Schrödinger solver

```
V = q*Ec_Ei - q*phi; % Potential energy, J
Nbulk = interface2-interface1-1; % Number of bulk silicon nodes
Hamil = zeros(Nbulk,Nbulk);
Hamil(1,1) = -2; Hamil(1,2) = 1;
for ii=2:Nbulk-1
    Hamil(ii,ii+1) =  1;
    Hamil(ii,ii  ) = -2;
    Hamil(ii,ii-1) =  1;
end
Hamil(Nbulk,Nbulk) = -2; Hamil(Nbulk,Nbulk-1) = 1;
for ii=1:Nbulk
    Hamil(ii,ii) = Hamil(ii,ii) -2*mzz*m0*(Deltaz/hbar)^2*V(ii+interface1,1);
end
```

MATLAB example (1)

- Continued discussion

  – Get the solution.

```
[eigenvectors,eigenvalues] = eig(Hamil);
```

  – Scale the eigen-energies.

```
scaledEz = diag(eigenvalues)/(-2*mzz*m0*(Deltaz/hbar)^2); % Eigenenergy, J
```

  – Sort the eigen-energies.
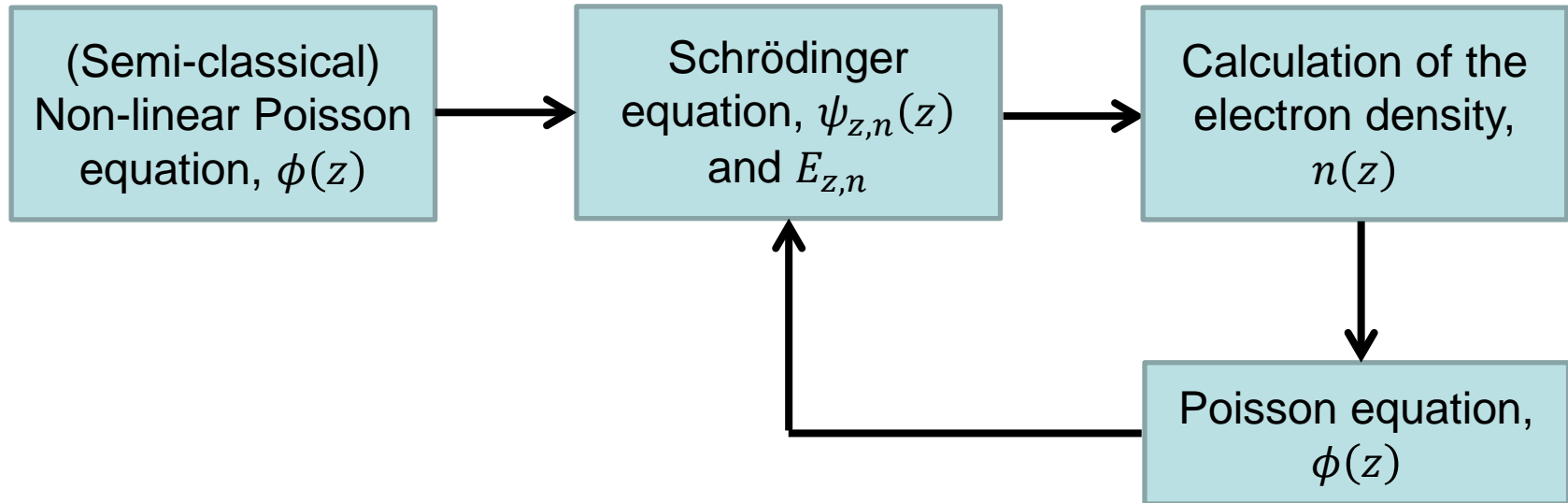
```
[sortedEz,sortedIndex] = sort(scaledEz);
```

MATLAB example (2)

- Calculation of the electron density

```
nSubband = 10;
elec = zeros(Nz,1); % Electron density, /m^3
totalNumber = 0;
for n=1:nSubband
  Ez = sortedEz(n,1);
  wavefunction2 = eigenvectors(:,sortedIndex(n)).^2;
  normalization = sum(wavefunction2)*Deltaz;
  wavefunction2 = wavefunction2 / normalization;
  subbandNumber = coef_Sch*log(1+exp(-Ez/(k_B*T)));
  totalNumber = totalNumber + subbandNumber;
  elec(interface1+1:interface2-1,1) = elec(interface1+1:interface2-1,1) +
1/(Lx*Ly)*wavefunction2*subbandNumber;
end
```

Simulation flow

- Self-consistency
  - The electron density is not consistent with the Poisson equation.
  - Better way?

$$
\boxed{\text{(Semi-classical) Non-linear Poisson equation, } \phi(z)} \rightarrow \boxed{\text{Schrödinger equation, } \psi_{z,n}(z) \text{ and } E_{z,n}} \rightarrow \boxed{\text{Calculation of the electron density, } n(z)}
$$

$$
\boxed{\text{Poisson equation, } \phi(z)}
$$

6 valleys in silicon

- Up to now, we have considered only one band.
  - In silicon, we need to consider three valley pairs.
  - They can be characterized by (for a certain channel direction)

$m_{xx} = 0.91 m_0, m_{yy} = 0.19 m_0, m_{zz} = 0.19 m_0$

$m_{xx} = 0.19 m_0, m_{yy} = 0.91 m_0, m_{zz} = 0.19 m_0$

$m_{xx} = 0.19 m_0, m_{yy} = 0.19 m_0, m_{zz} = 0.91 m_0$

  - Each of them has two-fold degeneracy.

MATLAB example

- Pseudocode

```
(Defining constants. Copy-and-paste)
(Semi-classical nonlinear Poisson equation. Copy-and-paste)
for iNewton = 1:20
    totalNumber = 0;
    elec = zeros(Nz,1); % Electron density, /m^3
    for iValley = 1:3
        mass = ones(3)*0.19;
        mass(iValley) = 0.91;
        coef_Sch = 2*Lx*Ly/(2*pi)*sqrt(mass(1)*mass(2))*m0/(hbar^2)*(k_B*T);
        (Schrodinger solver. Now mzz becomes mass(3).)
        (Calculation of electron density. Add it to elec.)
    end
    (Poisson equation)
end
```