Using NEMO5 to quantitatively predict topological insulator behaviour

Parijat Sengupta, Tillmann Kubis, Michael Povolotskyi, Jean Michel Sellier, Jim Fonseca, Gerhard Klimeck

Network for Computational Nanotechnology (NCN)
Electrical and Computer Engineering
Purdue University, West Lafayette IN, USA

Summer School 2012
Some commonly known facts

- Metal
- Semiconductor
- Insulator

Electron energy

Overlap

Conduction band

Fermi level

Valence band

Bandgap

*Wikipedia
An insulator such as $\text{SiO}_2$, $\text{Si}_3\text{N}_4$ with large band-gap.

- Conduction and Valence bands describe electrons and holes.
- The band gap is zero in a metal.

- Some materials possess the dual properties of metals and insulators.
- We will use NEMO5 to investigate some of these unique materials.
Primary examples are Bi$_2$Te$_3$, Bi$_2$Se$_3$, Bi$_x$Sb$_{1-x}$ etc…

They are known as topological insulators

These materials contain bound surface states

Dispersion obtained through ARPES

sp$^3$d$^5$s$^*$ tight binding calculation

Dirac Cone

Rev of Mod. Phys, Vol 82, Oct-Dec 2010
We will use a sp$^3$d$^5$s$^*$ tight binding model.

Bi$_2$Te$_3$ is our topological insulator material.

A 9 nm (appx) long quantum well will be the target device.

This will produce a band structure as shown on left.

Key feature of this dispersion is a graphene-like linear character (Dirac cone).

Linearity is not due to same reason as in graphene!
Details of the simulation structure

9 nm wide c-axis oriented Bi$_2$Te$_3$ thin film

- Bi terminated surface
- Te terminated surface

Different surface termination: Surface 1 (2) is Bi (Te2) terminated
The NEMO5 input deck skeleton

There are 3 blocks in the input deck:

- **Structure**
  
  ```
  {
  }
  ```
  **Defines material and simulation domains**

- **Solvers**
  
  ```
  {
  }
  ```
  **Sets up simulations that has to be solved, e.g. equations, boundary conditions, iterative processes, output, numerical options, etc.**

- **Global**
  
  ```
  {
  }
  ```
  **Defines global variables such as temperature, which database file to use, diagnostic output, etc.**
The NEMO5 input deck: The first block “Structure”

```
Structure
{
    Material
    {
        name = XYZ
        ...
    }
    Material
    {
        name = ABC
        ...
    }
    Domain
    {
        name = device
        ...
    }
    Domain
    {
        name = contact1
        ...
    }
    Geometry
    {
    }
}
```
Setting the input deck...

Every input deck begins with the **Structure** group.

Each region is identified by a name known as the **tag**.

The underlying crystal structure of the material needs to be provided.

Each simulation can have multiple **regions**, NEMO5 needs the exact number of those for correct execution.

The surface atoms can be controlled by asking NEMO5 to place a specific atom when constructing the first unit cell.
Defining parameters and the domain

Bands:TB:sp3d5sstar_SO:param_Lee_von_Allmen:V_S_S_Sigma_Bi_H = -10

- Any parameter defined in the input deck takes precedence over the corresponding database entry

```plaintext
Domain
{
    name = device
    type = pseudomorphic
    base_material = shell
    dimension = (3,1,1)
    starting_cell_coordinate = (0, 0, 0)
    periodic = (false, true, true)
    passivate = false
}
```

- **Dimension** creates a canvas of unit cells
- Periodicity set as false refers to a confined axis \( \rightarrow k \) is not a good quantum number
- **Passivate** options allows inclusion of Hydrogen atoms
Crystal orientation in NEMO5

Crystal directions set up the coordinate system within the crystal using the basis vectors.

The basis vectors are aligned to the Cartesian axes through the space orientation option.

In this example, `crystal_direction1 = (0,0,1)` is aligned along `space_orientation_dir1 = (0,0,1)` or the z-axis.

You need to specify only 2 directions. N5 computes the third using the crystal structure info.

```cpp
// miller_index_basis = primitive
crystal_direction1 = ( 0, 0, 1)
crystal_direction2 = ( 1, 0, 0)
crystal_direction3 = ( 0, 1, 0)

space_orientation_dir1 = (0.0, 0.0, 1.0)
space_orientation_dir2 = (1.0, 0.0, 0.0)

periodic = (false, true, true)

regions = (1)
```

`space_orientation_dir1` is the z-axis and is confined: the first entry of `periodic` option.
• What NEMO5 internally produces?
• It produces three basis vectors
• These basis vectors can be obtained by examining the first three lines of the device_coupling.dat file.

To obtain this file, include `output = (xyz, coupling)` in the Domain section. You will see an example of it in your first exercise.

How do you check that these vectors make sense?

All length units are in nm.
Our test case here is Bi$_2$Te$_3$ which has a hexagonal base

- Each edge of hexagon is 0.4383 nm
- Two hexagons are displaced along z-axis by 3.0487 nm

Three vectors in the `device_coupling.dat` file

1. $a = [0 \ 0 \ 3.0487]$
2. $b = [0.4383 \ 0 \ 0]$
3. $c = [-0.2191 \ 0.3796 \ 0]$

Check:

- Norm of vectors $b$ and $c$ is equal to 0.4383 nm
- Norm of vector $a$ is equal to 3.0487 nm
- Angle between $b$ and $c$ is 120° as expected
The actual geometry in numbers

```java
Geometry {
  Region {
    shape = cuboid
    region_number = 1
    priority = 1
    min = ( -100, -100, 0)
    max = ( 100, 100, 10)
  }
}
```

- Shape options allows you to input a geometric shape for your device
- Setting the `priority` gives you control over structure creation when dealing with composite geometric shapes.
  - Higher priority gets constructed first
- Actual device size is specified through min & max
There are 3 blocks in the input deck:

- **Structure**
  ```
  {
  }
  ```
  Defines material and simulation domains

- **Solvers**
  ```
  {
  }
  ```
  Sets up simulations that has to be solved, e.g. equations, boundary conditions, iterative processes, output, numerical options, etc.

- **Global**
  ```
  {
  }
  ```
  Defines global variables such as temperature, which database file to use, diagnostic output, etc.
Executing the input deck: The solvers!

Solvers
{
  solver
  {
    name = sim_output
    type = Structure
    domain = device
    output_format = vtk
  }
  structure_file = all_atoms.vtk
}

- Each solver has a name
- The first solver (usually) is the geometry constructor
- The atomic coordinate positions are dumped out in a structure file
- The structure file can have several formats such as vtk, silo, xyz, pdb…
Electronic structure calculation must set the type to Schroedinger

*job_list* shows what actions N5 must perform

Setting *orbital_resolved* to *true* shows the contribution of each orbital to overall band structure
**tb_basis** option lets you choose the band structure calculation model.

- **tb_basis** can also be set to effective mass (em)
- **k_space_basis** can also be set to Cartesian
- Number of nodes → Number of k-points in the chosen k interval

```python
tb_basis = sp3d5sstar_SO

k_space_basis = reciprocal

k_points = [(0, -0.25), (0.0, 0.25)]

number_of_nodes = 2000
```
The NEMO5 input deck skeleton

There are 3 blocks in the input deck:

1. **Structure**
   ```
   {
   }
   ```
   Defines material and simulation domains

2. **Solvers**
   ```
   {
   }
   ```
   Sets up simulations that has to be solved, e.g. equations, boundary conditions, iterative processes, output, numerical options, etc.

3. **Global**
   ```
   {
   }
   ```
   Defines global variables such as temperature, which database file to use, diagnostic output, etc.
Global
{
    solve = (sim_output, BiTe_bands)
    database = ../..../..../materials/all.mat
    messaging_level = 5
}

- The *global* section has the command *solve*: All job names are included under *solve*
- *database* is custom built for N5 and contains all material parameters
- *messaging level* indicates *code progress output* on screen. Level 5 is verbose
• Log in to your workspace account and create a folder **TI** in your home directory. Navigate to the **TI** folder
  – `mkdir TI`
  – `cd TI`

• Pull the necessary files in your TI folder by typing the following:

  ```
  cp ../public_examples/NCN_summer_school_2012/
  Topological_Insulators_psengupta/ex*.in  .
  cp ../public_examples/NCN_summer_school_2012/
  Topological_Insulators_psengupta/spin_analysis_ex*.m  .
  ```

• **ex1.in** is the input file to calculate the dispersion relationship for a 9.0 (appx) long Bi$_2$Te$_3$ quantum well

• Submit a job by typing the command:

  ```
  submit -v ncn-hub@coates -i ../all.mat -n 16 -N 8 nemo-r8028 ./ex1.in
  ```
What do you expect to see as a solution?

- NEMO5 will produce four files (SS12_TI_ex1_*.dat)
- Start MATLAB on your workspace
- Your folder has a MATLAB file called `spin_analysis_ex1.m`
- Execute the MATLAB script by typing `spin_analysis_ex1` at the command prompt
- You will have the figure on your left!
Pre-computed results are stored in folder 
/public_examples/NCN_summer_school_2012/Topological_Insulators_psengupta

- **Supplementary exercise**: 
  - Change line 20 of the `spin_analysis_ex1.m` file from the preset
    - $n = [1 \ 0 \ 0]$ to $n = [0 \ 0 \ 1]$ and $n = [0 \ 1 \ 0]$ 
  - Run Matlab
  - The three different spin-polarized plots that you obtain highlight a fundamental theory of TIs
Several solvers defined in the Solver block can be inter-linked:

```solvers
Solver1
{
}
Solver2
{
}
Solver3
{
}
```

We will see a specific example of this coupling of solvers in the next part of the tutorial.
NEMO5 is a toolbox....

Solvers can also speak to one another
- e.g. A self-consistent charge calculation

\[
\text{solve} = (\text{struct, schrödinger, poisson})
\]

Start

Density solver (e.g. Schrödinger)

Potential solver (e.g. Poisson (\(\phi\)))

Conv ?

End

Update \(\phi\)

No

Yes
A name is assigned to the Poisson solver (similar to previous slides)

A non-linear Poisson will be solved

A continuum domain is constructed

Linear solver settings

Electrostatic output option

```plaintext
solver {
  name = my_poisson
  type = NonlinearPoisson
  domain = continuum
  ksp_type = gmres
  pc_type = ilu
  atomistic_output = (potential, charge)
  one_dim_output = (potential, charge)
  one_dim_output_average = false
}
```
Linking Poisson to Schroedinger...

```python
charge_model = electron_core
rel_tolerance = 1e-6
max_nonlinear_step = 500
density_solver = BiTe_density
}
solver
{
    name = BiTe_density
    type = Schroedinger
    domain = device
    job_list = (electron_density, derivative_electron_density_over_potential)
    output = (electron_density_VTK)
```
A few options in the density solver

- First two blocks of input statements are identical to normal eigen value calculations done earlier

- `threshold_energy` lets you choose eigen states beginning with energy as set in the option

- `chem_pot` is the Fermi level to start calculations

- Schrodinger receives the potential from the `my_poisson potential solver`

```plaintext
charge_model = electron_core
tb_basis = sp3d5sstar_SO
number_of_eigenvalues = 100

k_space_basis = reciprocal
threshold_energy = -20

chem_pot = 0.0
number_of_k_points = 30
potential_solver = my_poisson
```

Inter-linking solvers: `my_poisson` and `BiTe_density`

More advanced calculations can use boundary condition options
Boundary conditions can be of two forms in NEMO5.

1) Dirchlet (potential ($\phi$) = constant)

2) Neumann ($\partial \phi$ = constant)

- The input deck statements shown on left can be included multiple times at all possible external boundaries.
- The right boundary region number must be specified.
- $E_{\text{field}}$ allows to apply an electric field of certain magnitude to device structure. It is in units of V/cm.
Updating Schrödinger for eigen states calculation

```python
# Exactly same format as in Exercise 1

# Potential solver is added which supplies potential to the tight-binding Hamiltonian

tb_basis = sp3d5sstar_SO
k_space_basis = reciprocal
k_points = [(0, -0.25), (0.0, 0.25)]
number_of_nodes = (2000)
potential_solver = my_poisson
```
• **ex2.in** is the input file to calculate charge self-consistent dispersion relationship for a 9.0 (aprx) long Bi$_2$Te$_3$ quantum well

• This task will produce a dispersion relationship, potential landscape, and the charge profile in the device

• Submit the job by typing the command:
  
  ```bash
  submit -v ncn-hub@coates -i ../all.mat -n 16 -N 8 nemo-r8028 ./ex2.in
  ```
What do you expect to see as a solution?

Self-consistent band structure for a 9.0 nm long Bi$_2$Te$_3$ quantum well

- NEMO5 will produce four .dat files (SS12_TI_ex2_*_.dat) and a .xy file (poisson_ex2.xy)
- Start MATLAB on your workspace
- Your folder has a MATLAB file called `spin_analysis_ex2.m`
- Execute the MATLAB script by typing `spin_analysis_ex2` at the command prompt
- You will have the figure on your left!
Electrostatic calculations

• Use the **poisson_ex2.xy file** (three column file) in your folder to plot the charge and potential profile

• Start MATLAB on your workspace

• Type the following for the charge & potential plot:

```matlab
 cp = load('poisson_ex2.xy');

% potential plot
figure(1)
plot(cp(:,1), cp(:,2))

% charge plot
figure(2)
plot(cp(:,1), cp(:,3))
```

You can also use any plotting software:

Please remember: Column 1 is device coordinate followed by potential and charge data on columns 2 and 3
How do the results from Exercise I and II look like?

Self-consistent electronic structure

Schrödinger 20 band tight binding

Poisson calculation has large impact:
- Energy separation between Dirac cones gets enhanced
- Fermi velocity of Dirac states changes (mobility)
- Dirac points move below the Fermi level, into the bulk DOS
Self-consistent electrostatic calculations

Bi terminated surf.
- Smooth perturbative potential
- Oscillations in the potential between anion/cation covered by bulk parameters
- Change of charge polarity at two surfaces due to different atomic termination

Common in TB:
Charge oscillations between atom types

Te terminated surf.

Intrinsic dipole due to distinct surfaces
- Band structure calculations
  - Can handle the newest materials like topological insulators

- Charge self-consistent band structure calculation
  - For accurate device prediction

- Use of solvers to accomplish modular tasks
  - As many solvers as needed can be added

- Solvers can be inter-linked
  - Simple process accomplished by inserting appropriate solver name

Thank you.