Chapter 1 – Introuction

PiLab (π laboratory) is a set of functions that implement PiLib (π library) to compute condensed matter physics related problems using tight-binding calculations. Its core part is developed by an open source script language Scilab and will be released as a toolbox in the future. Scilab has built-in graphic user interface and powerful applications in numerical and graphical problems, so one can further process and visualize the results in a unified environment. Python version using Numpy and Matplot is also developing and will be announced later.

The current version (1.0-beta) of PiLab has the following features:

- Basic:
  - lattice structure construction
  - hopping parameters using Slaster-Koster integrals
  - TB+U self-consistent calculations
  - compatible with 1D, quasi-1D, 2D, quasi-2D and 3D
- Application:
  - band structure
  - density of state
  - finite size and impurity
  - Floquet band structure
  - Phonon modes
  - Quantum Transport
  - topological invariant

Chapter 2 – Instillation

In the following, we will use the notation: "→" to represent a command in Scilab console. For example: → this is a book , means type "this is a book" in the console.

To install PiLab, all you have to do is to install Scilab. The latest version can be found here: http://www.scilab.org/ . You can follow its instruction to install it in your computer easily. Once it is install, just download PiLib, unzip it, and put it in any folder you like. Then we will set Scilab environmental variables to this folder:

- find Scilab.start
  - Scilab.start can be found in your installed Scilab folder. For PC users, it is usually located in C:\Program Files\scilab-5.5.0\etc\ . For Linux or Mac users, it is usually located in /scilab-5.5.0/share/scilab/etc/
- set PiLib path
  - open Scilab.start (administrator authority may needed) and search for the keyword:
    " Protect variable previously defined "
  - set the PiLib path just below the keyword and above the predef('all') command. For example:
    PiLib_path='C:\Users\pipidog\Dropbox\PiLib\'
    PiLib=PiLib_path+'PiLib_loader.sce'
  - Note that, only the path can be changed. All the others should keep the same. For example, if you put your PiLib is in: /home/user/work/ , the you should type:
    PiLib_path='/home/user/work/PiLib/';

PiLib=PiLib_path+'PiLib_loader.sce';

- To check your installation, open Scilab and type → exec(PiLib); If there is no any error message, you are all set !

3. Framework

PiLab is a modular program. It calculates everything step by step and you can always extract the calculated results and import them to any other code easily. The PiLab has a layer structure:

Layer-1	Layer-2	Layer-3	Layer-4	Layer-5
			_ban	
			_dsa	
_lat	_hop	_scc	_imp	
			_qtp	_qtp_crt
			…..	….

You cannot perform any calculation if its previous layers were not calculated first. For example, you cannot call the _ban module before the _lat, _hop and _scc were calculated. As for layer-5, it is an extension layer. All the functions defined in this layer are an extension of a particular function in layer-4, so we will name the functions in this layer with two module names by its father module's name and its own name.

All the calculations are controlled by an input file named by: "project_task.plb". "project" is the name of your project (can be anything), "task" is the module you want to call and "plb" is the file name extension of PiLab. For example, if one has a project "NaCl" and want to calculate the lattice structure, you should prepare a file named "NaCl_lat.plb" with appropriate input commands in it. Then PiLab will read its inputs and perform lattice structure calculation. Once the calculation is done, PiLab will attach its calculated results in the same file. Here we introduce each layer in their details.

- Layer-1: lattice structure generation (_lat module)
  - This layer generates all lattice structure information.
- Layer-2: hopping integral generation (_hop module)
  - This layer generates the hopping integrals of the systems, select tight-binding states, onsite energies and LS coupling.
- Layer-3: self-consistent calculation (_scc module)
  - This layer perform TB+U self-consistent calculations, Fermi energy, and band gap.
- Layer-4: Application layer (_ban, _dsa, _imp, …, etc)
  - This layer constitutes many modules. Each module can calculate different properties. For example: _ban module can calculate band structure, _dsa calculates the density of state, etc.
- Layer-5: Extension layer (_qtp_crt, …, etc)
  - This layer contains many modules that can further calculate other properties based on the calculation results of layer-4. Unlike layer-4 where all the modules require all layer-1 ~ layer-3 modules are calculated. Layer-5 only requires its father module is calculated. For example, _qtp calculates the transmission function of a quasi-1D transport problem and _qtp_crt can take an integral over the transmission function to obtain the current v.s voltage

properties. To do it, only _qtp is required. All the other layer-4 modules are not necessary. Also, because it is just an extension of a particular module in layer-4, not all modules in layer-4 have a layer-5 son module.

To call a particular module, you should prepare a file with name: "project_task.plb". PiLab will automatically read the input parameters of project_task.plb and all the inputs and calculated results in its lower layer, e.g NaCl_scc.plb will not only read NaCl_scc.plb but evertything in " NaCl_lat.plb" and "NaCl_hop.plb". So if you change any input or output results in those files, you should expect something different. Once the _scc calculation is done, PiLab will attach the calculated results in the file for the use of next layer. Therefore, to learn PiLab, all you have to do is to understand the command of each input and the output results.

Chapter 3 – A " Hello, World! " code

Before go through the details of PiLab, we will first calculate the band structure of a very simple system-- graphene. So users can have a quick tour of PiLab.

We will need to generate the lattice structure of graphene. To this end, you should prepare a working fold and change the current Scilab working fold to that folder. It can be achieved by using the built-in Scilab file browser or do it in the console by typing, say : → cd('c:\User\pipidog\working') . Then let's generate a template file by typing the command in Scilab console: → PiLab_create('graphene','lat')
If it warns you error message: "Undefined variable: PiLab", it means you has not load PiLab in the Scilab memory. To solve this, type: → exec(PiLib); , the semi-colon " ; " at the end of the command is to tell Scilab not to show interactive message. If you don't care about it, you can neglect the semi-colon.