AMDKIIT-v1.0.0

User Manual

April 24, 2024

Contents

1 Introduction

AMDKIIT is a density functional theory program package based on plane-wave basis sets for performing ab initio molecular dynamics at higher rungs of density functionals. This program is getting developed at the Indian Institute of Technology Kanpur through a funding received from the National Supercomputing Mission and Center for Development of Advanced Computing, Pune, India.

This user guide provides a general concept for using AMDKIIT, a new tool for molecular simulations using ab initio methods. This electronic structure program is built on the principles of Kohn-Sham Density Functional Theory (KS-DFT) along with Plane Wave (PW) basis sets and pseudoptentials. The manual contains detailed information about user-selected parameters in the input file.

1.1 Key Features

- Wavefunction Optimization
- Geometry Optimization
- Molecular Dynamics

1.2 Libraries

The framework is supported by essential libraries:

- **FFTW** (Fastest Fourier Transform in the West): Experience swift and efficient Fourier transforms, fundamental to many computational tasks in AMDKIIT.
- **BLAS** (Basic Linear Algebra Subprograms) & **LAPACK** (Linear Algebra PACKage): These libraries provide the computational backbone, handling linear algebra operations with reliability and speed.
- **MPI** (Message Passing Interface): AMDKIIT seamlessly integrates with parallel computing platforms, enhancing performance and scalability through efficient data exchange.
- **LIBXC** (Library of Exchange and Correlation functionals): For exchange-correlation functionals, AMDKIIT relies on the comprehensive capabilities of LIBXC, ensuring accurate and diverse options for your simulations.

1.3 Contributors

- Prof. Nisanth N. Nair, IIT Kanpur
- Dr. Paramita Ghosh, IIT Kanpur
- Dr. Sudhir K. Sahoo, IIT Dharwad
- Ms. Ritama Kar, IIT Kanpur

1.4 Contact

For any kind of query, send a mail to amdkiit@gmail.com. Contribution is welcome.

Copyright: This is an open-source (MIT licensed) project.

2 Installation

2.1 Download

To download the code go to the `GitHub` site, `https://github.com/AMDKIIT/amdkiit` and download the `tar` file in your system.

For source compilation, extract the archive using the usual command:

```
tar -xzvf amdkiit-main.tar.gz
```

The additional packages require to compile the code can be downloaded from different sites as mentioned in the next section.

2.2 Prerequisite

To install AMDKIIT software:

- User should have a minimal unix environment. Installation using the source code requires **Fortran** compiler (GNU or Intel). For parallel environment user should use MPI libraries and parallel compiler.
- Install `cmake` (version 3.4 or higher) to compile the code.
- yaml-cpp (`https://github.com/jbeder/yaml-cpp`) should be downloaded and copied inside the source directory.
- Install the libraries from the individual sites:
  - FFTW: Download and intstall from `http://www.fftw.org`
  - BLAS and LAPACK: Download and install from `https://netlib.org/lapack`
  - LIBXC: For exchange-correlation functional, download the latest library from `https://libxc.gitlab.io/`

2.3 Installation

To install the package, user should edit the path for lapack, fftw3 and libxc library in the `CMakeLists.txt` file inside `source` directory.

- `set(MATH_ROOT "path to the lapack library")`
- `set(FFTW_ROOT "path to the fftw directory")`
- `set(LIBXC_ROOT "path to the libxc directory")`

Instructions to `make` the executable:

```
$ mkdir build

$ cd build

$ cmake ..

$ make
```

This will generate the executable file `amdkiit.x` inside the `build` directory.

3 How to Run?

3.1 Input Files

To run a job using AMDKIIT, user needs three types of input files.

3.1.1 Input Coordinate File

The geometry of the system has to be provided in the form of `xyz`. User can build the structure in any software like GaussView, Avogadro, Molden etc. and save with `.xyz` extension.

3.1.2 Pseudopotential Files

AMDKIIT software is PW based electronic structure theory code with pseudopotentials. To use plane wave, user needs to provide the correct pseudopotential for different atoms. Currently norm-conserving pseudopotentials are enabled. It has to be built for the XC functional which are being used during the calculation. Only PSP / UPF format can be used. User can download pseudopotentials from:
`http://pseudopotentials.quantum-espresso.org/legacy_tables`.

3.1.3 Input

The main input file is in `yaml` format. User can choose the filename with the `yaml` extension.
Note:

1. Every keyword will be in upper case

2. For each keyword, there will be always a gap after colon

3. For a new dictionary, there will be at least one gap in the next line to start the new keyword or new sub-dictionary

The details of the keywords are described in chapter 4.

3.2 Commands

To run the job in serial, use the following command.

{path to the executable}`/amdkiit.x input.yaml > amdkiit.out`

For parallel machine, use the script file with number of cores user wants to use. Then, run the job using:

`mpirun -n` N_{core} {path to the executable}`/amdkiit.x input.yaml > amdkiit.out`

3.3 Output Files

The general out file contains all the information of the job being performed. Apart from that, different calculations AMDKIIT generates different output files for different type of calculations as mentioned here.

3.3.1 Wavefunction Optimization

- `SP_ENERGY.dat`: KS energy (in **Hartree**) during self-consistent field (SCF) calculations with CPU time per step is printed.

3.3.2 Geometry Optimization

- `atom_coord.xyz`: All ionic coordinates (in **Å**) are appended during geometry optimization in `xyz` format. Users can visualize the structure using any visualizing software like VMD.
- `atom_force.dat`: Forces along the geometry optimization is getting stored here in XYZ format. The KS energy along optimization is also printed at every step in this file.

3.3.3 Molecular Dynamics

- `MD_ENERGY.dat`: Instantaneous temperature (in **K**), KS energy (in **Hartree**), total energy (in **Hartree**), and CPU time in seconds per MD step is appended along dynamics.
- `MD_TRAJECTORY.xyz`: Geometry of the system (in **Å**) at every MD step is printed here in append mode. Users can visualize the structure using any visualization software like VMD.

4 Input File References

4.1 General Keywords

The basic information for SCF job control with the required parameters are listed here. This section contains the keywords which has to be provided to perform any tasks. Some of the listed keywords are optional with default values, which are mentioned in the following.

TASK:

This part defines what type of calculation will be performed. This section of the input file is **mandatory**.

- `WAVEFUNCTION_OPTIMIZATION`
  - to perform single-point energy calculations
- `GEOMETRY_OPTIMIZATION`
  - to optimize the molecular geometry
- `MOLECULAR_DYNAMICS`
  - perform molecular dynamics simulation.

INITIALIZATION:

This part of the input file is not mandatory. The default choices are described here. Additional information regarding the initialization can be specified using the following options.

- `WAVEFUNCTION: ATOMIC`
  - This option refers to to initial guess for wavefunction optimization. By default, the initial wavefunction is derived from the pseudo atomic orbitals (`ATOMIC`).
- `HESSIAN: UNIT`
  - This option refers to the initial approximate Hessian for a geometry optimization. By default, Hessian is initialized as a unit matrix (`UNIT`).
- `OPEN_SHELL: .FALSE. [.TRUE.]`

  - **TRUE** condition allows the spin-polarized DFT calculations. By default, it will perform closed-shell calculations.

- `MULTIPLICITY: 1`
  - Provide the multiplicity of the system. Default is 1.

OPTIMIZATION:

User can define this dictionary to customize the minimization technique and other related parameters in performing SCF calculations. Default values can be used otherwise. Additional sections will be described later.

- `WAVEFUNCTION:`
  - `MINIMIZER: CONJUGATE_GRADIENT`
    - This option specifies the algorithm that will be used to optimize the wavefunction of the system. By default, **conjugate gradient** is used.
  - `MAX_STEPS:`
    - maximum number of steps for wavefunction optimization is specified. Default is **800**.
  - `GRAD_CUTOFF:`
    - This parameter specifies the maximum allowable value for the largest element in the gradient of the wavefunction. Default convergence criteria for each SCF step is 10^{-7}.

This sub-dictionary is used only for wavefunction optimization. However, for geometry optimization different sub-dictionary can be added as discussed in section 4.2.

CUTOFF:

This dictionary contains the energy cutoff for the PW basis. If this dictionary is not mentioned in the input file, default will be used.

- `VALUE:`
  - specify the energy cutoff value to expand the PW basis set. Default value is **80**.

- `UNIT: RYDBERG`
  - specify the unit of the PW cutoff. By default, **Rydberg** is used.

SYMMETRY:

Symmetry of the supercell **has to be defined** here. The available options are listed below.

- `CUBIC`
- `HEXAGONAL`
- `FCC`
- `BCC`
- `TRIGONAL`
- `TETRAGONAL`
- `BODY CENTRED TETRAGONAL`
- `ORTHORHOMBIC`
- `MONOCLINIC`
- `TRICLINIC`

CELL:

Information regarding supercell size is **mandatory**.

- `UNIT: BOHR [ANGSTROM]`
  - This section specify the units of the supercell dimension. By default, the dimensions are read in **Bohr**.
- `DIMENSION:`
  - This section specify the dimensions of the supercell in the three directions, a, b and c.
- `ANGLE:`
  - This section specify the angles of the supercell in the three directions, α, β and γ in **degrees**.

INPUT:

- `COORDINATES_FILE:` *`filename`*
  - The name of the input coordinate file **has to be given** in this section.
- `UNIT: BOHR [ANGSTROM]`
  - specifies the unit of the coordinates provided in the coordinate file. Default is **Bohr**.

SPECIES:

- `LABEL:` to assign a label to each species. In cases where there are multiple species, each species will be labeled as **LABEL1**, **LABEL2**, and so on.
  - `ATOM:` specify the atom indices
  - `PP_FILE:` provide the corresponding pseudopotential file name
  - `PP_SCHEME:` mention if there is any scheme used
    - specify the method to calculate the nonlocal part of the pseudopotential.
  - `LMAX:` provides the information on the nonlocality of the pseudopotential, maximum l.

OUTPUT:

This dictionary is not mandatory. User can mention this for customization of output files.

- `DEBUG: .TRUE. [.FALSE.]`
  - The keyword **TRUE** prints the output results in details, where as **FALSE** will print only the basic quantities.
- `PRINT_DENSITY: .TRUE. [.FALSE.]`
  - The keyword **TRUE** prints the density in cube format. Currently, this part is only enabled for serial run.

DFT:

This dictionary is **mandatory** in the file.

- `EXCHANGE:` mention exchange functional name here.
- `CORRELATION:` user can mention correlation functional name here.
- `XC:` instead of mentioning `EXCHANGE` and `CORRELATION` separately, user can define the exchange-correlational functional to be used in the calculation. The functionals are available in **libxc**.

4.2 Task Specific Keywords

In this section, different task related parameters are described. All the keywords have to be used along with the general keywords mentioned in section 4.1.

GEOMETRY:

If the **TASK** is `GEOMETRY_OPTIMIZATION` as mentioned in `TASK` dictionary, this section can be added inside `OPTIMIZATION` dictionary. User can customize the technique and the other related parameters. In this version only **LBFGS** is implemented. The following additional keywords are given here.

- `MINIMIZER: LBFGS`
  - This option specifies the algorithm that will be used to optimize the molecular geometry of the system. By default, **LBFGS** is used.
- `MAX_STEPS:`
  - maximum number of steps for geometry optimization is specified. Default is 200.
- `GRAD_CUTOFF:`
  - The convergence criterion on the gradient. The default value is 10^{-4}.

DYNAMICS:

If the **TASK** is `MOLECULAR_DYNAMICS` as mentioned in section 4.1, user can specify this dictionary. Otherwise, all the values will be taken as default. As of now, only **NVE** simulations can be performed using AMDKIIT. The following keywords to perform the MD simulations are given here.

- `TEMPERATURE`
  - provide the initial temperature in **kelvin**. Default is 300 K.
- `MAX_STEPS:`
  - maximum number of MD steps is specified. This defines the length of the simulation. Default is 999.
- `TIMESTEP:`
  - The time step for ion dynamics are given in **femtosecond** unit. Default is 1.0 fs.