MateriApps Installer Documentation

Release 1.0

MateriApps Installer Development Team

Mar 13, 2021

CONTENTS

CHAPTER

ONE

WHAT IS MATERIAPPS INSTALLER ?

1.1 Overview of MateriApps Installer

MateriApps Installer is a collection of shell scripts to assist in the installation of computational material science applications for various computing environments. The MateriApps Installer includes shell scripts to be installed on Linux PCs, cluster workstations, and major supercomputer systems. MateriApps Installer is also used for pre-installation for domestic joint-use supercomputers (a list of installed software can be found in Chapter of Application List).

1.2 Background of the Development

Nowadays, computer numerical computation is indispensable to promote theoretical research in materials science. For advancement of computational materials science, the development of algorithms to solve equations of materials science efficiently plays an important role, and many excellent applications with efficient algorithms, excellent parallel performance, and state-of-the-art functions have been created. In 2013, we launched a portal site for materials science simulations, MateriApps, in order to disseminate information about the developed software to experimentalists and corporate researchers. I have been disseminating information about the application.

One of the obstacles for users to start using published applications in materials science is the installation of software. MateriApps LIVE! is an environment that allows users to easily try out computational materials science applications on their laptops and other devices. MateriApps LIVE! is a Virtual Hard Disk Image (OVA) of VirtualBox that includes applications, OS (Debian GNU/Linux), editors, visualization tools, and other environments needed to get started with the tutorial. By using MateriApps LIVE!, it is possible to easily set up a computing environment for participants in classes and software training sessions.

However, the calculation environment provided by MateriApps LIVE! is not sufficient for practical applications. To enable users to easily install applications of computational materials science in a wide range of computing environments, from major domestic joint-use supercomputers to clustered computers in laboratories and personal computers, the development of MateriApps Installer has been started.

1.3 Goal of MateriApps Installer

The goals of MateriApps Installer are as follows

- To install common applications (cf. MateriApps) on all major domestic and foreign supercomputers.
- Similarly, we will prepare scripts for installation on CentOS (RedHat), Debian (Ubuntu) and macOS environments.

1.4 List of tools and apps

The following tools and apps are in place (12/4/2020). `default.sh` is prepared in the `config` directory of each tool or application. In the following table, the supported compilers except `default.sh` are listed. :

1. Tools

name	intel	macos	gcc
boost	o	o	x
cmake	x	x	x
eigen3	x	x	x
fftw	o	o	x
gcc10	x	x	x
gcc8	x	x	x
git	x	x	x
gsl	o	x	x
hdf5	x	x	x
julia	o	x	x
lapack	x	x	x
libffi	x	x	x
openmpi	o	x	o
openssl	x	x	x
python3	o	o	x
scalapack	x	x	x
tcltk	x	x	x
zlib	x	x	x

2. Apps

name	intel	macos	gcc
ALPS	o x	o	
ALPSCore	o	o	o
DSQSS	o	x	o
QUANTUM ESPRESSO	o	x	x
HPhi	o	x	o
KOmega	o	x	x
LAMMPS	o	x	o
mVMC	o	x	o
OpenMX	o	x	x
RESPACK	o	x	x
TeNeS	o	x	x

3. Apps to be added in the future

   abICS, DCore, TRIQS

1.5 Design Policies

The design policy of MateriApps Installer is as follows:

1. Do not depend on special tools (shell, make, tar, etc. are sufficient).
2. Supercomputers are different from each other, so do not aim to make a universal installer, but make a separate script for exceptions.
3. As for the tools, if the package is already available for Linux (RPM Package, Debian Package) or macOS (Homebrew, Fink, MacPorts), use it.
4. The version of the application should be the same as MateriApps LIVE! Necessary patches should also be the same as in Debian Package for MateriApps LIVE!
5. Use a separate folder for each tool/application. Prepare a separate folder for each tool/application.
    - For example, in the case of cmake

```
$PREFIX_TOOL/cmake/cmake-3.2.1-1
```

6. Prepare an environment variable setting script for each tool/application/version.
    - Example for cmake

```
$PREFIX_TOOL/cmake/cmakevars-3.2.1-1.sh.
```

7. For the environment variables of the tools, link to `PREFIX_TOOL/env.d` so that we can keep the old version for collective setting in `PREFIX_TOOL/env.sh`.
8. Separate installation and relinking (it does not affect anything else until the relinking is done).
9. Install to a different location for testing.

1.6 Main developers

MateriApps Installer is developed by the following members.

- **ver. 0.1 -**
    - Synge Todo (Department of Physics/Institute for Solid State Physics, The University of Tokyo)
    - Yuichi Motoyama (Institute for Solid State Physics, The University of Tokyo)
    - Kazuyoshi Yoshimi (Institute for Solid State Physics, The University of Tokyo)
    - Takeo Kato (Institute for Solid State Physics, The University of Tokyo)

1.7 Version history

- 2021/03/13 ver. 1.0 was released.
- 2020/12/04 ver. 0.1 was released.

1.8 License

The program package and source code set of this software is distributed under the GNU General Public License version 3 (GPL v3). However, the patch files for each software are distributed under the license of the software.

1.9 Copyright

The University of Tokyo holds the copyright of MateriApps Installer, and it is distributed under the GNU General Public License version 3 (GPL v3). The patch files for each installed software are subject to the license of the respective software.

The development of the MateriApps Installer has been supported greatly by PASMUS software development project in FY2020 by Institute for Solid State Physics, the University of Tokyo.

CHAPTER

TWO

USAGE

2.1 Download

- You can download MateriApps Installer by the following steps.
  - Download the release version

    Go to the MateriApps Installer release page to download the zip file and then extract it. The zip file you download from the release page includes a pdf version of this manual.
  - Download with git

    You can download the MateriApps Installer by typing the following command.

    ```
    git clone https://github.com/wistaria/MateriAppsInstaller.git
    ```

2.2 Directory Structure

- The structure of the directory after extraction is as follows.

```
|– setup
|– apps
|– docs
|– tools
|– check
|   |– k.sh
|   |– macos.sh
|   |– sekirei.sh
|   |– zetta-gcc.sh
|   |– zetta-intel.sh
|– check_prefix.sh
|– fix_dylib.sh
|– list_maversion.sh
|– macosx
|   |– install.sh
|   |– ports.sh
|– README.md
|– util.sh
```

- The directory structure in setup, tools, and apps is given as follows.

```
-- software_name
      |- README.md
      |- download.sh
      |- link.sh
      |- setup.sh
      |- version.sh
      |- install.sh
      |- patch
      |- config
```

Each file and directory is described below (see File Format for details). Files marked with * indicate files that always exist in the directory.

  - README.md (*)
    - It includes a brief introduction of the software and the URLs of the official website.
  - download.sh (*)
    - Download the source code archive
  - link.sh (*)
    - Create symbolic links to installed directories and configuration files
  - setup.sh (*)
    - Extract the prepared source code archive and apply the patch (if it exists)
  - version.sh (*)
    - Specify the version to download
  - install.sh (*)
    - Building and installing the program
  - patch
    - The directory where the patches are stored
  - config
    - Additional settings for installation other than the default settings, such as when using the Intel Compiler

- In addition to the above, the following file directories are also available.
  - check_prefix.sh
    - Script to display variables that are commonly used in each script, such as the top installation directory
  - list_maversion.sh
    - A script that summarizes the information of `version.sh` in each directory
  - check directory
    - A script to run multiple installation scripts in sequence on various hosts
  - docs directory
    - A directory containing the manual and its source code
  - macosx directory
    - A directory containing scripts to install the necessary tools using Macports

- scripts directory
  - A directory containing a set of administrative scripts
- setup directory
  - A directory containing scripts to prepare for software installation (see Setup below for details)

2.3 Setup

- Run `setup/setup.sh` before installing the software

```
sh setup/setup.sh
```

This script creates installation scripts, installation scripts and working scripts

- Configuring the installation location for applications extracted by the MateriApps Installer
  - You can change the installation location by setting the following options in `$HOME/.mainstaller`.

    The installation location can be set in the `$HOME/.mainstaller` file as follows (you have to create it yourself)

    # Do not put spaces before or after = as it will be treated as a shell script

    MA_ROOT=$HOME/materiapps # Software installation directory BUILD_DIR=$HOME/build # Installation directory SOURCE_DIR=$HOME/source # File download directory

    Table 2.1: Explanation of options :header: “option”, “default”, “description” :widths: 15, 15, 30

MA_ROOT	$HOME/materiapps	Software installation directory
BUILD_DIR	$HOME/build	Installation directory
SOURCE_DIR	$HOME/source	Source code archive file download directory

  - If this file does not exist, the software will be installed under `$ HOME materiapps`
  - (*) Note that the actual installation location uses the contents of the `.mainstaller` file at the time of the installation work described below.

2.4 Install

- Move to each software directory and run `` install.sh``.

```
 sh install.sh


- When this script is executed, the build and installation will be performed␣
↪automatically after downloading (``download.sh``) and extracting (``setup.sh``)␣
↪the source code.
- Depending on the software, settings for the compiler and libraries may have␣
↪been defined, and they are stored as subdirectories under the ``config``␣
↪directory.
```

(continues on next page)

(continued from previous page)

```
- If you want to specify the compiler, you can add the compiler name after␣
↪``install.sh``. The following is an example of compiling with ``intel``.

  .. code-block:: bash

     sh install.sh ``intel``

    - If you specify a settings directory that does not exist, a list of␣
↪available settings is displayed.::

      $ sh install.sh help
      Error: unknown mode: help
      Available list:
      default
      intel

  - ``default``

    - Basic settings used when the argument is omitted

  - ``intel``

    - Settings for using Intel compiler, Intel MKL, Intel MPI

-  The compiler etc. can be directly specified using shell variables

  - ex.) A case of using the Intel compiler as the C compiler while using the␣
↪default settings ::

    CC=`which icc` sh install.sh

  - Compiler options can be added by setting ``MA_EXTRA_FLAGS`` ::

    MA_EXTRA_FLAGS="-march=core-avx2" sh install.sh intel

  - The path of the ``cmake`` command can be specified using ``CMAKE``

  - ``ISSP_UCOUNT`` is the path of the utilization rate measurement script in␣
↪Supercomputer on Institute for Solid State Physics, and most users do not have␣
↪to worry about it

  - For other variables available, see the beginning description of ``install.
↪sh``.
```

- Run a simple test with `sh runtest.sh`
  - Check the existence of the installation directory
  - Check the validity of the configuration file
  - Check if the software actually works
- The software is installed in the `$ MA_ROOT/NAME/NAME-VERSION-MA_REVISION` directory
  - `NAME` and `VERSION` are replaced with the software name and version respectively
    - `MA_REVISION` is an identifier that distinguishes when the MateriApps Installer is revised for the same version of software.

    * ex.) `hphi/hphi-3.4.0-1`

  – Along with the software, the configuration file `NAMEvars-VERSION-MA_REVISION.sh` that sets environment variables etc. is installed in `$MA_ROOT/NAME/`

    * ex.) `hphivars-3.4.0-1.sh`

    * Running `sh link.sh` creates a symbolic link `NAMEvars.sh` for `NAMEvars-VERSION.sh`

      · For apps, it is created under `NAME`

      · For tools, it is created under `$MA_ROOT/env.d` and loaded in `$MA_ROOT/env.sh`.

2.5 How to use the tools and apps

- Tools (cmake, hdf5, python, etc.)
  – Run the following command (or write the same command in a shell initialization script)

```
source $PREFIX_TOOL/env.sh
```

- Applications (ALPS, OpenMX, MODYLAS, etc.)
  – Set environment variables (e.g. `PATH`) for each application using scripts.

    For example, in the case of ALPS:

```
source $PREFIX_ALPS/alps/alpsvar.sh
```

  – If you want to fix the version, use the configuration file of that version.

```
source $MA_ROOT/alps/alpsvar-20201106-r7860-1.sh
```

CHAPTER

THREE

TUTORIAL

3.1 Installation to MacOS 10.15 (Catalina)

This section describes how to use the MateriApps Installer on MacOS 10.15 (Catalina).

3.1.1 Installing Tools

It is more convenient to install the tools in the tools directory using the MacOS package management software (Homebrew, find, macports, etc.). Here, we show examples using Homebrew. In this case, we use the default setting for directory of the application. The install directory is $HOME/materiapps and the build directory is $HOME/build.

Follow the instructions at https://brew.sh/ to install Homebrew.

```
$ /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/
↪master/install.sh)"
```

Then install the tools via Homebrew. If the installation has already been done by another method, you do not need to install it in Homebrew. You can install all of the following tools first, or you can install them when required by the app installation.

```
$ brew install gcc
$ brew install boost
$ brew install cmake
$ brew install eigen
$ brew install fftw
$ brew install git
$ brew install gsl
$ brew install hdf5
$ brew install lapack
$ brew install libffi
$ brew install openblas
$ brew install openmpi
$ brew install openssl
$ brew install python@3
$ brew install scalapack
$ brew install tcl-tk
$ brew install zlib
```

Some applications also require you to install the following tools:

```
$ brew install svn
$ brew install boost-python
```

(continues on next page)

(continued from previous page)

```
$ brew install boost-python3
$ brew install boost-mpi
$ brew install wget
```

In addition, some applications require you to install the python library. Install it with the following command:

```
$ pip3 install numpy --user
$ pip3 install scipy --user
$ pip3 install toml --user
```

3.1.2 Installing Applications

Perform initial setup (Create required directories, etc.). Go to the MateriAppsInstaller directory, and run the following command.

```
$ sh setup/setup.sh
```

Next, enter the directory of the application you want to install and run the following command to install it.

```
$ CC=gcc-10 FC=gfortran-10 CPP=cpp-10 sh install.sh
```

You can check whether it is installed correctly by executing the following command in the directory of each application.

```
$ sh runtest.sh
```

(You must install quantum ESPRESSO before running runtest.sh for respack. You must also set up an execution environment for quantum ESPRESSO as described in the next section.)

If you pause in the middle of downloading, the source files may remain, which may not work when you reinstall. In this case, delete all the target application directories in the source file (see the directory $HOME/materiapps/source in the default setting).

If you get errors during installation, take a closer look at the error messages. Errors often occur because the necessary tools are not installed. Read the required tools from the error message and install the tools.

3.1.3 Setting the execution environment

In order to run the application, you need to set the execution environment. By entering the following commands, you can create an environment in which each application can be executed. For example, if you want to run HΦ that is already installed, then output the filenames for the setting file as follows.

```
$ ls $HOME/materiapps/hphi
```

You will find the file named “hphivars.sh”. Execute this setting file as follows.

```
$ source $HOME/materiapps/hphi/hphivars.sh
```

Then, you are now ready to run the applications.

3.2 Installation to Linux (using gcc)

This section describes how to use the MateriApps Installer on Linux OS. We have checked it on CentOS 7 and Ubuntu 20. (You should be able to install other distributions using the same procedure.) Tools (gcc, cmake, etc.) preinstalled in Linux OS may be used, but if any trouble occurs, you have to estimate its reason from the error message and have to solve the trouble by your self. If you are not sure of doing this, we recommend that you install all the tools through MateriApps Installer.

3.2.1 Initial setup

Perform initial setup. Enter the root directory of MateriApps Installer and run the following command:

```
$ sh setup/setup.sh
```

You only need to do this once at the beginning.

3.2.2 Installing gcc

Install the compiler first. (It is also possible to use the gcc compiler already installed in Linux OS to install subsequent tools, but some applications may not install well with the older gcc compiler. If you are not familiar with troubleshooting, it is recommended that you use the gcc compiler with MateriApps Installer.) To install gcc 10, execute the following commands:

```
$ cd tools/gcc10
$ sh install.sh
```

Installing gcc10 takes a long time. If you are installing to a remote computer server, it may be helpful to run the last command in the background with the following changes:

```
$ sh install.sh > log &
```

In this case, the progress of the installation can be seen in the log file generated in the same directory (For example, run “cat log”). If the installation is successful, execute the following command:

```
$ sh link.sh
```

This copies a link to the script that initializes the tool usage to the given location (default: $HOME/materiapps/env.d). Finally, run the copied configuration file and make it available for subsequent tools and applications.

```
$ source $HOME/materiapps/env.sh
```

If you have successfully installed gcc 10, you do not need to install gcc 8. Rarely, installation of gcc 10 may fail on older Linux OSs. Install gcc8 only in this case. The installation procedure is the same as gcc10, except that you first enter tools/gcc8 directory.

3.2.3 Installing cmake

Some applications use cmake to install. Major Linux distributions already include cmake, but older versions of cmake may fail to install some applications. Versions 3.6 and later are fine. To install cmake, run the following command immediately after installing the gcc compiler:

```
$ cd ../cmake
$ sh install.sh
$ sh link.sh
$ source $HOME/materiapps/env.sh
```

3.2.4 Installing other tools

Install the remaining tools in the same way. For example, if you want to install git immediately after you install cmake, run the following command:

```
$ cd ../git
$ sh install.sh
$ sh link.sh
$ source $HOME/materiapps/env.sh
```

To install all the tools, please first install git, libffi, and python3 in this order, and then install the remaining tools in alphabetical order (boost, eigen3, fftw, gsl, hdf5, lapack, libffi, openmpi, openssl, scalapack, tcltk, zlib). (You must install openmpi first before installing scalapack.) The command for installation is identical except that you first enter the directory, which is the name of the tool to be installed.

3.2.5 Installing applications

After installing the tools, enter the directory of the application you want to install, set the execution environment of the tools, and execute the installation script:

```
$ sh $HOME/materiapps/env.sh
$ sh install.sh
```

You can check whether it is installed correctly by executing the following command in the directory of each application:

```
$ sh runtest.sh
```

(You must install quantum ESPRESSO before running runtest.sh for respack. You must also set up an execution environment for quantum ESPRESSO as described in the next section.)

If you pause in the middle of downloading and installing an application, the source files may remain, which may not work when you reinstall. In that case, delete the directory of the target application in the source file ($HOME/materiapps/source by default).

If you get errors during installation, take a closer look at the error messages. Errors often occur because the necessary tools are not installed. Read the required tools from the error message and install the tools.

Once you've successfully completed the test, use the following command to make a link of the preferences file of the application (see also the next section):

```
$ sh link.sh
```

3.2.6 Setting the execution environment for each application

In order to run the application, you need to set the execution environment. By entering the following commands, you can create an environment in which each application can be executed and the application can be executed immediately. For example, if you have installed HPhi (the installation directory is assumed to be the default, i.e., $HOME/materiapps), you should see a configuration file named hphivars.sh when you execute the following command:

```
$ ls $HOME/materiapps/hphi
```

Run this configuration file as follows to configure the execution environment.

```
$ source $HOME/materiapps/hphi/hphivars.sh
```

Then, you are now ready to use the application.

CHAPTER

FOUR

FILE FORMAT

4.1 Installation-related file

This section describes the format and rules for each file required for the installation of the software.

4.1.1 default.sh

A script for specifying the software version.

Variables sandwiched between two underscores are for use in other scripts.

- <NAME>_VERSION
  - Version number of software
- <NAME>_MA_REVISION
  - An identifier to distinguish between revised scripts of the MAInstaller for the same version.
- __NAME__
  - Software names used for directory names.
  - In MAInstaller, the name is written in lower case.
- __VERSION__
  - <NAME>_VERSION is specified.
- __MA_REVISION__
  - <NAME>_MA_REVISION is specified.

Example

```
TENES_VERSION="1.1.2"
TENES_MA_REVISION="0"

__NAME__=tenes
__VERSION__=${TENES_VERSION}
__MA_REVISION__=${TENES_MA_REVISION}
```

4.1.2 download.sh

A script to get the source code from the internet.

- The source code will be downloaded to `$SOURCE_DIR`.
  - Rename the saved file according to the format to reduce the difference between software as much as possible.
  - In particular, be sure to give the version number so that it can be distinguished.
- Skip if already downloaded
  - In the environment where external access is not possible, files can be placed manually.

Example

```
SCRIPT_DIR=$(cd "$(dirname $0)"; pwd)
. $SCRIPT_DIR/../../scripts/util.sh
. $SCRIPT_DIR/version.sh
set_prefix

if [ -f $SOURCE_DIR/${__NAME__}-${__VERSION__}.tar.gz ]; then :; else
  check wget https://github.com/issp-center-dev/TeNeS/archive/v${__VERSION__}.tar.gz -
↪O $SOURCE_DIR/${__NAME__}-${__VERSION__}.tar.gz
fi
```

4.1.3 setup.sh

A script for extracting archive files.

- Run `download.sh` and get the archive file.
  - `download.sh` will check if it has already been downloaded.
- Extract archive files in `$SOURCE_DIR` to `$BUILD_DIR`.
  - Rename the extracted directory according to the format.
- Apply patch files, if any, in the `patch` directory.

Example

```
SCRIPT_DIR=$(cd "$(dirname $0)"; pwd)
. $SCRIPT_DIR/../../scripts/util.sh
. $SCRIPT_DIR/version.sh
set_prefix

sh ${SCRIPT_DIR}/download.sh

cd $BUILD_DIR
if [ -d ${__NAME__}-${__VERSION__} ]; then :; else
  check mkdir -p ${__NAME__}-$__VERSION__
  tarfile=$SOURCE_DIR/${__NAME__}-${__VERSION__}.tar.gz
  sc=`calc_strip_components $tarfile README.md`
  check tar zxf $tarfile -C ${__NAME__}-${__VERSION__} --strip-components=$sc
  cd ${__NAME__}-$__VERSION__
  if [ -f $SCRIPT_DIR/patch/${__NAME__}-${__VERSION__}.patch ]; then
    patch -p1 < $SCRIPT_DIR/patch/${__NAME__}-${__VERSION__}.patch
  fi
fi
```

4.1.4 link.sh

A script that creates a symbolic link for a configuration file.

- The destination of the symbolic link is different for the application and the tool.
  - For software packages, `$MA_ROOT/${__NAME__}/${__NAME__}vars.sh`
  - Fot tools, `$MA_ROOT/env.d/${__NAME__}vars.sh`

Example

```
SCRIPT_DIR=$(cd "$(dirname $0)"; pwd)
. $SCRIPT_DIR/../../scripts/util.sh
. $SCRIPT_DIR/version.sh
set_prefix

. $MA_ROOT/env.sh

VARS_SH=$MA_ROOT/${__NAME__}/${__NAME__}vars-${__VERSION__}-${__MA_REVISION__}.sh
rm -f $MA_ROOT/${__NAME__}/${__NAME__}vars.sh
ln -s $VARS_SH $MA_ROOT/${__NAME__}/${__NAME__}vars.sh
```

4.1.5 README.md

The following information on the software is described:

- Software name
- Summary
- License
- Official page
- MateriApps URL

The information is taken from A portal site of Materials Science Simulation MateriApps.

Example

```
HPhi
====================

SUMMARY
-------

An exact diagonalization package for a wide range of quantum lattice
models (e.g. multi-orbital Hubbard model, Heisenberg model, Kondo
lattice model). HPhi also supports the massively parallel computations.
The Lanczos algorithm for obtaining the ground state and thermal pure
quantum state method for finite-temperature calculations are
implemented. In addition, dynamical Green's functions can be calculated
using , KOmega which is a library of the shifted Krylov subspace method. It
is possible to perform simulations for real-time evolution from ver. 3.0.

LICENSE
-------

GNU GPL version 3
```

(continues on next page)

(continued from previous page)

```
OFFICIAL PAGE
-------------

http://www.pasums.issp.u-tokyo.ac.jp/hphi/en/

MateriApps URL
--------------

https://ma.issp.u-tokyo.ac.jp/en/app/367
```

4.2 Management file (`scripts/`)

4.2.1 util.sh

A script for defining the functions used.

4.2.2 check_prefix.sh

A script for defining the functions used.

4.2.3 fix_dylib.sh

A script for modifying shared library name etc. on macOS

4.2.4 list_maversion.sh

A script for showing the versions of the various apps supported by the MateriApps Installer.

CHAPTER

FIVE

APPLICATION LIST

5.1 ALPS

5.1.1 SUMMARY

ALPS is a numerical simulation library for strongly correlated systems such as magnetic materials or correlated electrons. It contains typicalsolvers for strongly correlated systems: Monte Carlo methods, exact diagonalization, the density matrix renormalization group, etc. It can be used to calculate heat capacities, susceptibilities, magnetization processes in interacting spin systems, the density of states in strongly correlated electrons, etc. A highly efficient scheduler for parallel computing is another improvement.

5.1.2 LICENSE

ALPS Licence

5.1.3 OFFICIAL PAGE

http://alps.comp-phys.org

5.1.4 MateriApps URL

https://ma.issp.u-tokyo.ac.jp/en/app/217

5.2 ALPSCore

5.2.1 SUMMARY

Open-source software for building computational physics applications. Common C++ auxiliary modules required for various methods in computational physics such as the quantum Monte Carlo method are prepared. This software helps to build reusable codes and to reduce development time for complex computational science applications. It also supports parallel programming based on MPI or OpenMP.

5.2.2 LICENSE

The GNU General Public License (GPL), Version 2

5.2.3 OFFICIAL PAGE

https://alpscore.org/

5.2.4 MateriApps URL

https://ma.issp.u-tokyo.ac.jp/en/app/2873

5.3 DSQSS

5.3.1 SUMMARY

DSQSS is an application program for solving quantum many body problems in a discrete set (typically a lattice). It carries out quantum Monte Carlo simulations that sample from the Feynman path integral using the worm update. It can handle any lattice geometry and interaction.

5.3.2 LICENSE

This software is distributed with GNU General Public License (GPL) Version 3.

5.3.3 OFFICIAL PAGE

https://github.com/issp-center-dev/dsqss

5.3.4 MateriApps URL

https://ma.issp.u-tokyo.ac.jp/en/app/180

5.4 QUANTUM ESPRESSO

5.4.1 SUMMARY

Open-source program for first-principles calculation based on pseudo-potential and plane-wave basis. This package performs electronic-state calculation with high accuracy based on density functional theory. In addition to basic-set programs, many core-packages and plugins are included. This package can be utilized for academic research and industrial development, and also supports parallel computing.

5.4.2 LICENSE

GNU GPL v2

5.4.3 OFFICIAL PAGE

http://www.quantum-espresso.org/

5.4.4 MateriApps URL

https://ma.issp.u-tokyo.ac.jp/en/app/740

5.5 HΦ

5.5.1 SUMMARY

An exact diagonalization package for a wide range of quantum lattice models (e.g. multi-orbital Hubbard model, Heisenberg model, Kondo lattice model). HΦ also supports the massively parallel computations. The Lanczos algorithm for obtaining the ground state and thermal pure quantum state method for finite-temperature calculations are implemented. In addition, dynamical Green's functions can be calculated using , Kω which is a library of the shifted Krylov subspace method. It is possible to perform simulations for real-time evolution from ver. 3.0.

5.5.2 LICENSE

GNU GPL version 3

5.5.3 OFFICIAL PAGE

http://www.pasums.issp.u-tokyo.ac.jp/hphi/en/

5.5.4 MateriApps URL

https://ma.issp.u-tokyo.ac.jp/en/app/367

5.6 Kω

5.6.1 SUMMARY

Kω implements large-scale parallel computing of the shifted Krylov subspace method. Using Kω, dynamical correlation functions can be efficiently calculated. This application includes a mini-application for calculating dynamical correlation functions of quantum lattice models such as the Hubbard model, the Kondo model, and the Heisenberg model in combination with the quantum lattice solver of quantum many-body problems,

5.6.2 LICENSE

GNU LESSER GENERAL PUBLIC LICENSE Version 3

5.6.3 OFFICIAL PAGE

https://github.com/issp-center-dev/Komega

5.6.4 MateriApps URL

https://ma.issp.u-tokyo.ac.jp/en/app/546

5.7 LAMMPS

5.7.1 SUMMARY

A general-purpose open-source application for classical molecular dynamics simulation, distributed under the GPL license. This package can perform molecular dynamics calculation of various systems such as soft matters, solids, and mesoscopic systems. It can be used as a simulator of classical dynamics of realistic atoms as well as general model particles. It supports parallel computing through spatial divisions. Its codes are designed so that their modification and extension are easy.

5.7.2 LICENSE

GPLv2

5.7.3 OFFICIAL PAGE

https://lammps.sandia.gov

5.7.4 MateriApps URL

https://ma.issp.u-tokyo.ac.jp/en/app/613

5.8 mVMC

5.8.1 SUMMARY

A low-energy solver for a wide ranger of quantum lattice models (multi-orbital Hubbard model, Heisenberg model, Kondo-lattice model) by using variational Monte Carlo method. User can obtain high-accuracy wave functions for ground states of above models. Users flexibly choose the correlation factors in wavefunctions such as Gutzwiller, Jastrow, and doublon-holon binding factors and optimize more the ten thousand variational parameters. It is also possible to obtain the low-energy excited states by specifying the quantum number using the quantum number projection.

5.8.2 LICENSE

GNU General Public License version 3

5.8.3 OFFICIAL PAGE

http://www.pasums.issp.u-tokyo.ac.jp/mvmc/en/

5.8.4 MateriApps URL

https://ma.issp.u-tokyo.ac.jp/en/app/518

5.9 OpenMX

5.9.1 SUMMARY

OpenMX is a first-principles software based on the pseudo-atomic localized basis functions. It calculates electronic structure rapidly for a wide range of materials including crystals, interfaces, liquids, etc. It speedily provides molecular dynamics simulation and structural optimization of large-scale systems and also implements a hybrid parallelism. It is able to deal with non-collinear magnetism and non-equilibrium Green's function calculations for electrical conductions.

5.9.2 LICENSE

GNU-GPL

5.9.3 OFFICIAL PAGE

http://www.openmx-square.org/

5.9.4 MateriApps URL

https://ma.issp.u-tokyo.ac.jp/en/app/649

5.10 RESPACK

5.10.1 SUMMARY

RESPACK is a first-principles calculation software for evaluating the interaction parameters of materials. It is able to calculate the maximally localized Wannier functions, the RPA response functions, and frequency-dependent electronic interaction parameters. RESPACK receives its input data from a band calculation using norm-conserving pseudopotentials with plane-wave basis sets. Utilities which convert a result of xTAPP or Quantum ESPRESSO to an input for RESPACK are prepared. The software has been used successfully for a wide range of materials such as metals, semiconductors, transition-metal compounds, and organic compounds. It supports OpenMP / MPI parallelization.

5.10.2 LICENSE

GNU GPL v3

5.10.3 OFFICIAL PAGE

https://sites.google.com/view/kazuma7k6r

5.10.4 MateriApps URL

https://ma.issp.u-tokyo.ac.jp/en/app/772

5.11 TeNeS

5.11.1 SUMMARY

A solver program for two dimensional quantum lattice model based on a projected entangled pair state wavefunction and the corner transfer matrix renormalization group method.

5.11.2 LICENSE

GNU GPLv3

5.11.3 OFFICIAL PAGE

https://github.com/issp-center-dev/TeNeS

5.11.4 MateriApps URL

https://ma.issp.u-tokyo.ac.jp/en/app/2291

CHAPTER

SIX

CONTACT

- Bug reports

  Please report all problems and bugs on the GitHub Issues page . To resolve bugs early, follow these guidelines when reporting:

  - Please specify the version of Materiapps Installer you are using.
  - If there are problems for installation, please inform us about your operating system and the compiler.
  - If a problem occurs during execution, enter the input file used for execution and its output.

- Contact

  If you have any questions about topics related to your research that are difficult to consult at Issues on GitHub, please contact the following contact information.

  E-mail: mainstaller-dev_at_issp.u-tokyo.ac.jp (replace _at_ by @)

CHAPTER

SEVEN

ACKNOWLEDGEMENT

The development of this software has been supported by following projects and computer resources.

- "New Materials and Energy Creation" of the HPCI Strategic Program for Innovative Research (SPIRE), MEXT, Japan
- Social and Scientific Priority Issue "Creation of new functional devices and high-performance materials to support next-generation industries" to be Tackled by Using Post-K Computer, MEXT, JAPAN

We would also like to express our thanks for the support of the "Project for advancement of software usability in materials science" of The Institute for Solid State Physics, The University of Tokyo, for the development of MateriApps Installer.