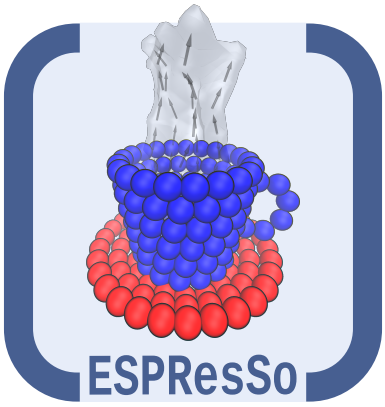

Setting up ESPResSo *

G. Rempfer and F. Weik

July 27, 2017

1 Introduction

ESPResSo is distributed as source code, because this allows the user to disable features that are mutually exclusive or would otherwise decrease the performance unnecessarily. Users typically build a custom ESPResSo binary for a specific simulation, containing only the minimal set of features required for this application. This tutorial will guide you through this whole process from obtaining the source code of the development and the release version, to configuring and building binaries, to compiling the documentation.

2 Getting the Source Code

The procedure of building ESPResSo the development and the release versions differs only slightly.

2.1 Development Version

For the development version, first clone the ESPResSo reposity from github by running

*For ESPResSo 3.4-dev-4615-g5db05d407

```
git clone https://github.com/espressomd/espresso.git
```

in a terminal. This creates a local copy of the current ESPResSo repository in the folder `espresso`. Enter the newly create directory using

```
cd espresso
```

and run

```
./bootstrap.sh
```

to configure the build system. This step is only necessary in the development version.

2.2 Release Version

The ESPResSo releases can be downloaded from the ESPResSo website at `http://espressomd.org/wordpress/download/`. After downloading the latest version unpack it by running

```
tar xfv espresso-X.Y.Z.tar.gz
```

where X Y an Z have to be replaced with the actual version numbers.

2.3 Compiling ESPResSo

From here on the procedure is the same for both versions. Change into the respective newly created ESPResSo directory and create another subfolder to hold your build and enter it by issuing

```
mkdir build
cd build
```

You can maintain several such build folders with different feature sets using the same ESPResSo source folder. They don't have to reside within the source directory. From within the build folder, run

```
../configure
```

which create the makefiles. ESPResSo requires a number of external libraries, such as `libtcl` and others, depending on the feature set you plan to use. Most, if not all of these libraries will be available through your Linux distribution's repositories. If you manually

install dependencies in a non-standard location, you have to specify this location during the `configure` stage of the build process. If, for example, you plan to use GPU features and your cuda toolkit is not located under `/usr/local/cuda`, you will have to specify its location using

```
../configure --with-cuda=/PATH/TO/CUDA
```

A complete list of options can be obtained through

```
../configure --help
```

Finally compile your version of ESPResSo with

```
make -j 8
```

This produces an executable named `Espresso` in the build directory which can be run by

```
./Espresso
```

2.4 User's guide

You can build the ESPResSo manual called *ESPResSo User's Guide* by running

```
make ug
```

in the build directory. The User's Guide will be located under `build/doc/ug/ug.pdf`. It is an (almost) complete documentation covering all the features and commands available in ESPResSo. In addition, the User's Guide features some theoretical background information on some of the methods as well as references to the relevant literature.

2.5 Configuring features

After successfully running the configure script, your build folder will contain a file `myconfig-sample.hpp`. Create a copy of this file called `myconfig.hpp` by executing

```
cp myconfig-sample.hpp myconfig.hpp
```

Then uncomment the respective lines in `myconfig.hpp` (by removing the leading slashes) to enable features your simulation requires. The User's Guide contains information

about required features for every ESPResSo command. When you activate or deactivate features, you have to `rebuild` the binaries by running

```
make -j 8
```

in the build directory.