Practical Guide to DL_FFLUX

Compiling DL_FFLUX

DL_FFLUX can be found in the `popelier-group/FFLUX_MPI` GitHub repository. DL_FFLUX is compiled using an Intel MPI compiler (tested with 18.0.3), on both ffluxlab and csf3, this compiler can be made available by loading the `mpi/intel/18.0.3` module. In the `Source` directory a `Makefile` can be found, DL_FFLUX is compiled using the `hpc` rule, note the `Makefile` default is to use `mpiifort`, if this does not exist on your system, modify the `FC` variable on line 23 of the `Makefile` to a suitable compiler (usually `mpif90` or `mpifort`).

```
git clone https://github.com/popelier-group/FFLUX_MPI.git
cd FFLUX_MPI/Source
make hpc
```

The executable will then be found at `FFLUX_MPI/execute/DLPOLY.Z` (relative to the `Source` directory, this becomes `../execute/DLPOLY.Z`).

Running DL_FFLUX

In order to run DL_FFLUX, 3 files and the GPR models are required:

- `CONFIG`
  - This file contains the geometry information of the system that will be ran
- `CONTROL`
  - This file contains the control parameters used to run the simulation
- `FIELD`
  - This file contains the atom type information needed to run the simulation
- `model_krig`
  - All model files are placed in this directory

`CONFIG`

The `CONFIG` file is responsible for defining the input geometry to the simulation as well as the unit cell size and the model that should be used by each atom in the simulation. A brief description of the `CONFIG` file format can be found here, (the full specification can be found in section 4.1.2 of the DLPOLY User Manual) note that DL_FFLUX is a slightly modified version of the standard file format with the addition of the model identification for each atom. The following is an example `CONFIG` input for a formamide monomer simulation:

```
Frame :          1
        0        1
  50.0 0.0 0.0
  0.0   50.0   0.0
  0.0   0.0   50.0
C   1   FORMAMIDE_C1
-8.18738677              5.15550517              0.58626642
```

```
O  2  FORMAMIDE_O2
-8.5991229              5.06276356              1.72604339
N  3  FORMAMIDE_N3
-6.89430273             5.51926523              0.29088674
H  4  FORMAMIDE_H4
-8.83052459             4.9455594               -0.36975386
H  5  FORMAMIDE_H5
-6.1406049              5.50427809              1.00295123
H  6  FORMAMIDE_H6
-6.59977684             5.39100455              -0.66507858
```

The first line is a title line, the second line includes the `levcfg` and `imcon` variables. `levcfg` defines whether the file includes coordinates, velocities and forces (`levcfg=0` specifies coordinates only) and the `imcon` value specifies what periodic boundary is being used (`imcon=1` specifies a cubic boundary condition). The next three lines are the unit cell dimensions for the cubic unit cell (in Angstroms). Then each atom has a header line stating the atom type, the atom index and the model to associate to said atom followed by the cartesian coordinates on the next line (note that if `levcfg>0` then subsequent velocity and force lines may be needed). For the first atom (`C1`), DL_FFLUX will predict all atomic properties using the files `model_krig/FORMAMIDE_{property}_C1.model`, where `{property}` will be replaced by the atomic property to be predicted (e.g. `iqa` or `q00` etc.).

CONTROL

The `CONTROL` file contains the control parameters used to run a DL_FFLUX simulation. The full list of control parameters can be found in section 4.1.1 in the DLPOLY User Manual. Note that DL_FFLUX has added several control parameters specific to FFLUX, these may be found in the DL_FFLUX user manual.

```
Title: FORMAMIDE

ensemble nvt hoover 0.04
temperature 1

timestep 0.001
steps 1000
scale 100

cutoff  8.0
rvdw    8.0
vdw direct
vdw shift
fflux cluster L1

dump  1000
traj 0 1 0
print every 1
stats every 1
fflux print 0 1
job time 10000000
close time 20000
finish
```

The above `CONTROL` file will run a 1 K simulation (typical for geometry optimisations) for 1000 timesteps using the Nosé-Hoover thermostat.

`FIELD`

The `FIELD` file specifies the parameters for each atom used in the simulation. The full `FIELD` file specification can be found in section 4.1.3 in the DLPOLY User Manual. The `FIELD` file format has not been modified by DL_FFLUX.

```
DL_FIELD v3.00
Units kJ/mol
Molecular types 1
FORMAMIDE
nummols 1
atoms 6
C               12.0106000    0.10   1   0
O               15.9994000    0.10   1   0
N               14.0068550    0.10   1   0
H               1.0079750     0.10  1   0
H               1.0079750     0.10   1   0
H               1.0079750     0.10   1   0
BONDS 5
harm 1 2 0.0 0.0
harm 1 3 0.0 0.0
harm 1 4 0.0 0.0
harm 3 5 0.0 0.0
harm 3 6 0.0 0.0
ANGLES 6
harm 1 3 5 0.0 0.0
harm 1 3 6 0.0 0.0
harm 2 1 3 0.0 0.0
harm 2 1 4 0.0 0.0
harm 3 1 4 0.0 0.0
harm 5 3 6 0.0 0.0
DIHEDRALS 4
harm 2 1 3 5 0.0 0.0
harm 2 1 3 6 0.0 0.0
harm 4 1 3 5 0.0 0.0
harm 4 1 3 6 0.0 0.0
finish
close
```

The above `FIELD` file describes a formamide monomer. Each atom requires the atom type, atomic weight, charge, number of repeats and whether the atom is frozen. Note that the charge is predicted so isn't needed but must be a nonzero value otherwise DL_FFLUX will break. After specifying every atom, the bonds, angles and dihedrals must be specified with the parameters for the harmonic potential. It is also the case that the harmonic potential from DLPOLY is not used by DL_FFLUX so although one has to list each bond, angle and dihedral, the harmonic potential parameters may be set to 0. ICHOR provides a `write_field` function (as well as functions to write the other DL_FFLUX input files) to help with the writing of such a tedious file.

`model_krig`

The `model_krig` directory contains all of the `.model` files that will be used for the simulation. For monomeric simulations (no intermolecular interactions required), only `iqa` models are required, otherwise `iqa` and multipole moment (`q00` through `q44s`) models are required.

Running the DL_FFLUX executable

With the `CONFIG`, `CONTROL`, `FIELD` and `model_krig` all in a single directory, the DL_FFLUX executable should be placed in the same directory and ran with the following:

```
mpirun -n <NUM_THREADS> ./DLPOLY.z
```

With `<NUM_THREADS>` replaced with the number of threads to run DL_FFLUX with (it is important to have the MPI module loaded on ffluxlab or csf).

After running, many output files will be generated, the two most important are the `OUTPUT` and `HISTORY` files. The former containing outputs of the whole system for each timestep (such as total energy etc.) and the latter containing the trajectory of the simulation.