Documentation for DPLM Simulation

Ling-Han Zhang[*]

Department of Applied Physics, Hong Kong Polytechnic University

(Dated: July 31, 2016)

CONTENTS

[*] Email: ns.zhang@connect.polyu.hk; linghanz@cmu.edu; Visit: https://github.com/zhangns

I. OVERVIEW

This documentation describes the basic usage of the simulation program for the Distinguishable-Particle Lattice Model (DPLM) in 2D.

II. PREPARATION

A. Prerequisites

For full features, one needs

OS: Windows (confirmed), Linux (confirmed), or others.

Simulation: recent versions of Visual Studio (confirmed) or GCC (confirmed), or other C++11 conforming compilers.

Real-time visualization: libpng, and FLTK with OpenGL support. On Windows the packages are installed automatically by Visual Studio. On Linux, you may install them with APT: `sudo apt-get install libpng-dev libfltk1.3-dev`

Data processing and plotting: MATLAB.

B. Building

The executable programs, `glass` and `batchglass`, need to be built from C++ source (found in `./src/`). On Windows, open and build the solution (`Glass.sln`) with Visual Studio. On Linux, simply locate `makefile`, and `make`

III. SIMULATION

A. Parameters

Possible input parameters to a simulation run are given in Table I. Parameters are processed in `Config.cpp`. You can choose to run a simulation in either of two output modes, macromode (default option) and micromode. Columns "MA" and "MI" in the table tell if a parameter is relevant under the two modes respectively. Under macromode, the program dumps particle coordinates to file `traj` periodically in every time interval `dt`. This is suitable

TABLE I. Input Parameters

Name	Type	Default	MA	MI	Description
`L`	`int`	`100`	✓	✓	Lattice side length.
`N`	`int`	`9900`	✓	✓	Number of particles. Overrides.
`phi`	`double`		✓	✓	Particle concentration $\phi = N/L^2$. Overrides.
`phiv`	`double`		✓	✓	Void concentration $\phi_v = 1 - \phi$. Overrides.
`boltz`	`int`	`-1`	✓		Number of frozen particles in Boltzmann test. -1 = turn off test.
`seed`	`int`	`42`	✓	✓	Random seed for the RAN2 PRNG.
`resume`	`int`	`0`	✓		Whether try to resume simulation.
`ap`	`int`	`1`	✓	✓	Equilibration mode. 0 = none. 1 = instant. 2 = instant + preheat.
`out`	`int`	`0`	✓	✓	Output mode. 0 = macromode. 1 = micromode.
`ndt`	`int`	`10`	✓		Number of `dt` to simulate. 0 = infinite.
`nmcs`	`int`	`10000000`		✓	Number of Monte Carlo steps (MCS) to simulate.
`calmsd`	`int`	`0`		✓	Whether compute mean square displacement $g(t)$.
`calsisf`	`int`	`0`		✓	Whether compute self-intermediate scattering function $F_s(k,t)$.
`calfpcf`	`int`	`0`		✓	Whether compute four-point correlation function $S_4(k,q,t)$.
`calpret`	`int`	`0`		✓	Whether compute returning and non-returning hopping probabilities P_{ret} and P_2.
`Vmin`	`double`	`-0.5`	✓	✓	Minimum bond energy ($V_{ijkl} \in [V_{min}, V_{min}+1]$).
`T`	`double`	`0.2`	✓	✓	Temperature.
`dt`	`double`	`1e6`	✓	✓	Time interval. In micromode, -1 = let `dt` $\approx$ 1 MCS.
`frinc`	`double`	`1.4`		✓	Fractional increment of t for measurements.
`k`	`ints`	`25`		✓	Comma-separated reduced wavenumbers for $F_s(k,t)$ where $k = 2\pi\mathtt{k}/L$.
`q`	`ints`			✓	Comma-separated reduced wavenumbers for $S_4(k,q,t)$. Do not include $\mathtt{q} = 0$.

for measurements over long time scales (typically `dt` $> 10^6$ MCS). Note that measurements over time scales shorter than `dt` cannot be done based on the particle trajectories dumped. Under micromode, the program performs `nmcs` MCS and directly computes quantities as requested, e.g. $g(t = \mathtt{dt})$, $g(t = \mathtt{frinc} \times \mathtt{dt})$, $g(t = \mathtt{frinc}^2 \times \mathtt{dt})$, etc.. This is suitable for measurements over short time scales (typically 1 MCS $\sim 10^8$ MCS). The two output modes are implemented in `MacroMode.cpp` and `MicroMode.cpp`. All output files are plain-text and use the most straightforward format.

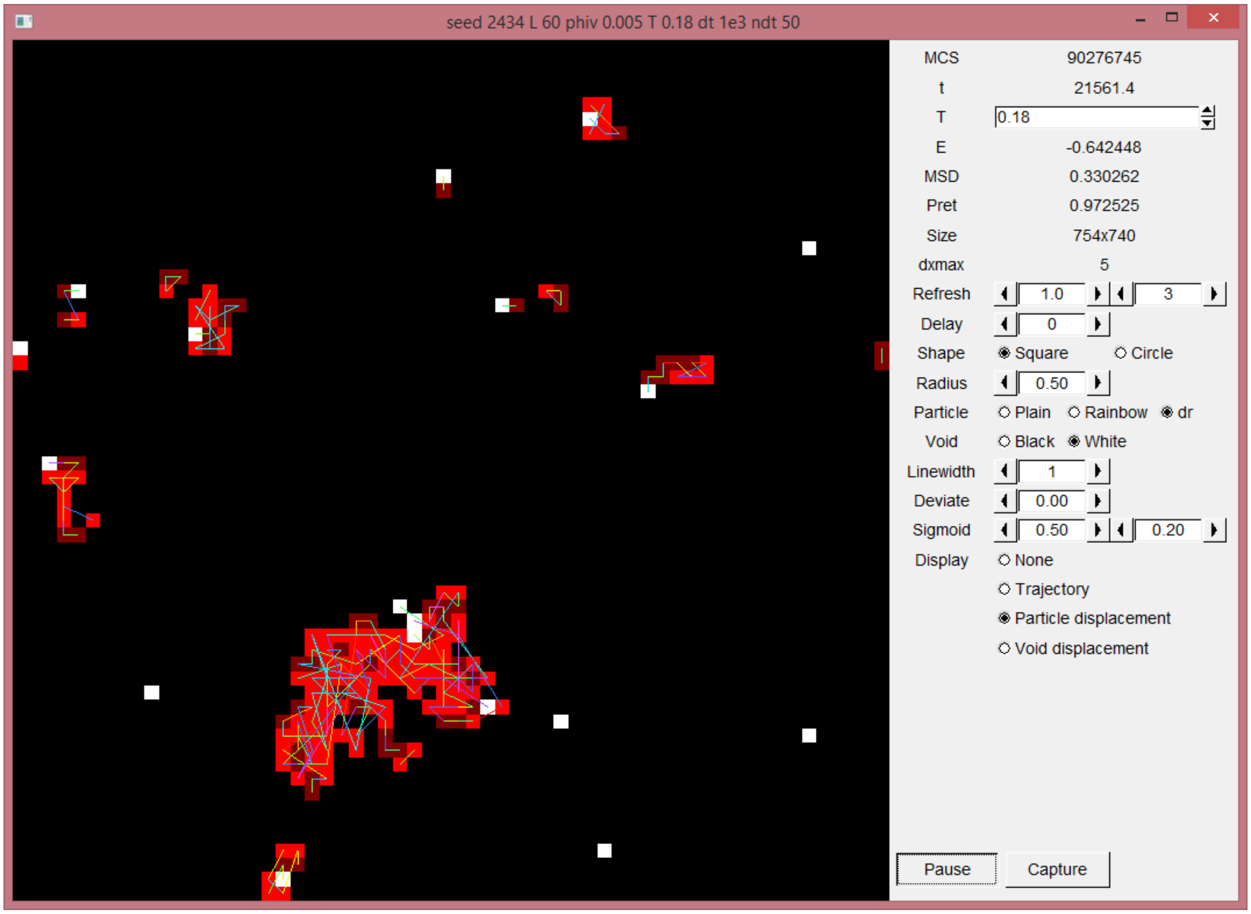
B. Single run with visualization

The executable `glass` performs a single run and provides real-time visualization within a GUI. The input parameters are supplied with command-line arguments, with the format

```
<path to>/glass [<name> <value>]...
```

Ouput data files will be generated in the current working directory. Example:

```
glass seed 2434 L 60 phiv 0.005 T 0.18 dt 1e3 ndt 50
```



seed 2434 L 60 phiv 0.005 T 0.18 dt 1e3 ndt 50

```
MCS         90276745
t           21561.4
T           0.18
E           -0.642448
MSD         0.330262
Pret        0.972525
Size        754x740
dxmax       5
Refresh     1.0   3
Delay       0
Shape       ● Square     ○ Circle
Radius      0.50
Particle    ○ Plain  ○ Rainbow  ● dr
Void        ○ Black  ● White
Linewidth   1
Deviate     0.00
Sigmoid     0.50   0.20
Display     ○ None
            ○ Trajectory
            ● Particle displacement
            ○ Void displacement
```

C. Concurrent runs in batch

The executable `batchglass` can perform multiple runs, with one thread dedicated for each. This is suitable for multicore or multiprocessor systems, and for generating a lot of data with least human intervention. The input parameters of all runs are supplied in a single batch file, in which each run is specified by one line with the format

```
<rundir> [<name> <value>]...
```

where `<rundir>` is the relative path where data files for that run will be saved.

Example batch file for macromode:

```
T0220/v014/r01 seed 65596257 L 100 N 9860 g0 -0.5 T 0.220 ap 2 resume 1 dt 7.7e+00 ndt 10000
T0220/v018/r01 seed 57851882 L 100 N 9820 g0 -0.5 T 0.220 ap 2 resume 1 dt 6.0e+00 ndt 10000
T0220/v023/r01 seed 89610750 L 100 N 9770 g0 -0.5 T 0.220 ap 2 resume 1 dt 4.8e+00 ndt 10000
T0220/v030/r01 seed 41735407 L 100 N 9700 g0 -0.5 T 0.220 ap 2 resume 1 dt 3.6e+00 ndt 10000
T0220/v039/r01 seed 08954296 L 100 N 9610 g0 -0.5 T 0.220 ap 2 resume 1 dt 2.9e+00 ndt 10000
T0220/v050/r01 seed 16960722 L 100 N 9500 g0 -0.5 T 0.220 ap 2 resume 1 dt 2.2e+00 ndt 10000
```

Example batch file for micromode:

```
T0180/v005/r04 seed 0936 L 100 N 9950 g0 -0.5 T 0.18 out 1 nmcs 50000000 dt 8.4e-5 calmsd 1 calsisf 1 k 15 calfpcf 1 q 1,2
T0180/v006/r04 seed 9331 L 100 N 9940 g0 -0.5 T 0.18 out 1 nmcs 50000000 dt 6.9e-5 calmsd 1 calsisf 1 k 15 calfpcf 1 q 1,2
T0180/v008/r04 seed 0572 L 100 N 9920 g0 -0.5 T 0.18 out 1 nmcs 50000000 dt 4.9e-5 calmsd 1 calsisf 1 k 15 calfpcf 1 q 1,2
T0180/v011/r04 seed 9398 L 100 N 9890 g0 -0.5 T 0.18 out 1 nmcs 50000000 dt 3.7e-5 calmsd 1 calsisf 1 k 15 calfpcf 1 q 1,2
T0180/v014/r04 seed 2671 L 100 N 9860 g0 -0.5 T 0.18 out 1 nmcs 50000000 dt 2.9e-5 calmsd 1 calsisf 1 k 15 calfpcf 1 q 1,2
```

Finally, run the simulations specified in the batch file with

```
<path to>/batchglass <batchfile> [<maxNumThreads>]
```

IV. DATA PROCESSING

The data generated by the simulation program can be readily processed and analyzed with the help of MATLAB, or other numerical software. I would not go into details here. The MATLAB scripts I used are provided in `./matlab/`

V. FURTHER RESEARCH & DEVELOPMENT

Possible further work includes but is not limited to

1. Analytical calculations on the model

2. Other measurements, e.g. van Hove function $G_s(r,t)$, linear response $\chi_T(\omega)$

3. Extending to cubic lattice in 3D

4. Using a different bond energy distribution $g(V_{ijkl})$ to change fragility, e.g. Gaussian, or bimodal

5. Introducing other types of interactions to the Hamiltonian, e.g. between next-nearest neighbors (NNN), and possibly applying the free-surface condition

6. Introducing other types of events to the dynamics, e.g. coherent hoppings of multiple voids, and particle insertion/deletion (μVT ensemble)

7. Using different system sizes to investigate finite-size effects

8. Modify the quenched disorder in the interactions, e.g. depending only on particle indices not the lattice sites