

MMM Hub Software Spotlight: DL_Poly Molecular Simulation Package

Alin M Elena

stfc daresbury laboratory

May, 5 2021

DL_POLY

Helpers

demo

DL_POLY

Daresbury Laboratory



Disclaimer

- ▶ the talk is about DL_POLY not about molecular dynamics
- ▶ for MD
 - ▶ Computer Simulation of Liquids, **Michael P. Allen and Dominic J. Tildesley**, 2nd ed, OUP (2017)
 - ▶ Understanding Molecular Simulation: From Algorithms to Applications, **Daan Frenkel & Berend Smit**, Academic Press, 2nd ed, (2001)
 - ▶ Statistical Mechanics: Theory and Molecular Simulation, **Mark Tuckerman**, OUP (2010)
 - ▶ CCP5 Summer School, (summer2021.ccp5.ac.uk)

What is DL_POLY_4?

- ▶ classical classical molecular dynamics code, started by Bill Smith&Co under CCP5 umbrella.
- ▶ started in 1992, initially memory replicated paradigm, current version is memory distributed, all in MPI.
- ▶ access to modern classical molecular dynamics via PLUMED2.
- ▶ written in Fortran, rewritten few times, currently modern Fortran
- ▶ <https://gitlab.com/ccp5/dl-poly>
- ▶ LGPL 3.0

Who?



Engineering and
Physical Sciences
Research Council



Science and
Technology
Facilities Council



NATURAL
ENVIRONMENT
RESEARCH COUNCIL

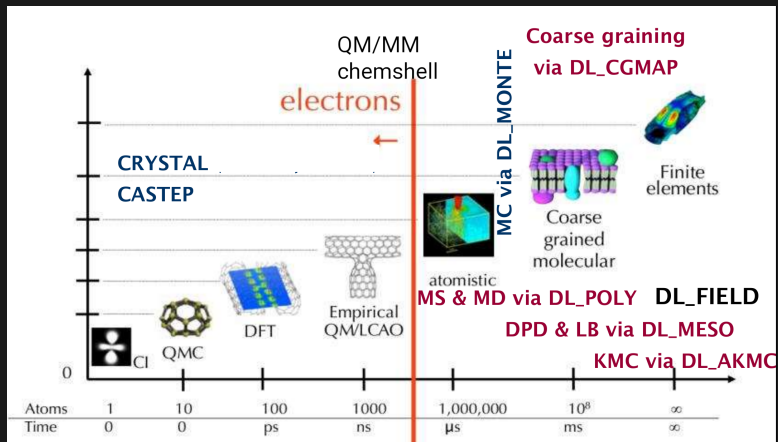


Who ?

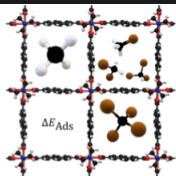
Bill Smith, Tim Forester, Maurice Leslie, Ilian Todorov, Ian Bush, Michael Seaton, Andrey Brukno, Jim Madge, Ivan Scivetti, Jacob Wilkins, Alex Buccheri, Aaron Diver, Oliver Dicks, S L Daraszewicz, G Khara, S Murphy, Pierre Cazade, David Quigley, Alin Elena, Christos Kartsaklis, Ruslan Davidchak, Henri Boateng

groups who have their own version of modified DL_POLY but not released to public.

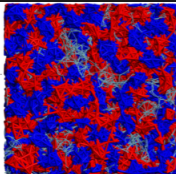
Bigger picture



Where



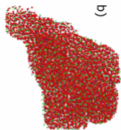
MOFs



Ionic liquids

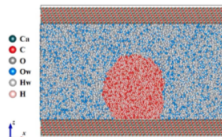


(c)

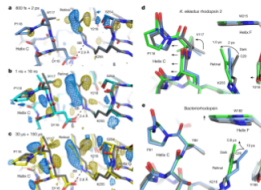


(d)

Radiation damage helium diffusion in zircon



oil/brine/calcite



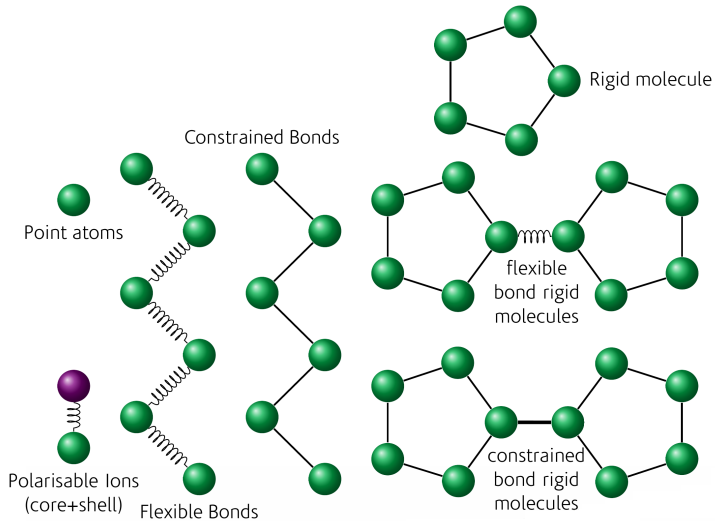
light-driven sodium pump (via chemshell)

- ▶ <https://doi.org/10.1021/acs.jpcc.0c02966>
- ▶ <https://doi.org/10.1016/j.fuel.2020.117615>
- ▶ <https://arxiv.org/abs/2104.12599>
- ▶ <https://doi.org/10.1016/j.molliq.2019.112402>
- ▶ <https://www.nature.com/articles/s41586-020-2307-8>

where 2

- ▶ Probing the dynamics and structure of confined benzene in MCM-41 based catalysts
- ▶ Elucidating esterification reaction during deposition of cutin monomers from classical molecular dynamics simulations
- ▶ Crossover of dynamical instability and chaos in the supercritical state
- ▶ Computational Assessment of Water Desalination Performance of Multi-Walled Carbon Nanotubes
- ▶ Understanding and controlling the glass transition of HTPB oligomers
- ▶ Controlling CaCO₃ Particle Size with {Ca²⁺}:{CO₃²⁻} Ratios in Aqueous Environments

Atoms and molecules



- ▶ *Intra-molecular interactions*: chemical bonds, bond angles, dihedral angles, improper dihedral angles, inversions. tethers, core shells units, holonomic constraint and PMF units, rigid body units. These are defined by site.
- ▶ *Inter-molecular interactions*: van der Waals, metal (2B/E/EAM, Gupta, Finnis-Sinclair, Sutton-Chen), Tersoff, three-body, four-body. can be analytical or tabulated. These are always spherically symmetric! Defined by species.

- ▶ *Electrostatics*: SPM Ewald (3D FFTs), Force-Shifted Coulomb, Reaction Field, Fennell damped FSC+RF, Distance dependent dielectric constant, Fuchs correction for non charge neutral MD cells.
- ▶ *Ion polarisation* via Dynamic (Adiabatic) or Relaxed shell model.
- ▶ *External fields*: Electric, Magnetic, Gravitational, Oscillating Continuous Shear, Containing Sphere, Repulsive Wall.

$$\begin{aligned}
V(r_1, \dots, r_N) = & \sum_{i,j=1}^N V_{pair}(|r_i - r_j|) + \sum_{i,j=1}^N \frac{q_i q_j}{4\pi\epsilon_0\epsilon_r |r_i - r_j|} + \\
& + \sum_{i,j,k=1}^N V_{3b}(r_i, r_j, r_k) + \sum_{i,j,k,l=1}^N V_{4b}(r_i, r_j, r_k, r_l) \\
& + \epsilon_{metal} \left(\sum_{i,j=1}^N V_m(|r_i - r_j|) + \sum_{i=1}^N F \left(\sum_{i,j=1}^N \rho_{ij}(|r_i - r_j|) \right) \right) + \\
& + \sum_{i_{ab}=1}^{N_{bonds}} V_{bond}(i_{ab}, r_a, r_b) + \sum_{i_{abc}=1}^{N_{angles}} V_{angle}(i_{abc}, r_a, r_b, r_c) + \\
& + \sum_{i_{abcd}=1}^{N_{dihedrals}} V_{dihedral}(i_{abcd}, r_a, r_b, r_c, r_d) + \sum_{i_{abcd}=1}^{N_{inverse}} V_{inverse}(i_{abcd}, r_a, r_b, r_c, r_d) + \\
& + \sum_{i_a}^{N_{tethers}} V_{tether}(i_a, r_a^{\tau=t}, r_a^{\tau=0}) + \sum_{i_{cs}=1}^{N_{shells}} V_{core-shell}(i_{cs}, r_c, r_s) + \sum_{i=1}^N \varphi_{ext}(r_i)
\end{aligned}$$

Periodic boundary conditions

- ▶ None (e.g. isolated macromolecules)
- ▶ Cubic periodic boundaries
- ▶ Orthorhombic periodic boundaries
- ▶ Parallelepiped (triclinic) periodic boundaries
- ▶ in dlpoly classic you can find
 - ▶ Truncated octahedral periodic boundaries
 - ▶ Rhombic dodecahedral periodic boundaries

integrations of EOM

- ▶ Velocity Verlet (fixed timestep, variable timestep)
- ▶ Thermostats
 - ▶ NVE
 - ▶ NVT (E kin) Evans, Andersen, Langevin, Berendsen, Nosé-Hoover, Gentle stochastic thermostat
 - ▶ NPT Langevin, Berendsen, Nosé-Hoover, Martyna-Tuckerman-Klein
 - ▶ NoT/NPnAT/NPnyT Langevin, Berendsen, Nosé-Hoover, Martyna-Tuckerman-Klein
 - ▶ NVT dpdS1 dpdS2 Sharlow 1st or 2nd order splitting
 - ▶ boundary thermostat (radiation damage) similar in spirit with Ciccotti and Lev Kantorovich's ones
 - ▶ two temperature thermostat
- ▶ constraints: parallelised SHAKE/RATTLE
- ▶ rigid bodies: No_squish

- ▶ domain decomposition is used for MPI parallelisation
- ▶ neighbour lists are computed using linked cells with subcelling.

calculations

- ▶ Statistics of common thermodynamic properties: temperature, pressure, energy, enthalpy, volume, virials
- ▶ RDF, VAF, MSD, Z density profiles, various pdf (angles, bonds)
- ▶ generate trajectories
- ▶ defects analysis
- ▶ replay trajectories
- ▶ thermal conductivity
- ▶ short range corrections (or exact calculations)
- ▶ optional Empirical Valence Bond

- ▶ mpi memory distributed (almost all codes are these days)
- ▶ forcefield agnostic
- ▶ all in one, well almost
- ▶ an entire zoology of statistical ensembles...
- ▶ rigid bodies, holonomic constraints
- ▶ special radiation damage analysis tools
- ▶ extra potentials via openKIM
- ▶ modern modelling techniques via PLUMED2
- ▶ geometry minimisation
- ▶ parallel IO via MPI

Numbers

Language	Fortran
files	<i>111</i>
blank	<i>35511</i>
comment	<i>21925</i>
code	<i>100572</i>
tests	<i>170</i>

August 2020

standard 2003/2008 (current 2018 but not well supported)

contributors

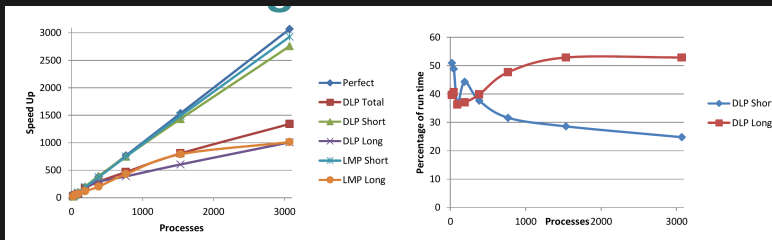
commits	author
747 452	Alin M Elena (STFC)
480 381	Jim Madge (STFC, ATI)
135 117	Jacob Wilkins (Oxford, now STFC)
66 66	Aaron Diver (QMUL)
94 56	Ivan Scivetti (STFC)
34 33	Yaser Afshar (U of Minnesota)
33 28	Michael Seaton (STFC)
27 24	Oliver A Dicks (QMUL)
29 16	Ilian Todorov (STFC)
21 15	Alexander Buccheri (Bristol, now FHI)
15 13	Aidan Chalk (STFC, HC)

last few years

what refactoring did for us

- ▶ a new build system around, cmake
- ▶ continuous integration and the necessary tests
- ▶ devise a new control format, much friendlier for automatic reading
- ▶ remove old features (leap frog for example as integration method)
- ▶ deprecate and mark for removal netcdf support, replace with hdf5 or adios
- ▶ discover new bugs (eg. multipoles are broken)
- ▶ add new timing system
- ▶ speedup the code a little bit

Scaling



- ▶ 1728000 atoms (NaCl system Van der Waals and electrostatics only)
- ▶ Martyn F. Guest, Alin M. Elena & Aidan B. G. Chalk Molecular Simulation (2019), <https://doi.org/10.1080/08927022.2019.1603380>

- ▶ report bugs on
<https://gitlab.com/ccp5/dl-poly>
- ▶ Mat Sci forum:
<https://matsci.org/c/dlpoly/28>
- ▶ matrix room: see link on gitlab
- ▶ you can still use the mailing list.
dlpoly@jiscmail.ac.uk

Helpers

- ▶ python companion:
<https://gitlab.com/drFaustroll/dlpoly-py>
- ▶ devs: Jacob Wilkins(Oxford period), Alin Elena
- ▶ license: BSD 3.0

features

- ▶ reads dlpoly input files, provides python objects
- ▶ converts all control to new control style
- ▶ reads main output files: REVCN, STATIS, RDF...
- ▶ does not read trajectory, for that we use ASE
- ▶ writes input files for dlpoly
- ▶ runs the code via files.
- ▶ config builder?

software engineering

- ▶ modern python (no version 2 support)
- ▶ small size 3000 lines or so
- ▶ unit tests, using tox 50% coverage
- ▶ documentation, minimal
- ▶ style enforcement, flake8
- ▶ pylint to be added?

install

```
pip install dlpoly-py
```

https://www.ccp5.ac.uk/DL_FIELD

- ▶ Force field model convertor: Conversion of a user's atomic configuration into input files (FIELD, CONFIG) for DL_POLY molecular dynamics software based on the user-selectable force field (FF)
- ▶ Available FF schemes: CHARMM, AMBER (inc. Glycam), OPLS2005, CL&P, PCFF, CVFF, DREIDING and G54A7, CHARMM19 (united atom). Inorganic force fields for ionic solids and minerals including CLAYFF, zeolites. These schemes are all expressed in a consistent file format within DL_FIELD.

demo

installation

```
git clone -b 5.0.0 https://gitlab.com/ccp5/dl-poly.  
dl-poly-5.0.0
```

```
cmake -S dl-poly-5.0.0 -Bbuild-dlpoly \  
-DCMAKE_BUILD_TYPE=Release
```

```
cmake --build build-dlpoly
```

```
mpirun -n 8 build-dlpoly/bin/DLPOLY.Z -c Ar.control
```

input files

- ▶ input (minimal): config file (CONFIG), forcefield file (FIELD), control file (CONTROL)
- ▶ output: statistics file (STATIS), output log (OUTPUT)
- ▶ restart files: positions, velocities, forces (REVCN), statistics (REVIVE)

other files, tabulated potentials TABLE, RDF (RDFDAT), trajectory (HISTORY)

full listing in the manual.