THE DL_FIELD USER MANUAL

VERSION 4.13, JUNE 2026

Chin W Yong

Classical Materials & Molecular Modelling Group,
Scientific Computing Department
UK Research and Innovation,
Science and Technology Facility Council,
Daresbury Laboratory,
Warrington WA4 4AD,
Cheshire, UK

The DL_FIELD manual assumes readers possess at least a basic knowledge in molecular simulation and potential force fields. It mainly describes the functionality of DL_FIELD. It does not describe how and when to use the features offer by DL_FIELD.

DL_FIELD is a computer program package written in C that primarily serves as a support application software tool for DL_POLY molecular dynamics simulation package. DL_FIELD is developed at Daresbury Laboratory by C. W. Yong. The software was developed under the auspices of the Computational Science Centre for Research Communities (CoSeC) in support of several EPSRC-funded communities including Collaborative Computational Project for the Computer Simulation of Condensed Phases (CCP5) and HEC-MCC (High-End Computing for Materials Chemistry Consortium).

If you use DL_FIELD in your work, please include the following reference in your publication:

C W Yong, 'Descriptions and Implementations of DL_F Notation: A Natural Chemical Expression System of Atom Types for Molecular Simulations', J. Chem. Inf. Model. 56, 1405–1409 (2016)

Disclaimer

While extensive tests have been made to ensure smooth working of DL_FIELD and the accuracy of the supplied data parameters, neither the UKRI, STFC, EPSRC, CCP5, HEC-MCC nor the author of the DL_FIELD package or its derivatives guarantee that the software package is free from error. We disclaim any responsibility for any failure, inaccuracy, harm and damage to your projects, theoretical or experimental works as a result of using DL_FIELD.

Table of Contents

1 Introduction

DL_FIELD is a powerful and flexible application tool to setup force field (FF) models for DL_POLY, LAMMPS and GROMACS molecular dynamics (MD) simulation package and to facilitate the use of a wide range of features included in the MD program suite.

DL_FIELD functions:

(1) Force field (FF) model convertor: DL_FIELD converts user's atom models, particularly those of large complex biomolecular systems, into file formats that are recognisable and ready to run in DL_POLY_2.19, DL_POLY_C, DL_POLY_3, DL_POLY_4, LAMMPS and GROMACS programs with minimum user's intervention.

(2) Force field editor: DL_FIELD allows user to edit or modify a force field model to produce a customised scheme that is specific to a simulation model. For instance, introduction of pseudo points and rigid body implementation to an otherwise standard potential scheme such as CHARMM, AMBER, etc.

(3) Force field model repertoire: DL_FIELD has a consistent file structure format for all FF schemes and molecular structure definitions. It also allows user to easily expand the existing standard model library to include user-defined molecular models.

(4) Fully automatic identification of the chemical nature of every atom in a molecule, and from such, to assign a universal atomic expression that is independent of the force field schemes (the DL_F Notation).

(5) Force field file conversion to DL_FIELD format: It converts third-party simulations and FF file formats within DL_FIELD and produces FF files for DL_POLY and GROMACS.

1.1 File components

DL_FIELD program package consists of the following files:

(1) DL_FIELD source codes together with a *Makefile*, in the */source* directory

(2) Data library files located in the */lib* directory. These files are

Molecular structure data files, with filenames end with the extension *.sf*.

Potential parameter data files, with filename end with the extension *.par*.

The DL_F notation data file, *DLF_Notation*.

Atom type conversion data file, *dl_field.atom_type*.

Atomic chemical and physical data, *dl_field.atom_data*.

Other supplementary information specific to a force field scheme such as charge increment, atom equivalence table, etc.

(3) DL_FIELD *control* file, of which the name and location is defined in the *dl_f_path* file. The *control* file contains all possible options to setup force field models, including choice of force field schemes and input configuration files.

(5) Solvent templates, in the */solvent* directory.

(6) A collection of utility program scripts, in the */utility* directory.

(7) Converted force field files (such as *dl_poly.CONFIG*, *dl_poly.FIELD*), located in the */output* directory. For example, these files can be renamed as CONFIG and FIELD files, respectively for DL_POLY runs.

(8) The *dlf_notation.output* file in the *output/* directory. This file is generated only if the user's input configuration is in the *xyz* format. It describes the chemical identity of every atom in the system.

(9) Examples of system configurations in PDB, *xyz* and *mol2* formats, located in the */Examples* directory.

(10) The *control_files/* directory contains a collection of DL_FIELD's *control* sample files for structures in the *Examples/* directory.

(11) The status output file, *dl_field.output,* generated during DL_FIELD runs, in the *output/* directory.

(12) The *dl_f_path* file that specifies the location paths for various file components, including the DL_FIELD *control* filename and its location, relative to the DL_FIELD home directory. The home directory is the directory path where the DL_FIELD executable and *dl_f_path* files are located. Alternatively, an absolute location path can be specified. All directory paths can be redefined in the *dl_f_path* file if file components are to move away from the default location. **Note**: if an output location path does not exist, DL_FIELD will automatically create the folder.

User must supply a pre-built atomic configuration file as DL_FIELD does not include most model building features. However, DL_FIELD can setup liquids and solution models based on a single molecular template (See Solution Maker). DL_FILED recognises the following input structure files: standard PDB format, a simple *xyz* format and Tripos *mol2* format.

In addition, DL_FIELD can also read third-party force field files and convert these into DL_FIELD's *udff* file. These files are specified in *dl_f_path* file. For more information please refer to Section 13.

Upon a successful conversion, DL_FIELD will produce the following output files:

(1) The CONFIG file, *dl_poly.CONFIG,* in the */output* directory.

(2) The FIELD file, *dl_poly.FIELD,* in the */output* directory.

(3) The CONTROL file, with the default name *dl_poly.CONTROL,* in the */output* directory. The location and the name of the file can be changed and specify this in the *dl_f_path* file.

(4) The *dl_field.output* file, which reports the status of the conversion process. This is created in the *output/* directory.

(5) Model-specific *udff* model files, *dl_field.udff*. See Chapter 5.3 for more details.

(6) An optional PDB output file can be produced and contains the user's configuration with atoms rearranged according to orders specified by the force field files.

(7) A user-structure file, *dlf_notation.output* in the *output/* directory, if the user configuration is in the *xyz* or *mol2* formats. This file contains atomic structure expressed in the standard DL_F Notation independent of force field schemes.

User will have to move the DL_POLY output files to a directory where DL_POLY simulations can be run and rename them as CONFIG, FIELD and CONTROL, respectively.

If the option to produce GROMACS is also selected, then these additional files will also produce:

(8) The system topology file, *gromacs.top*, in the *output/* folder.

(9) The include topology file(s), *gromacsX.itp* in the *output/* folder, where *X* = 1, 2, … The number of files produce will be dependent on the Molecular Groups defined for the system.

(10) The simulation control parameter file, *gromacs.mdp* in the *output/* folder.

For more details about GROMACS output files, please refer to Chapter 14.

If the option to produce LAMMPS is also selected, then these additional files will also produce:

(11) The system control and input files, *lammps.in*, in the *output/* folder.

(12) Topology and configuration data files, *lammpsX.data*, where *X* = 1, 2, … The number of files produce will be dependent on the Molecular Groups defined for the system.

For more details about LAMMPS output files, please refer to Chapter 15.

1.2 Quick Start Guide

The following instructions show you how to run DL_FIELD.

Step 1: Compile DL_FIELD (Chapter 1.4) – go to the *source/* directory. Type 'make'. This will produce the *dl_field.exe* executable file in the DL_FIELD home directory.

Step 2: Open *dl_f_path* file. This is the first entry point for DL_FIELD to locate various file components to run the program. Make sure you are using the correct control file. By default, this is pointed to *dl_field.control*

control = dl_field.control

Step 3: Save (if require) and close *dl_f_path* and run DL_FIELD, by simply typing:

./dl_field

DL_FIELD will read the configuration file defined in the *dl_field.control* file and setup the corresponding FIELD and CONFIG files in the *output/* directory.

Step 4: You can run some other example structures described in Chapter 16 of the Manual. You need to make sure it is pointed to the correct control file, pre-defined for each example structure.

Open *dl_f_path* and comment out the default *dl_field.control* file and remove the comment as follows:

control = control_files/example_1a.control
 # control = dl_field.control

This will run DL_FIELD and reads the Example 1a structure as described in Chapter 16. All example structures are listed in the *Examples/* folder.

Whenever DL_FIELD is run, it will create a new folder, *dlf_outputX* (where *X* is a number), located in the *output/* directory. This is where all output files, including FF models will be located.

1.3 Force field (FF) Schemes

The following potential models are available in *DL_FIELD*:

(1) The CHARMM potential parameters are derived from the standard ***toppar_c36_jul17*** directory.

(2) The united-atom CHARMM19 potential models for protein are derived from the **c35b2_c36a2** directory.

(3) The AMBER potential parameters for proteins are derived from the *ff03* force field (Duan *et al.* 2003) including the *frcmod.ff03* supplementary file: *all_amino03.lib, all_aminont03.lib* and *all_aminoct03.lib*

(4) The AMBER Glycam potential parameters for carbohydrates and glycoproteins are derived from *Glycam_06h* files.
Kirschner, K.N., Yongye, A.B., Tschampel, S.M., Daniels, C.R., Foley, B.L, Woods, R.J. *J. Comput. Chem.* **29**, 622-655 (2008)

(5) The AMBER's General Amber Force Field (GAFF), Version 2.1 (April 2016), for general organic molecules including small drug molecules.
Wang, J., Wolf, R. M.; Caldwell, J. W.; Kollman, P. A.; Case, D. A. "Development and testing of a general AMBER force field". Journal of Computational Chemistry, 2004 v. 25, 1157-1174.

(6) DL_FIELD supports the enhanced version of OPLS, namely, the OPLS_2005 force field, which covers a larger set of organic functional groups with refitted dihedral parameters.
J. L. Banks *et. al.*, 'Integrated Modelling Program, Applied Chemical Theory (IMPACT)', *J. Comput. Chem.* **26**, 1752-1780 (2005)
W. Jorgensen, D. Maxwell and J. Tirado-Rives, 'Development and Testing of the OPLS All-Atom Force Field on Conformational Energetics and Properties of Organic Liquids' *J. Am. Chem. Soc.*, **118**, 11225–11236 (1996)

(7) OPLS AA/M for protein. This is an improvement force field over OPLS AA/L
M. J. Robertson, J. Tirado-Rives and W. L. Jorgensen, *J. Chem. Theo. Comp.* **11**, 3499-3509 (2015)

(8) OPLS force field for ionic liquid (CL & P)
J.N.C. Canongia Lopes and A.A. H. Padua, 'CL&P: A generic and systematic force field for ionic liquids modelling', *Theor. Chem. Acc.* **131**, 1129 (2012).

(9) OPLS force field for deep-eutectic solvents (OPLS_DES).
B. Doherty and O. Acevedo, *J. Phys. Chem. B*, **122**, 9982-9993 (2018).

(10) DREIDING force field, for general organic molecules.
S. L. Mayo, B. D. Olafson and W. A. Goddard III, 'DREIDING: A Generic Force Field for Molecular Simulations', *J. Phys. Chem.* **94**, 8897-8909 (1990).

(11) Polymer Consistent Force Field (PCFF) is based on CFF91 that includes a wide range of parameters for organic polymers, inorganic metals and zeolites. Used of both ab-initio and empirical methods to obtain parameters base on static simulations at 0 K.
Huai Sun, Stephen J. Mumby, Jon R. Maple, Arnold T. Hagler 'An ab Initio CFF93 All-Atom Force Field for Polycarbonates', *J. Am. Chem. Soc.* **116** 2978–2987 (1994)

(12) Condensed-phase optimized molecular potentials for atomistic simulation studies (COMPASS). It is a general force field for organic molecules. Based on PCFF, it is re-parametrised for condensed-phase systems.
H. Sun 'COMPASS: An ab Initio Force-Field Optimized for Condensed-Phase Applications – Overview with Details on Alkane and Benzene Compounds' *J. Phys. Chem. B* **102**, 7338-7364 (1998).

Note: Only a subset of COMPASS parameters was published. The corresponding missing ATOM_KEYs are still defined in the DL_FIELD library and are tagged with a remark 'Missing'. During runtime, DL_FIELD will bypass the check for these missing parameters for COMPASS.

(13) Transferable Potentials for Phase Equilibria-explicit hydrogen (TraPPE-EH). Potential force field for a collection of organic molecules, with aims to determine thermophysical properties in different phases and compositions.
B.L. Eggimann, A.J. Sunnarborg, H.D. Stern, A.P. Bliss, and J. I. Siepmann 'An Online Parameter and Property Database for the TraPPE Force Field,' *Mol. Simul.* **40**, 101-105 (2014).

Note for using TraPPE-EH: In the original paper, most molecules are modelled as rigid bodies without intramolecular interaction. For this reason, only bond length and angles are specified for the geometry.

However, due to practical considerations, bending, stretching and dihedral parameters are included in DL_FIELD library. Studies show that molecular flexibilities should have only a negligible effect on vapour-liquid equilibria. These parameters are derived from Amber GAFF.

Alternatively, a DL_FIELD's *[RIDIG]* directive can be introduced into the MOLECULE template to setup a rigid molecule.

(14) Transferable Potentials for Phase Equilibria – united atom model (TraPPE-UA). Potential force field for a collection of organic molecules, with aims to determine thermophysical properties in different phases and compositions.
B.L. Eggimann, A.J. Sunnarborg, H.D. Stern, A.P. Bliss, and J. I. Siepmann 'An Online Parameter and Property Database for the TraPPE Force Field,' *Mol. Simul.* **40**, 101-105 (2014).

Note when using TraPPE-UA: Some of bonds are modelled as rigid bond. In DL_FIELD, a harmonic spring with a fix, stiff constant of 1000.0 kcal/mol, to approximate rigid bonds. This also applies to angular 3-body interactions if they are treated as rigid.

(15) Consistent valence force field (CVFF). Originally adopted in the Discover program, it is a general force field fit to small organic crystals and gas phase structures. Can handle peptides, proteins and a range of organic systems.

(16) The united atom model Gromos G54A7.

N. Schmid, *et. Al.* 'Definition and testing of the GROMOS force-field versions 54A7 and 54B7', *Eur. Phys. J.* **40**, 843-856 (2011)

(17) Inorganic force fields, classify according to types of materials: binary oxides, ternary oxides, clay, glass, zeolites, etc.

Version 4.10 onwards also includes Hill-Sauer force field for aluminosilicate zeolites, silica and general silicic molecules (zeolite-Hill Sauer force field).
Jorg-R. Hill and J. Sauer, 'Molecular Mechanics Potential for silica and zeolite catalysts based on ab-initio calculations. 2. Aluminosilicates, *J. Phys. Chem.* **99**, 9536-9550 (1995).

Note: DL_FIELD does not implement cross-correlation interactions (such as bond-bond, bond-angle etc) for PCFF, CVFF, COMPASS and zeolite-Hill Sauer.

1.4 Program Compilation

A standard C compiler must be pre-installed in your operating system. To compile the program, in the */source* directory type and enter the following command:

make

A *Makefile* must be included in the directory where source codes are located. The *Makefile* contains a series of instructions to compile and link the codes.

Upon successful compilation, a *dl_field* executable file will be produced in the *dl_f_X/* home directory, where *X* is the Version number. Type *./dl_field* to run the program.

1.5 DL_FIELD Features and Release History

The following lists illustrate the development history of DL_FIELD. Collectively, it is a complete list of DL_FIELD features shown in a timeline order.

DL_FIELD version 1.0 (released March 2010) contains the following features:

- Convert a PDB structure to CONFIG and FIELD files for DL_POLY.
- CHARMM force fields for proteins, lipids and carbohydrates.
- Freeze and tether atoms assignment.
- Display additional information (such as protein residues etc) in CONFIG and FIELD files.
- Atom label display in either DL_FIELD internal (non DL_F Notation) format or potential-specific format.

DL_FIELD version 1.1 (released November 2010) contains the following additional features:

- Additional CHARMM force fields for lipids, ethers and other small molecules.
- Assign bond constrain and rigid water.
- Additional water models (TIP3P, TIP3P-PME and SPC).

DL_FIELD version 2.0 (released May 2011) contains the following additional features:

- AMBER force field for protein (Duan. et. al. 2003).
- MM energy calculation for user's configuration model.
- Able to switch between different potential schemes without any modification to the PDB file.

DL_FIELD version 2.1 (released October 2011) contains the following additional features:

- OPLS-AA standard force field for protein.
- The user-define force field (udff) file.
- Option to produce a smaller FIELD file size.

DL_FIELD version 2.2 (released April 2012) contains the following additional features:

- DREIDING force field for general molecules.
- Polymer Consistent Force Field (PCFF) for general organic and polymeric molecules.
- Equivalent atom keys (EQUIVALENCE statement).

DL_FIELD version 2.3 (released October 2012) contains the following additional features:

- AMBER's Glycam force field for carbohydrates and glycoproteins.
- Rigid atoms assignment to define rigid bodies.
- Pseudo point assignment.
- User-define scaling factors for 1-4 intra-interactions.
- Periodic boundary conditions.

DL_FIELD version 3.0 (released April 2013) was a major release and contained the following additional features:

- Introduction of inorganic force fields (for simple metal compounds).
- User-selectable energy units (eV, kcal/mol and kJ/mol).
- Random velocity assignment.
- Core-shell models.
- Solvation features with counter-ion insertion.
- Auto-CONNECT features.

DL_FIELD version 3.1 (released November 2013) contains the following additional features

- Update lipid CHARMM force field (all36_lipid).
- CHARMM19 united-atom force field for proteins.
- Directives to introduce/remove angles and dihedrals selectively.
- More solvent choices for solvations, TIPnP, where n = 3, 4 and 5.

DL_FIELD version 3.2 (released June 2014) contains the following additional features.

- Inter-conversion of cell vectors and cell parameters
- Automatic determination of periodic boundary conditions (reads CRYST1 statement in PDB files)
- User-selectable mixing rules for vdw LJ12-6 potential (geometric and arithmetic).
- Include methylated and acetylated terminal for CHARMM proteins.
- Improved warning/error reports.

DL_FIELD version 3.3 (released December 2014) contains the following additional features.

- Improved inorganic force fields including CLAYFF for clay minerals.
- OPLS2005 force field for proteins and general organic molecules.
- Full inclusion of directives to introduce/remove bonds, angles and dihedrals selectively or entirely.
- Miscellaneous improvements on conversion routines and addition of solvent molecules.

DL_FIELD version 3.4 (released October 2015) contains the following additional features.

- Implementation of the standardised atom typing format, the DL_F Notation, for PCFF and OPLS2005 force fields.
- Able to read configurations in simple *xyz* format for PCFF and OPLS2005 force fields, with fully automatic assignment of ATOM_TYPEs in the DL_F notation.
- Additional solvent choices such as SPC and SPC/E water models.

From version 3.4 onwards: ATOM_KEYs, ATOM_TYPEs, MOLECULE_TYPEs, MOLECULE_KEYs and element symbols are now case-sensitive. For example, the molecule keys 'ABC' and 'AbC' are distinct keys for different molecules.

DL_FIELD version 3.5 (released April 2016) contains the following additional features:

- CVFF - Consistent valence force field that implements the DL_F Notation.
- Addition of new Chemical Groups to OPLS2005 and PCFF for the DL_F Notation.
- Inclusion of more solvent templates.
- Some improvement on memory usage.

DL_FIELD version 4.1 (released December 2016) is a major revision and contains the following features:

- Introduction of multiple potentials - can include more than one FF schemes in a system model, such as mixtures of organic force fields or combination of organic/inorganic force fields (PDB input format only).
- Reorganised CHARMM force field into various components with the introduction of additional sets of force fields: CHARMM22_prot, CHARMM36_prot, CHARMM36_lipid and CGenFF (on-going task).

- DL_F Notation development: Include more new Chemical Groups. Extension of the DL_F Notation to CHARMM force fields.
- Better memory usage with the storage of selective force field data.

DL_FIELD version 4.1.1 (released April 2017) is a minor revision mainly to improve certain existing force field schemes as shown below:

- Improvement of INORGANIC force field, with the introduction of three-body term, assignment of freeze and tether atoms (on PDB format only)
- Reorganised INORGANIC force field into various components according to the type of materials: INORGANIC_binary_oxide, binary_halide, ternary_oxide, glass and clay.
- Inclusion of auto-assignment and correction to scaling due to dihedral degeneration, on PCFF and CVFF.
- DL_F Notation development: Include more new Chemical Groups.

DL_FIELD version 4.2 (released January 2018), a revision that contains the following features:

- G54A7 - Gromos united atom force field.
- CHARMM36 force field for carbohydrate (CHARMM36_carb)
- Inclusion of more solvent templates.
- Introduce *VDW_FIX* directive. Overrides vdw parameters of an atom pair instead of using parameters derived from some mixing rules.
- DL_F Notation development: Include more new Chemical Groups.

DL_FIELD version 4.3 (released July 2018), a revision that contains the following features:

- Solution Maker - automatically setup a disorder system such as liquid or solution (further equilibration require).
- Second-tier *EQUIVALENCE* parameter assignments.
- DL_F Notation development: More new Chemical Groups, including complex, large fused aromatic systems and pyranose sugar units (in chair conformations).
- Write out *dlf_notation.output* user structure file (for *xyz* input format only) that contains list of ATOM_TYPEs in DL_F Notation.

DL_FIELD version 4.4 (released January 2019), a revision that contains the following additional features:

- Amber GAFF general force (Amber16 version)
- OPLS-AA/M force field for protein systems.
- DL_F Notation development: More new Chemical Groups.
- Inorganic system configurations expressed in the *xyz* input format.

- Improvement on multiple potential features: Allow Molecular Group definitions and multiple potential capability for *xyz* input format for both organic and inorganic models.
- Custom definition of *control* file and various directory paths (defined in *dl_f_path* file).

DL_FIELD version 4.5 (released June 2019), a revision that contains the following new features:

- User's configuration files in the *mol2* format (for organic systems only).
- Include new vdw mixing rules for multiple-potential systems, which can be independently set for the sigma and epsilon parameters for Lennard-Jones 12-6 functions.
- DL_F Notation development: Include more new Chemical Groups (small non-aromatic heterocyclic groups)
- Introduce *EXCLUSION_14* directive to control 1-4 interaction exclusions in MOLECULE templates.
- Run DL_POLY program via fork process (for Linux/Unix type systems) for single-point calculation and equilibration runs.

DL_FIELD version 4.6 (released January 2020), a revision that contains the following new features:

- OPLS force field for deep eutectic solvents (OPLS_DES)
- Inorganic force field library for zeolites (INORGANIC_ZEOLITE)
- Non-specific, or miscellaneous force field library (MISC_FF)
- Auto-insertion of core-shell models for inorganic materials.
- Auto-identification of MOLECULE template for template-based force field schemes (for *xyz* format only).
- DL_F Notation development: Include more new Chemical Groups (both small and complex heterocyclic groups, some organosilicon groups).

DL_FIELD version 4.7 (released December 2020), a revision that contains the following new features:

- Improve overall program stability and remove memory leaks.
- New force field for ionic liquids (OPLS_CL_P).
- Auto detection of atom types for clay minerals using CLAYFF force field.
- Include new solvent templates.
- DL_F Notation development: Include more new Chemical Groups (mainly azoles and organic ionic species).

DL_FIELD version 4.8 (released September 2021), a revision that contains the following new features:

- For PDB file: Auto-detect a residue whether it forms links with neighbouring residues. The linked symbol, -XX-, is no longer needed in PDB files.

- Van der Waals (vdw) mixing using Slater-Kirkwood formalism (for multiple potential systems).
- Improvement on auto-identification of MOLECULE template for template-based force field schemes (for xyz format only): some capability to distinguish isomers.
- Force field library update: Include more new Chemical Groups and protein data for CHARMM.
- DL_F Notation for inorganic systems.

DL_FIELD version 4.81 (released February 2022) is a minor revision mainly to improve software stability and conversion errors as follows:

- Removed memory leaks, especially for multiple potential model setup.
- Include a few more Chemical Groups (benzothiophenes and quinazolines) and readjust organosilicon and silica detections.
- Adjust biphenyl detection to ensure correct FF assignment for polyaromatics.

DL_FIELD version 4.9 (released November 2022), a revision that contains the following new features:

- Enable solvation by using solvent models from different FF schemes, including multiple potential systems.
- COMPASS force field for general molecules in condensed phase.
- CHARMM36 for nucleic acids, CHARMM36_nucl (in PDB structure format).
- Conversion of Lennard-Jones 9-6 to 12-6 forms and make use of the LJ12-6 vdw mixing schemes in multiple potential systems: Allow mixing of COMPASS and PCFF with other LJ12-6 based FF schemes.
- Improvement on MISC_FF scheme: Enable vdw mixing between different MOLECULE template models and inclusion of additional vdw functions.

DL_FIELD version 4.10 (released December 2023), a major revision that contains the following new features:

- New force field scheme: TraPPE (Transferable Potentials for Phase Equilibria Force Field) for explicit hydrogen (TraPPE_EH), aromatic models.
- New inorganic force field: Hill-Sauer's force fields for aluminosilicate zeolites and general silicic molecules.
- Inorganic force field updates – included new materials for binary oxides, perovskites, pyrochlores (ternary oxides).
- Improved and modified DL_F Notation for inorganic systems with automatic atom typing capability.
- Conversion of Morse to Lennard-Jones 12-6 form and make use of the LJ12-6 vdw mixing schemes in multiple potential systems.
- Format conversion: Convert CHARMM's prm, rtf and psf to DL_FIELD udff.
- Convert FF models for novel molecules obtained from LigParGen and CHARMM's MATCH web servers.

- Inclusions of more organic solvents for setting up solution models.
- DL_POLY run: specify number of processors for MPI runs.

DL_FIELD version 4.11 (released August 2024), a major revision that contains the following new features:

- Produce ready-to-run force field files (matching gro, top and itp) for GROMACS simulation package. See Section 14.
- Automatic run single-point calculations in GROMACS (if supplied) after conversion.
- Inclusions of more organic solvents for setting up solution models (see *solvent_list* file in the *solvent/* folder).
- General force field library update.

DL_FIELD version 4.12 (released May 2025), a major revision that contains the following new features:

- Produce ready-to-run force field files for LAMMPS (matching data and input files).
- Automatic run single-point calculations in LAMMPS (if supplied) after conversion.
- New force field scheme: TraPPE (Transferable Potentials for Phase Equilibria Force Field) for united atom models (TraPPE_UA).
- Include CMAP feature, for CHARMM22_prot and CHARMM36_prot protein models for GROMACS files.
- Addition of new FF parameters – small molecular gas (H2, O2, N2).

DL_FIELD version 4.13 (released June 2026).

- New force field scheme: Amber (2025 version) GAFF general force (amber25_gaff version)
- Write out and read in user model-specific *udff* model files. See Chapter 5.3.
- Include new Chemical Groups and improves CG detections and assignments.
- Write output files in new folders for each run.

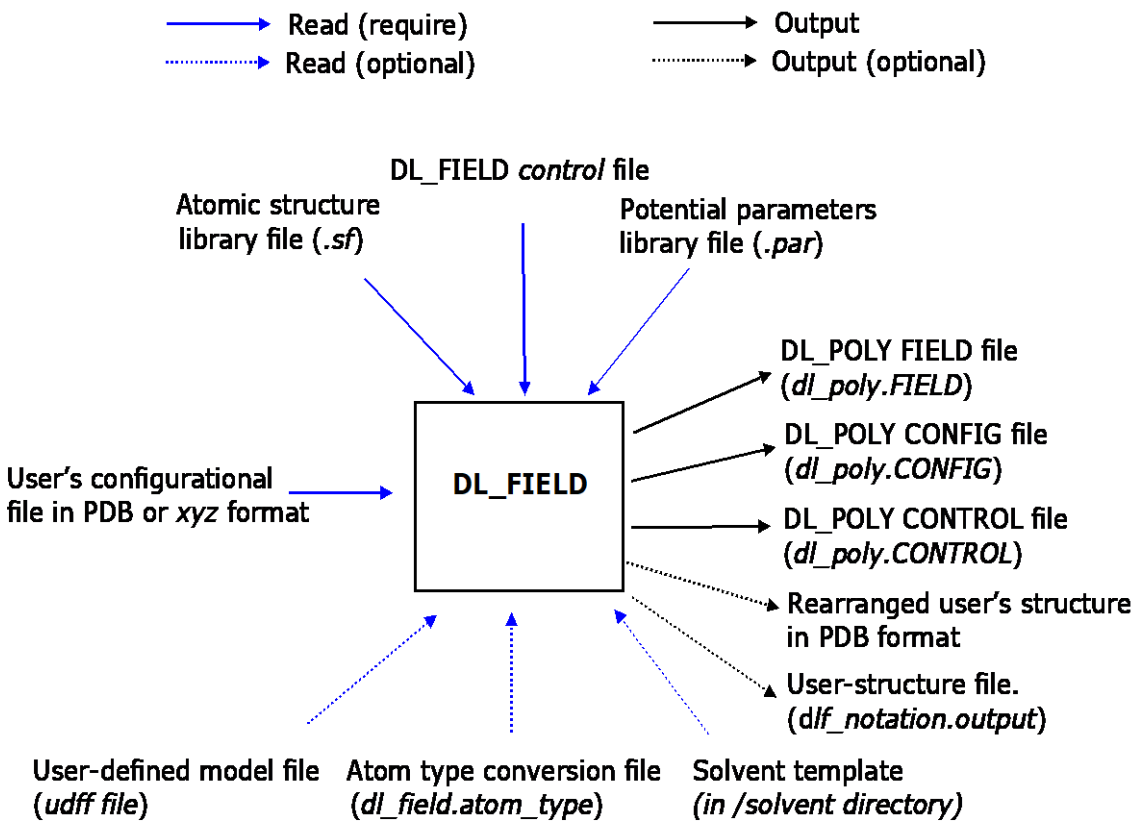
From Version 4.10 onwards, choosing type of solvent would be the filename itself. In addition, the filenames in the */lib* folder have been simplified. These changes do not change the outcome of the FF conversion process.

The MM calculation feature has been removed in the control file since such calculations can be easily obtained from the DL_POLY and GROMACS single-point run.

1.6 DL_FIELD Schematic Flowchart

The main thrust behind DL_FIELD development is to minimise the requirement for users to understand detail knowledge and inner workings of force field descriptions and model preparation procedures. It is intended to be highly agnostic and serve as a user-friendly

tool that can process the molecular information with minimum user's intervention. Diagram below shows schematically the functionality of DL_FIELD.



DL_FIELD reads several directive commands in standard library files (the *.sf* and *.par* files) and in DL_FIELD *control* file. The use of these directives is crucial to produce FF output files. DL_FIELD also produces generic control files. For DL_POLY_4, these files can be renamed as *CONFIG*, *FIELD* and *CONTROL* respectively for simulation runs.

If the option to produce GROMACS force field files is also selected, then DL_FIELD will also produce concurrently a set of matching top, itp and gro files. A generic mdp control file will also produce.

If the option to produce LAMMPS FF files is also selected, then DL_FIELD will also produce concurrently a set of matching LAMMPS input and data files.

Directive commands in DL_FIELD will be explained in detail in the following chapters and they appear in capitals as ***BOLD_ITALICS***. Directives take parameter data will appear as *italics*. Optional directives will be enclosed in the *[**SQUARE BRACKET**]*.

1.7 DL_FIELD Definitions

The following section describes definitions specific to DL_FIELD. These terms appear as CAPITAL letters in the Manual.

(1) MOLECULE - An entity consists of a collection of ATOMs, which consist of normal atoms, pseudo points and atoms with core-shell components, or even united atoms and coarse-grain particles, that are defined within the ***MOLECULE*** and ***END MOLECULE*** directives. It is also called residues, similar those of amino acid components that made up a protein molecule. It can be either a complete molecule or a molecular fragment that can form bonds with other MOLECULEs. It can also refer to as a single atom or ion.

(2) MOLECULE_TYPE - The name of a MOLECULE, often in a human-readable form.

(3) MOLECULE_KEY - The corresponding code (up to four characters) for a MOLECULE_TYPE. This is equivalent to the amino acid residue code as appears in PDB files.

(4) ATOM - An entity or member that made up a MOLECULE. It can be a normal atom, ion, a pseudo point, core and shell part of an atom, united atom or coarse-grain particle etc. They are listed within a ***MOLECULE*** and ***END MOLECULE*** directives.

(5) ATOM_TYPE - The name or identity of an ATOM, often in a human-readable form, distinguishes by its chemical nature or ATOMic characteristics within a MOLECULE. This is different from an element whereby there can be several ATOM_TYPEs for a given element. An element in different molecules can have different chemical behaviour due to its chemical environment. Different atom definitions are therefore needed and mapped to different sets of potential parameters to reflect the differences in chemical behaviour. For instance, an aliphatic carbon would be chemically different than, say, a carbonyl carbon. Two different ATOM_TYPEs will have to define to reflect these differences.

(6) ATOM_KEY - The abbreviation symbol for an ATOM_TYPE and is used to map to the corresponding potential parameters. ATOM_KEYs are defined as 'atoms', or the atom labels that would appear in the *CONFIG* and *FIELD* input files for DL_POLY.

(7) ATOM_LABEL - The unique label of an ATOM, visible only within a ***MOLECULE*** directive. In other words, it is used to distinguish the ATOMs from one another within a MOLECULE.

(8) CONNECT - The way how an ATOM is related to other ATOMs in a MOLECULE. DL_FIELD recognises three different ways: normal-CONNECT between ATOMs, self-CONNECT and auto-CONNECT. These are distinguished from one another by using the appropriate ***CONNECT*** directives.

(9) ORGANIC force field - A general reference to a group of force field schemes that is used to model covalent molecules such as amino acids, sugars and other types of organic molecules. Examples of such are CHARMM, AMBER and OPLS-AAM etc.

(10) INORGANIC force field - A general reference to a group of force field schemes for inorganic molecules. See Chapter 9 for more information.

(11) Potential (or force field (FF)) Scheme – Refer to a force field data set, which are usually given a name to identify the data set. For example, CHARMM, AMBER and OPLS-AA are three well-known force field schemes to model proteins and organic molecules in the condensed phase simulations.

(12) DL_F notation - The DL_FIELD standard ATOM_TYPE notation for a range of FF scheme including CHARMM, PCFF, CVFF, OPLS force fields, etc. It is the universal atom typing form that encodes with chemical information of which the force fields share the common ATOM_TYPE and ATOM_KEY descriptions for a given system. See Chapter 12.

(13) Directive – A command that is recognisable by DL_FIELD to perform a function, or as an indication of types of data. Directives mostly appear in the library files but also in the DL_FIELD's *control* files, as well as in the user's configuration files. Directives are described in ***bold italics***, and in [***square brackets***] for optional directives.

(14) Control Options – The force field model conversions are driven by a set of control Options in the *control* files. The sequence of these Options is fixed and they are expressed as Option **X** in the Manual, where **X** is the unique numerical value for different controls.

2 Molecular Structure File (*.sf*)

A structure file is a standard library file expressed in the ASCII text format where all ATOM_TYPEs and MOLECULEs are defined. It is like the CHARMM's topology file or the Amber's 'prep' file.

The file is free format, but each line is restricted to 120 characters in length. It contains several simple directives. Different types of definitions were enclosed within a ***DIRECTIVE*** and the ***END DIRECTIVE*** block.

The available ***DIRECTIVE***s are shown below:

2.1 ATOM_TYPE and END ATOM_TYPE

Enclosed within these directives is where all the ATOM_TYPEs of a given potential scheme are defined.

Each entry must take five parameters:

(a) ATOM_TYPE name - This can be anything up to 50 characters in length (case-sensitive).

(b) ATOM_KEY - This can be anything up to 15 characters in length (case-sensitive). This is the atom label that will show in the *dl_poly.CONFIG* and *dl_poly.FIELD* files. The ATOM_KEY can be any alphanumeric characters. The exception being the character 'X', which is reserved by the DL_FIELD to indicate a generic or a representation of 'any-atom' in the parameter (*.par*) files.

(c) Element - The element symbol (case-sensitive) of an ATOM_TYPE according to the standard Periodic Table of the Elements. However, DL_FIELD allows user-define 'element' symbols, such as those for pseudo points and united atoms.

(d) Relative atomic mass - This value can be shown up to 3-4 significant figures.

(e) Remark - This can be anything and is ignored by DL_FIELD. It is used to provide brief description to an ATOM_TYPE.

An example of ***ATOM_TYPE*** usage is shown below (extracted from *DLPOLY_PCFF.sf*)

```
ATOM_TYPE   key element mass      remark (Ver. Ref.)
Ct_alkane     c1   C       12.0115   sp3 carbon -CH (1.0 1) - connect to three C
Cs_alkane     c2   C       12.0115   sp3 carbon -CH2 (1.0 1) - connect to two C
Cp_alkane     c3   C       12.0115   sp3 carbon -CH3 (1.0 1) - connect to one C
…
…
END ATOM_TYPE
```

The example above defined three different kinds of carbon atoms: ATOM_TYPEs Ct_alkane, Cs_alkane and Cp_alkane, referring to tertiary, secondary and primary alkyl carbon atom, respectively. The corresponding ATOM_KEYs are c1, c2 and c3 respectively.

Note that each ATOM_TYPE is unique, whereas the corresponding ATOM_KEY is not. In other words, more than one ATOM_TYPEs can share a same ATOM_KEY. DL_FIELD will flag up an error if more than one identical ATOM_TYPEs are defined.

2.2 MOLECULE_TYPE and END MOLECULE_TYPE

This directive set displays a list of available MOLECULE templates in the *.sf* file. During the conversion process, DL_FIELD extracts the MOLECULE_KEYs from the user configuration (PDB file format) and then look for the corresponding MOLECULE_TYPEs. The latter information is required for DL_FIELD to look for the correct MOLECULE templates, which are defined by the ***MOLECULE*** and ***END MOLECULE*** directives.

The template list is useful for users to look up what MOLECULE templates are available in a structure file. The order of the MOLECULE_TYPEs in the list is not important. They can be grouped according to types of MOLECULEs, for example, into amino acids and carbohydrates. Alternatively, MOLECULEs can be grouped according to functional groups or molecular weight, etc.

For configurations expressed in *xyz* and *mol2* formats, DL_FIELD does not need the MOLECULE_TYPE information or MOLECULE template definition. The exception would be for those template-based FF schemes such as AMBER and CHARMM. In the latter cases, DL_FIELD will identify molecules present in the system and match against the MOLECULEs in the library files.

In the case of inorganic systems in *xyz* format, the MOLECULE_KEY must always be specified. See Section 6.2 for more details.

Each entry within the ***MOLECULE_TYPE*** directive must take four parameters (a), (b), (c), and (d) as shown below:

(a) MOLECULE_TYPE - This can be anything up to 30 characters in length. This is the molecule name (often in a human-readable form) that identifies a MOLECULE.

(b) MOLECULE_KEY - The unique identity code of the MOLECULE up to four character long. This code is used to identify a MOLECULE in the user's atomic configuration files such as a PDB file. It is equivalent to the protein residue labels in a PDB file. Note that to conform to the PDB file format, MOLECULE_KEYs must have up to 4 characters, though DL_FIELD allows more than that. In addition, all MOLECULE_KEYs are case-sensitive. In other words, MOLECULE_KEYs HIS and HIs are treated as two different keys that corresponds to two different MOLECULEs.

For some MOLECULEs, such as those of carbohydrates, nuclei acids and amino acids, DL_FIELD will *automatically* determine if they are whole molecules or formed part of linkages as molecular residues. In the latter case, DL_FIELD will modify the MOLECULE_KEY to -ABC- where ABC is the original key defined in the user's configuration file.

For instance, in a PDB file, the amino acid residue ALA (alanine) which formed part of a protein, will be recognised by DL_FIELD as -ALA-. For AMBER's Glycam molecule, the molecule residue 3EA (3-α-D-altropyranose terminal) will be renamed as -3EA-. DL_FIELD will automatically search and assign additional potential parameters involving bond linkages that straddle across two MOLECULEs.

(c) Relative molecular mass - This can be calculated based on some standard atomic weight. DL_FIELD need this information for amino acid residues to determine various charge states of some amino acids such as histidine and aspartic acid.

(d) Remark - Brief description of the MOLECULE. This information is ignored by DL_FIELD.

An example of the ***MOLECULE_TYPE*** directive is shown below (extracted from the *DLPOLY_CHARMM.sf* file):

```
MOLECULE_TYPE       key      mw      remark
ethane              ETHA    30.071   ! C2H6
propane             PRPA    44.098   ! C3H8
hexane              HEXA    86.181   ! C6H14
…
…
END MOLECULE_TYPE
```

DL_FIELD will always check that there is no repeat of MOLECULE_TYPE and MOLECULE_KEY. In other words, each MOLECULE_TYPE is uniquely referred to a MOLECULE_KEY. This contrasts with ATOM_TYPEs and ATOM_KEYs, where different ATOM_TYPEs can share the same ATOM_KEY.

2.3 MOLECULE and END MOLECULE

This directive block is perhaps the most encountered in an *sf* file. This is where a MOLECULE's structure and model states are defined, called a MOLECULE template. Only one MOLECULE template can be defined within the directive set. It is used to match with the user's PDB structure and, from such, to derive the corresponding force field model state.

A MOLECULE definition will always take the following format and sequence order:

```
MOLECULE MOLECULE_TYPE  total_atom  total_charge
ATOM_LABEL1             ATOM_TYPE1      charge
ATOM_LABEL2             ATOM_TYPE2      charge
....
ATOM_LABELn             ATOM_TYPEn      charge
CONNECT         ATOM_LABEL1   > ATOM_LABEL …
CONNECT         ATOM_LABEL2   > ATOM_LABEL …
…
CONNECT         ATOM_LABELn   > ATOM_LABEL …
[RIGID]         ATOM_LABEL1 ATOM_LABEL2 …
[DIHEDRAL REMOVE/ONLY]  ATOM_LABEL1 ATOM_LABEL2 ATOM_LABEL3 ATOM_LABEL4
[ANGLE REMOVE/ONLY]     ATOM_LABEL1 ATOM_LABEL2 ATOM_LABEL3 ATOM_LABEL4
[IMPROPER]      ATOM_LABEL1 ATOM_LABEL2 ATOM_LABEL3 ATOM_LABEL4
[INVERSION]     ATOM_LABEL1 ATOM_LABEL2 ATOM_LABEL3 ATOM_LABEL4
[SHELL]  ATOM_LABEL_core ATOM_LABEL_shell
…
…
END MOLECULE [some information or remark]
```

Each molecular definition must always end with the ***END MOLECULE*** directive. However, the optional directives can appear in any sequence.

The *total_atom* indicates the total number of ATOMs in the MOLECULE and each ATOM must be clearly defined. The *total_charge* is the net charge of the MOLECULE, by summing up all the individual partial charges of ATOMs.

Each ATOM must have the unique ATOM_LABEL. This is followed by the corresponding ATOM_TYPE, as defined in the ***ATOM_TYPE*** directive. Finally, the *charge* value of the ATOM must also be defined. If the total charge is set to some 'ridiculously' large value (10.0 or above), then DL_FIELD will not look for the charge values of the individual ATOMs. Instead, charge values will be determined from the bond charge increment data. This usually applies to force field schemes such as PCFF, CVFF and OPLS2005.

The connectivity information must also be given, immediately after the ATOM definitions. The connectivity of each ATOM must be clearly defined using the directive ***CONNECT***. The sequence of ***CONNECT*** ATOM statements must follow strictly that of ATOM sequence defined earlier. ATOM_LABELs after the '**>**' symbol indicate the neighbouring ATOMs that are connected to the ATOM.

2.3.1 Bond Connectivity

There are three different types of connections: (1) the normal bond connection between ATOMs, (2) the self-CONNECT and (3) the auto-CONNECT. These different types of connections are distinguished from one another by using the ***CONNECT*** statement in different formats. This is shown below:

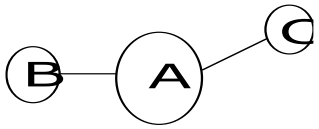
(a) Normal CONNECT

The directive ***CONNECT*** means a physical (covalent) bond is established between two ATOMs when it is defined as follows:

CONNECT A **>** B C

CONNECT B **>** A

CONNECT C **>** A



The directive statements above define bonds A-B and A-C and DL_FIELD will automatically search for all angles and dihedrals, if any.

(b) Self-CONNECT

When an ATOM is defined with this type of connection, DL_FIELD will assume there is no physical bond connected to this ATOM and, hence, no angle and dihedral. This essentially treats it as an isolated ATOM within the MOLECULE and DL_FIELD will only assign the charge and vdw parameters to the ATOM:

CONNECT A **>** A

When defining a self-CONNECT ATOM, the correct ATOM_KEY ***must be used*** instead of ATOM_LABELs. The same ATOM_KEYs must be used also as the atom labels in the user's PDB file.

For ORGANIC force field, self-CONNECT only valid for pseudo ATOMs and the shell part of a polarisable atom.

For INORGANIC force field, self-CONNECT is normally used since ionic materials are largely interacted via non-bonded interactions. Example below shows how to define a self-CONNECT ATOM. Note that The ATOM_KEY OP must be used also as the label in the PDB file.

```
ATOM_TYPE
O_phosphate    OP    O    15.9994    ! Phosphate O
END ATOM_TYPE

MOLECULE an_example   10  0.0
OP   O_phosphate   -0.3
....
CONNECT  OP >  OP
...
END MOLECULE
```

(c) Auto-CONNECT

This type of connection instructs DL_FIELD to determine the bond connectivity automatically and assign all the necessary 2- (bond), 3- (angle) and 4-bodies (dihedral) interactions accordingly. The general syntax is as follows:

CONNECT ATOM_KEY > *number_of_neighbours* ***AUTO***

Note that, similar to the self-CONNECT, ATOM_KEYs *must be used*, instead of some ATOM_LABELs in the MOLECULE templates and the user's PDB file.

The directive ***AUTO*** is required to indicate the ***CONNECT*** statement is of auto-CONNECT type.

The auto-CONNECT is a powerful feature to construct MOLECULEs consist of many ATOMs, or MOLECULEs with no definite connection order. Examples of such are random polymers, hydrogels, molecules with complex inter-connectivity such as networked polymers and graphenes. While these MOLECULEs can be constructed by using the normal CONNECT feature, use of the auto-CONNECT type is less error-prone, and produced a simpler MOLECULE definition than using the normal CONNECT feature.

The *number_of_neighbours* indicates the number of bonds that are expected to connect to the ATOM. DL_FIELD will flag up an error if the number of neighbours in the user's configuration does not match with what is defined in the ***CONNECT*** statement.

The *number_of_neighbours* can also be replaced with an asterisk, *, to indicate either an unknown or variable number of neighbours. Example of such use is shown for the CLAYFF in the *DLPOLY_INORGANIC_clay.sf* file. Care must be taken when such feature is used, and users must ensure all bond connections are properly accounted for.

The auto-CONNECT feature is also useful for certain ORGANIC force fields to construct a MOLECULE to represent a class of MOLECULEs that share a common functional group. An example is shown below (extracted from *alcohol.udff* in the *Examples/* directory):

```
MOLECULE_TYPE
...
aliphatic_alcohol        ROH     999.000    Aliphatic alcohol, CnH2n+1–OH, n is not 1
...
END MOLECULE_TYPE

MOLECULE aliphatic_alcohol 6 -0.22       Any aliphatic alcohol, except methanol
CT3  Cp_alkane    -0.27
CT2  Cs_alkane     0.05            H1   H1    H1
CT1  Ct_alkane     0.14             |   |     |
HA   HC_alkane     0.09        H1-C3-C2-...C2-OH-HO
OH1  O_alcohol    -0.66             |   |     |
H    HO_alcohol    0.43           H1  H1    H1
CONNECT CT3 > 4  AUTO
CONNECT CT2 > 4  AUTO
CONNECT CT1 > 4  AUTO
CONNECT HA  > 1  AUTO
CONNECT OH1 > 2  AUTO
CONNECT H   > 1  AUTO
END MOLECULE
```

The example above shows that the MOLECULE aliphatic_alcohol represents all types of aliphatic alcohols (all primary, secondary and tertiary) such as ethanol, propanol, pentanol, etc. Unlike the MOLECULEs contain the normal CONNECT type, DL_FIELD permits the use of a subset number of ATOMs within the MOLECULEs with the auto-CONNECT type. For instance, branched alcohols such as propan-2-ol would need the ATOM_TYPE Ct_alkane, whereas linear alcohols do not. DL_FIELD will choose the suitable ATOM_TYPEs according to the user's input structures.

Example uses of the auto-CONNECT feature are illustrated for alcohol structures in the *Examples/* directory (*ethanol_auto1.pdb, ethanol_auto2.pdb* and *alcohols.pdb* files). Note that they require the *udff* file called *alcohol.udff* for DL_FIELD to convert these structures. To use the *udff* feature, turn it on by providing the correct *udff* file location, such as *Examples/alcohol.udff*, in the *control* file. Remember to select the CHARMM22_prot as the FF scheme.

Example uses of the auto-CONNECT feature for inorganic materials can be found in the ternary oxides library such as the MOLECULE calcium_carbonate1 template.

2.3.2 Optional directives

The following directives can appear in any order within a MOLECULE definition.

(a) [***IMPROPER***] directive

This directive defines improper potentials. It reads four ATOM_LABELs with three of them linking to a fourth central atom.

(b) [***INVERSION***] directive

This directive indicates the requirement of inversion potentials. It reads four ATOM_LABELs with three of them linking to a fourth central atom.

(c) [***RIGID***] directive

Note: this directive is not applicable to GROMACS.

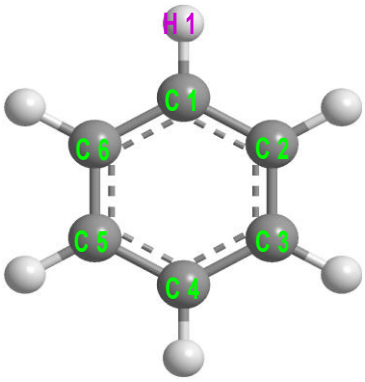
This directive defines rigid ATOMs in a molecule. Each [***RIGID***] statement defines collectively a group of ATOMs as a single rigid body.

For the [***RIGID***] statements defined within a MOLECULE to become active, users must also specify a [***RIGID***] directive to the Molecular Group to which the MOLECULE belongs, in the *control* file. In other words, the [***RIGID***] statements specified within the ***MOLECULE*** defines the rigid body of the molecule, while the [***RIGID***] directive specified in the *control* file decides whether the rigid body should be implemented for the MOLECULEs that belongs to a Molecular Group.

Consider a MOLECULE benzene with all the carbon ATOM_LABELs as shown below and let's suppose all six carbon atoms are to be defined as a single rigid body. The [***RIGID***] directive within the MOLECULE is defined as follows:

```
MOLECULE benzene 12 0.0     Benzene
C1  C_benzene    -0.115
H1  HC_aromatic   0.115
C2  C_benzene    -0.115
H2  HC_aromatic   0.115
C3  C_benzene    -0.115
…
…
CONNECT C4 > C3 C5 H4
CONNECT H4 > C4
CONNECT C5 > C4 C6 H5
CONNECT H5 > C5
CONNECT C6 > C1 C5 H6
CONNECT H6 > C6
RIGID C1 C2 C3 C4 C5 C6
END MOLECULE BENZ
```

The order of the carbon atoms is not important. The [***RIGID***] directive defines a rigid ring and, except for vdw and coulombic interatomic interactions, all intra-interactions that involve members of the rigid ATOMs will be excluded in force calculations. The intra-interactions (bonds, angles and dihedrals) among the rigid atoms will not be shown in the *dl_poly.FIELD* file.



Note that hydrogen atoms are still flexible relative to the rigid ring. The following [***RIGID***] directive defines the H1 hydrogen atom to form part of the rigid body:

RIGID C1 H1 C3 C2 C6 C4 C5

DL_FIELD Limitation:
Each [***RIGID***] statement in DL_FIELD refers to a single rigid unit within a MOLECULE can only contain up to 15 atoms. This contrasts with DL_POLY, which allows more than that but confines to 15 atoms in a statement line in the *FIELD* file.

Let's suppose the benzene molecule in a user molecular configuration file is assigned to a Molecular Group ORG (can be any name), the rigid body option must be switched on (Option **24**) and the ORG Molecular Group must be specified in a DL_FIELD *control* file:

```
…
…
1       * Tether atoms? 1 = Yes (see below)  0 = No
0       * Constrain bonds? 1 = Yes (see below) 0 = No
1       * Apply rigid body? 1 = Yes (see below) 0 = No
…
…
#############################################################
Atom state specification:  type  Molecular_Group   filter   [value]

RIGID ORG

TETHER …

#############################################################
```

(d) [***SHELL***] directive

The directive [***SHELL***] introduces core-shell polarisability effect to an ATOM within a MOLECULE. The syntax to define a core-shall model of an ATOM is as follows:

SHELL ATOM_LABEL_*core* ATOM_LABEL_*shell*

The example above shows DL_FIELD will consider ATOM_LABEL_*core* and ATOM_LABEL_*shell* as the core and shell component of an atom respectively.

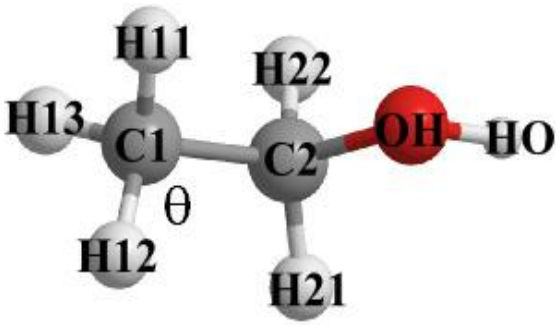
Core-shell models are usually available for inorganic models. However, DL_FIELD permits core-shell models on all organic potential schemes (see Section 11.2). For details about core-shell implementations on inorganic materials, please see Section 11.2.

(e) [***ANGLE ONLY***], [***ANGLE REMOVE***] and [***ANGLE OFF***] directives

These are optional angle exclusion directives. Only one of these directives is allowed for a MOLECULE definition. By default, once all ATOMs within a MOLECULE

template are matched with the user's configuration, all possible sets of angles contained within a MOLECULE will be identified and assigned with the corresponding potential parameters for the angle interactions.

However, users can selectively remove a specific set of angles by using the [***ANGLE REMOVE***] directive. This directive instructs DL_FIELD to ignore the assignment of potential parameters to the angle set. Consider a MOLECULE ethanol with ATOM_LABELs as shown in the sketch below:



By issuing the directive statement, say, [***ANGLE REMOVE***] H12 C1 C2, it tells DL_FIELD to ignore the angle interaction θ (H12-C1-C2) but still consider the rest of the interactions involving all other angles.

If the directive statement [***ANGLE ONLY***] H12 C1 C2 is issued, this instructs DL_FIELD to assign *only* the angle potential interactions about H12-C1-C2 and ignore the rest.

If the directive statement [***ANGLE OFF***] is issued, then *all* angles will be ignored. Note that this directive does not take any ATOM_LABEL as the input parameter.

(f) [***DIHEDRAL ONLY***], [***DIHEDRAL REMOVE***] and [***DIHEDRAL OFF***] directives

These optional directives determines dihedral angle exclusions. They function exactly the same way as that of ***ANGLE*** directives. The syntax is as follows:

DIHEDRAL ONLY LABEL1 LABEL2 LABEL3 LABEL4
DIHEDRAL REMOVE LABEL1 LABEL2 LABEL3 LABEL4
Or
DIHEDRAL OFF

Note that the 1-4 non-bonded interactions between LABEL1 and LABEL4 still apply, even if the dihedral angle involving these ATOMs is excluded.

(g) [***BOND ONLY***], [***BOND REMOVE***] and [***BOND OFF***] directives

These optional directives determine bond exclusions. They function exactly the same way as that of ***ANGLE*** directives. The syntax is as follows:

BOND ONLY/***REMOVE*** ATOM_LABEL1 ATOM_LABEL2
Or
BOND OFF

(h) [***THREE-BODY***] directive

This directive defines three-body interactions with the following syntax

THREE-BODY ATOM_LABEL1 ATOM_LABEL2 ATOM_LABEL3

Once a three-body interaction is defined, DL_FIELD will look for the corresponding parameters in the *.par* file, under the directive ***THREE_BODY***. An example usage of the three-body interactions can be found for MOLECULE soda-lime_silicate1 in the *DLPOLY_INORGANIC_glass.sf* file.

(i) [***EXCLUSION_14***] directive

An atom pair that is separated by three bonds will usually be treated with vdw and coulombic interactions, called the 1-4 non-bonded interactions. Quite often, such interactions are also scaled with some scaling factors. With this directive, such interactions can be switched off selectively within a MOLECULE. The syntax is as follows:

EXCLUSION_14 ATOM_LABEL1 ATOM_LABEL2 …

Each statement includes a group of ATOM_LABELs and exclusions will apply if any 1-4 atom pairs were identified within the atom group. By default, up to a maximum number of 15 atoms (set by MAX_EXCLUSION_14 in *dl_field.h*) can be specified in a statement.

More than one ***EXCLUSION_14*** statement is allowed within a MOLECULE and they are independent from each other. In other words, the 1-4 exclusions **only** apply to the group of atoms defined in each statement.

(j) [***CMAP***] directive

Also called correction map, this directive is only applicable to GROMACS FF files and if either CHARMM22_prot or CHARMM36_prot FF scheme is used. Atoms involve in CMAP are usually those consist of protein backbones, to improve the dihedral representation of protein models. This is done by adding correction terms to the standard dihedral potentials by adding a two-dimensional grid-based energy term that depends on the φ/ψ backbone dihedral angles.

The syntax is as follows:

CMAP ATOM_LABEL1 ATOM_LABEL2 … ATOM_LABEL5

Each CMAP statement consists of five atoms where the first and the fifth are atoms located at the adjacent (before and after) of the current residues. If CMAP atom set is detected, then DL_FIELD will create a special itp file called *gromacs_cmapX.itp*, which contains sets of correction values for dihedral angle intervals, spanning the whole range, from -180° to 165° at 15° intervals. *X* is the Molecular Group number, to indicate CMAP will be used for atoms located in Molecular Group *X*. This means that atoms where CMAPs are applied are defined in *gromacsX.itp* file.

The CMAP data is in the *lib/* folder (for example, *CHARMM22_prot.cmap*) and DL_FIELD will extract the CMAP information and rewrite out into a compatible *gromacs itp* file.

2.4 Example of a Structure File

Below shows a typical example of a *sf* file composition, extracted from the *CHARMM22_prot.sf*.

```
ATOM_TYPE
...
...
HC_alkene       HE2       H        1.00797   ! 12 alkene H2C=CR (generic)
C_aldehyde      CD        C        12.0115   ! 24 carbonyl C (top_all22_model)
O_aldehyde      O         O        15.9994   ! 80 carbonyl oxygen
HC_aldehyde     HR1       H        1.00797   ! 46 his he1, (+) his HG,HD2
Np_amine        NH2       N        14.0067   ! 75 amide hydrogen
HN_amine        HC        H        1.00797   ! 42 N-ter H
C_amide         CC        C        12.0115   ! 64 carbonyl C, asn,asp,gln,glu,cter,ct2
Cs_amide        C         C        12.0115   ! 52 carbonyl C, peptide backbone
O_amide         O         O        15.9994   ! 80 carbonyl oxygen
Np_amide        NH2       N        14.0067   ! 75 amide nitrogen
...
END ATOM TYPE
.
.

MOLECULE_TYPE
...
...
acetamide         ACEM     59.068    From CHARMM top_all22_model.inp
...
...
END MOLECULE_TYPE
.
.
MOLECULE acetamide 9 0.0   methyl amide (Acetamide)
C1  Cp_alkane      -0.27
H11 HC_alkane       0.09
H12 HC_alkane       0.09       H11 O      H1
H13 HC_alkane       0.09         |  ||    /
C   C_amide         0.55   H12-C1--C--N
O   O_amide        -0.55          |        \
N   Np_amide       -0.64       H13        H2
H1  HN_amide        0.32
H2  HN_amide        0.32
CONNECT C1  > H11 H12 H13 C
CONNECT H11 > C1
CONNECT H12 > C1
CONNECT H13 > C1
CONNECT C   > C1 O N
CONNECT O   > C
CONNECT N   > C H1 H2
CONNECT H1  > N
CONNECT H2  > N
IMPROPER C  C1 N  O
IMPROPER C  N  C1 O
IMPROPER N  C  H1 H2
IMPROPER N  C  H2 H1
END MOLECULE
```

The layout is similar to the CHARMM topology file. A molecule sketch (or any other information) is permitted alongside the ATOM definitions. This information is ignored by DL_FIELD. However, the ***CONNECT*** directive must only contain DL_FIELD readable connectivity information.

Do take extra care when constructing a MOLECULE as it completely defines the state of a molecular model. DL_FIELD has been programmed to detect many possible typo and

syntax errors in the MOLECULE templates. However, it is not able to detect what should (or should not) be included. For instance, no error will flag up if an amide group is defined without the [***IMPROPER***] or [***INVERSION***] directive although this may be needed, to keep the amide functional group in a planar configuration.

All user-defined MOLECULEs can be defined in a separate file called the *udff* file (See Chapter 5). Users are strongly recommended to do so since this is the only way to safely include new MOLECULEs without having to tamper with the standard force field library files included in the DL_FIELD software.

3 Parameter File (.par)

The parameter file is a standard force field library file expressed in the standard ASCII text format and contains all atomic interaction parameters. All *.par* files are located in the directory */lib*.

The DL_FIELD's parameter file is similar to CHARMM's parameter file or AMBER's .dat file. Each line is restricted to 120 characters in length. It contains several directives that indicate various types of potential interactions. Each type of potential interaction is grouped into a block and enclosed within the appropriate ***DIRECTIVE*** and the corresponding ***END DIRECTIVE***. Each potential type can only have one set of ***DIRECTIVES***. For example, there can be only one set of ***BOND*** and ***END BOND*** block within a parameter file. All bond parameters can only be defined within this ***DIRECTIVE*** block set.

The order of the ***DIRECTIVE*** block sets is not important. All ***DIRECTIVES*** have the same functionality for all potential schemes. However, the order and the type of potential parameters appear within each ***DIRECTIVE*** may vary according to the potential schemes. In addition, some functional forms of a potential type may also be different depending on the potential schemes. Wherever possible, the way the parameters listed in the *.par* file is the same as that of original data library file of a given potential scheme. DL_FIELD will carry out all the necessary conversion internally to produce equivalent parameter sets in the FIELD file for DL_POLY.

For example, in DL_POLY, the spring constant is defined as $0.5k_b$ and DL_FIELD will rescale this according to the selected potential scheme, so that the potential parameters for that scheme can be entered 'as is' into the DL_FIELD's parameter file. For instance, in the CHARMM force field parameter files, the CT3-CT3 bond defines k_b = 222.5 kcal/mol. DL_FIELD will rescale this as 445.0 in the FIELD file. Internally, DL_POLY will see this as 0.5 x 445.0 when running the simulation.

The following directives are used in a typical *.par* file.

3.1 *UNIT* directive

This directive indicates the energy unit that applies to all parameters in the file. It only takes one variable:

UNIT *energy_unit*

The available options are *kcal/mol*, *eV*, *kJ/mol* and *K*. If the energy unit option (Option **2** in the *control* file) is specified other than what is defined in the *.par* file, DL_FIELD will carry out unit conversions automatically for all potential parameters before it is written to the *dl_poly.FIELD* file.

For GROMACS output files, the energy unit is always converted to kJ/mol and in unit nm for the distance.

For LAMMPS output files, the energy unit is always converted to kcal/mol.

3.2 *POTENTIAL* and *END POTENTIAL* directives

This directive block indicates the type of potential scheme. Enclosed within these directives contain all types of interactions for the force field. The directive POTENTIAL only takes one variable, the *potential_scheme:*

POTENTIAL *potential_scheme*

The available *potential_scheme* in DL_FIELD are as follows:

Name of potentials	***potential_scheme***
All-atom CHARMM*	CHARMM
CHRMM22 for proteins	CHARMM22_prot
CHARMM36 for amino acids, proteins	CHARMM36_prot
CHARMM36 for nuclei acids	CHARMM36_nucl
CHARMM36 for lipids	CHARMM36_lipid
CHARMM36 for carbohydrates	CHARMM36_carb
CHARMM general force field	CHARMM36_cgenff
United-atom for CHARMM	CHARMM19
All-atom Amber	AMBER
AMBER general force field	AMBER16_gaff, AMBER25_gaff
OPLS-AA	OPLSAA
OPLS_2005	OPLS2005
OPLS AA/M for proteins	OPLS_AAM
CL&P for ionic liquids	OPLS_CL_P
OPLS for deep eutectic solvent	OPLS_DES
DREIDING	DREIDING
Polymer consistent force field	PCFF
Consistent Valence Force Field	CVFF
COMPASS force field	COMPASS
Transferable potentials (explicit hydrogen) for phase equilibria	TRAPPE_EH
Transferable potentials (united atom) for phase equilibria	TRAPPE_UA
Gromos united atom G54A7	G54A7
Inorganic force field**	INORGANIC
Inorganic FF for binary oxides	INORGANIC_binary_oxide
Inorganic FF for ternary oxides	INORGANIC_ternary_oxide
Inorganic FF for binary halides	INORGANIC_binary_halide
Inorganic FF for glass	IORGANIC_glass
Inorganic FF for clay	INORGANIC_clay
Inorganic FF for zeolite	INORGANIC_zeolite
Zeolite Hill-Sauer	INORGANIC_zeolite_HS

Non-specific, miscellaneous FF	MISC_FF
Multiple potential	multiple

References and notes about these FF schemes can be found in Section 1.3.

Note that OPLS_2005 and OPLS-AA essentially share the same potential functional forms. In the following description, anything refers to OPLS-AA also applies to OPLS_2005. However, OPLSAA is essentially decrepit within DL_FIELD since the introduction of the DL_F Notation.

In general, there are three ways to map a molecular model to a potential scheme: (1) template-base (MOLECULE template) and (2) topological analysis using DL_F Notation. Examples of template-based FF including CHARMM, AMBER, Gromos and some of the OPLS FF components, such as OPLS_CL_P. Examples of topological-based FF including OPLS2005, CVFF and PCFF. (3) Combination of (1) and (2).

Depending on the FF schemes and system models, one file formats would be more suited than the other. For instance, the use of PDB file format would be required for system models contain proteins. For more information, please see Section 6.

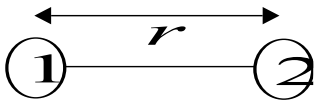
The MISC_FF is the most flexible, generic scheme and does not confine to any FF models. It is useful to define MOLECULEs and parameters that do not fall into any standard FF schemes. It allows definitions of more than one version of identical MOLECULE templates with different sets of FF parameters. For more information, consult the remarks described in the library files, *DLPOLY_MISC.sf* and *DLPOLY_MISC.par*.

*Prior to version 4.1, all CHARMM force fields are collectively included in a single *.sf* file and a single *.par* file. This 'general' set of CHARMM files will be removed in some future DL_FIELD versions once all individual CHARMM force field components are set up.

**Prior to version 4.1.1, all INORGANIC force fields are collectively included in a single *.sf* file and a single *.par* file. This 'general' set of INORGANIC files has now been removed. All individual force field components are migrated into appropriate library files classified according to the types of inorganic materials.

3.3 *BOND* and *END BOND* directive

Enclosed within the *BOND* directive is the definition of all two-body bond interactions. The bonds between two atoms are usually represented by a harmonic expression:



$$U(r) = k_b(r - b_o)^2$$

Where k_b is the spring constant and b_0 is the bond equilibrium distance.

Some potential schemes such as PCFF and DREIDING use different functional forms for bonds:

$$U(r) = k_1(r - b_o)^2 + k_2(r - b_0)^3 + k_3(r - b_0)^4$$

$$U(r) = D_e[1 - e^{-\alpha(r-b_0)}]^2$$

$$U(r) = \frac{1}{4}k(r^2 - b_0^2)^2$$

Quartic or expanded harmonic for PCFF type of consistent FF.

Morse potential for DREIDING

4th power harmonic for Gromos G54A7

In DREIDING force field, users are given the option to choose either harmonic or Morse potential for the bond interactions (Option **4** in the *control file*).

In CHARMM, the parameters for two-body Urey-Bradley (UB) terms, which involve 1-3 bonded atoms, are defined in the ***ANGLE*** directives in the *.par* file. However, UB parameters will be listed under the ***bond*** directive in the DL_POLY's *FIELD* file.

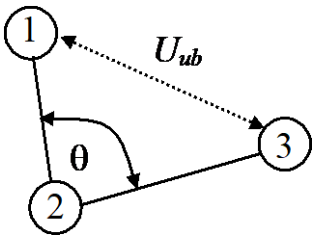
The 4th power harmonic form is not available in DL_POLY. For G54A7, the harmonic function is used, by scaling the force constant.

$$k_b = 2k^2 b_0^2$$

Where k_b is the force constant for the usual harmonic function.

3.4 *ANGLE* and *END ANGLE* directive

Enclosed within the *ANGLE* directive is the definition of three-body (1-3) angle interactions. In most cases, harmonic angle is being used, given by the equation.



$$U(\theta) = k_\theta(\theta - \theta_o)^2$$

$$U_{ub}(r) = k_{ub}(u - u_0)^2$$

Harmonic angle

(CHARMM only)

Where k_θ is the spring constant for the bond angle. For CHARMM, the two-body Urey-Bradley (UB) 1-3 interaction terms, U_{ub}, is also defined here where k_{ub} is the corresponding spring constant. If there is no Urey-Bradley term, then k_{ub} and u_0 must set to zero (0.0). For other potential schemes, k_{ub} and u_0 must be left out entirely.

Some potential schemes such as DREIDING and PCFF use different functional forms for angles:

$$U(\theta) = k_{\theta 1}(\theta - \theta_o)^2 + k_{\theta 2}(\theta - \theta_0)^3 + k_{\theta 3}(\theta - \theta_0)^4$$

$$U(\theta) = k_c(cos\theta - cos\theta_0)^2$$

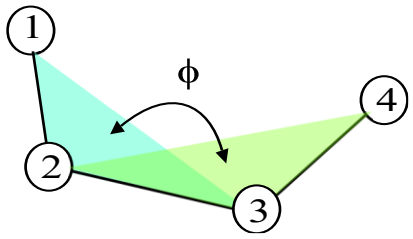
Quartic or expanded harmonic (PCFF)

Harmonic cosine in DREIDING and G54A7

When using DREIDING force field, users are given options to choose either harmonic or harmonic cosine functions for the angle interactions (Option **5** in the *control* file).

3.5 *DIHEDRAL* and *END DIHEDRAL* directives

Enclosed within the *DIHEDRAL* directive contains the four-body (1-4) dihedral interactions.



CHARMM, CVFF, G54A7 $U(\phi) = k_\phi[1+\cos(n\phi-\delta)\ ]$

AMBER $U(\phi) = \frac{k_\phi}{f}[1+\cos(n\phi-\delta)\ ]$

OPLS $U(\phi) = \frac{V_1}{2}[1+\cos(\phi)\ ]+\frac{V_2}{2}[1-\cos(2\phi)\ ]+\frac{V_3}{2}[1+\cos(3\phi)\ ]$

PCFF $U(\phi) = \sum_{n=1}^{3} V_n[1-\cos(n\phi-\delta)]$

DREIDING $U(\phi) = V[1-\cos(n(\phi-\delta))]$

Where k_ϕ and V are dihedral constants, n is the periodicity and δ is the phase shift in unit angle degree. In AMBER, the quantity f is the scaling factor to the torsion barrier.

If an atom set contains more than one dihedral term then the multiplicity value, n, should be set to a negative number $-n$ with the last term set as the normal positive value n. Each dihedral term must be defined in separate line of statements in the parameter file. The negative value of n is just an indication for DL_FIELD to ensure the 1-4 interaction scale factors (see below) are only applied once for a particular dihedral atom set with more than one dihedral terms.

In addition to the cosine dihedral shown above, the Ryckaert-Belleman dihedral is also implemented for the G54A7 force field:

$$U(\phi) = A[a+b\cos(\phi)+c\cos^2(\phi)+d\cos^3(\phi)+e\cos^4(\phi)+f\cos^5(\phi)]$$

Where the prefactor *A* is set to 1.0.

3.6 The 1-4 scaling factors

Quite often, the 1-4 electrostatic scaling factor, $\varepsilon_{1\text{-}4}$, and the 1-4 van der Waal's scaling factor, $v_{1\text{-}4}$, are also included and these values can be different according to the type of potential schemes.

DL_FIELD will set these scaling factors to appropriate values according to the potential scheme in the ***dihedral*** directive in the DL_POLY's *FIELD* file.

For GROMACS, the 1-4 scaling factors are included in the [pairs] GROMACS directive in the itp files.

For CHARMM, $\varepsilon_{1\text{-}4}$ =1.0 and $v_{1\text{-}4}$ = 1.0. However, DL_FIELD will define $\varepsilon_{1\text{-}4}$ =1.0 and $v_{1\text{-}4}$ = 0.0 in the FIELD file since all van der Waals 1-4 interactions will be explicitly defined in the ***bond*** directive of the *dl_poly.FIELD* file (see ***VDW*** directive below for further details).

For AMBER, $\varepsilon_{1\text{-}4}$ =1.0/1.2 and $v_{1\text{-}4}$ = 1.0/2.0 by default. However, Glycam models use $\varepsilon_{1\text{-}4}$ = $v_{1\text{-}4}$ = 1.0 by default. The variation of scaling factors will be automatically taken care of by DL_FIELD. If proteins were also present, then the values of 1.0/1.2 and 1.0/2.0 should be used instead. To change the scaling factors, the Glycam's default values can be overridden by user-defined values define in the *dl_field.control* file. All van der Waals 1-4 interactions share the same potential parameter sets for the non-bonded van der Waals interactions.

For OPLS force fields, $\varepsilon_{1\text{-}4}$ =1.0/2.0 and $v_{1\text{-}4}$ = 1.0/2.0. All van der Waals 1-4 interactions share the same potential parameter sets for the non-bonded van der Waals interactions.

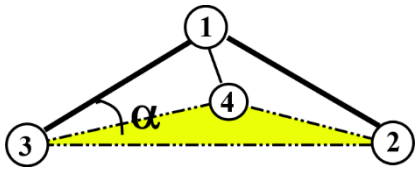
For PCFF, CVFF and DREIDING$\varepsilon_{1\text{-}4}$ =1.0 and $v_{1\text{-}4}$ = 1.0. All van der Waals 1-4 interactions share the same potential parameter sets for the non-bonded van der Waals interactions.

For G54A7, $\varepsilon_{1\text{-}4}$ =1.0 and $v_{1\text{-}4}$ = 1.0, However, DL_FIELD will define $\varepsilon_{1\text{-}4}$ =1.0 and $v_{1\text{-}4}$ = 0.0 in the *dl_poly.FIELD* file since all van der Waals 1-4 interactions will be explicitly defined in the ***bond*** directive in the *dl_poly.FIELD* file.

For TraPPE_EH force fields – there is no 1-4 interaction and $\varepsilon_{1\text{-}4}$ = $v_{1\text{-}4}$ = 0.

3.7 *IMPROPER* and *END IMPROPER* directives

Enclosed within these directives contains all improper terms. These terms are required for molecules that contain certain molecular functional group in order to preserve a particular geometry of three atoms around a central atom. Examples of these functional groups are carbonate and carbonyls.



The improper geometry is preserved by the following expressions:

CHARMM, G54A7: $$U(\alpha) = k_\alpha (\alpha - \alpha_0)^2$$

AMBER: $$U(\alpha) = k_\alpha (1 + \cos(n\alpha - \gamma))$$

OPLS: $$U(\alpha) = \frac{V_1}{2}[1 + \cos(\alpha)] + \frac{V_2}{2}[1 - \cos(2\alpha)] + \frac{V_3}{2}[1 + \cos(3\alpha)]$$

DREIDING and PCFF: similar to CHARMM. CVFF: similar to AMBER.

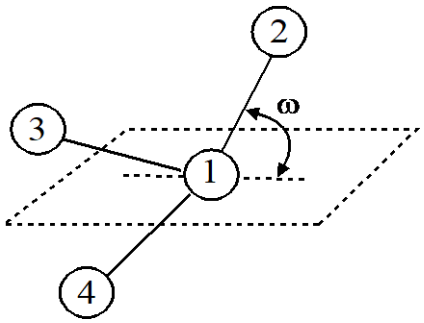
Where α is the angle of vector 1-3 relative to the plane consists of atom 2, 3 and 4 and k_α is the corresponding force constant. For AMBER, n is the periodicity and γ is the phase shift angle. Note that some potential schemes such as DREIDING and PCFF do not use improper torsion to preserve the structure. Instead, these potential schemes use inversion.

However, DL_FIELD does allow improper terms to be described for these potentials and the improper interaction definition would be the same as that of CHARMM.

DL_FIELD will search for any improper interactions if it is defined in the structure file and will list under the ***dihedral*** directive in the *dl_poly.FIELD* file.

3.8 *INVERSION* and *END INVERSION* directives

Enclosed within these directives contain all inversion terms. These terms are required for molecules that contain certain molecular functional group to preserve a particular geometry of three atoms around a central atom. Examples are nitrogen atom surrounded by three hydrogen atoms in the ammonia molecule to form a trigonal pyramid structure. The hydrogen can flip like an inverting umbrella to form another structure. The flipping can be prevented by applying an inversion term.



PCFF: $U(\omega) = k_i(\omega - \omega_0)^2$

DREIDING: $U(\omega) = k_i[1 - \cos(\omega)\ ]^2$

CHARMM, AMBER, CVFF, OPLS: similar to PCFF.

Note that the inversion angle potential is a sum of the three possible inversion angle terms. The example above shows one of the possible terms with the inversion angle w due to vector 1-2 relative to the plane consists of atoms 1,3 and 4.

Some potential schemes such as CHARMM, AMBER and OPLSAA do not use inversion torsion to preserve the structure. Instead, these potential schemes use improper functions. However, DL_FIELD does allow inversion terms to be described for these potentials and the inversion definition would be the same as that of PCFF (harmonic function).

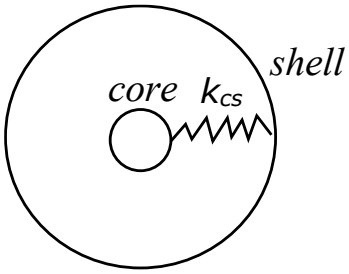
DL_FIELD will search for any inversion interactions if it is defined in the structure file and will list under the ***inversion*** directive in the *dl_poly.FIELD* file.

3.9 *SHELL* and *END SHELL* directives

Enclosed within these directives list the spring constants for all core-shell terms. The core-shell model is usually employed in inorganic materials to simulate polarisability effect of an atom such as oxygen. However, DL_FIELD also allows core-shell model to be included in any organic force field schemes. For more details, please see Section 11.2.
Each core-shell set will have the following format:

ATOM_KEY (or ATOM_LABEL) *for core* ATOM_KEY *for shell* *spring constant* *Remark*

For the INORGANIC force field, ATOM_KEYs must be used for both core and shell components. For the ORGANIC force field, an ATOM_LABEL can be used for the core component and the shell component must always be defined as the self-CONNECT type.



Schematic diagram showing core-shell component (not to scale) of an atom.

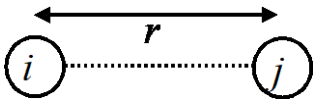
DL_FIELD considers collectively the core and shell components as a single entity of an atom. Both components are connected to a harmonic spring with a constant k_{cs}, to mimic the polarisability effect of an atom.

Usually, the net charge of a core-shell atom consists of partial charges from the core and shell components, respectively. For Coulombic calculations, all possible combinations of core and shell charges will be considered, except for the core and the shell belong to a same atom. For vdw calculations, core-core interactions are ignored (assume to be zero) since the 'chemistry' should only occur at shell-component of an atom.

A large proportion of the mass of a core-shell atom is assigned to the core-component. The mass of a shell can be either zero (static shell) or a fraction of the total mass to permit a dynamic description (adiabatic shell). For more information about the types of shells, please consult the DL_POLY user manual.

3.10 *VDW* and *END VDW* directives

Enclosed within these directives define the non-bonded Van-der-Waals (vdw) interactions between atom *i* and *j.* Except for PCFF, all potential schemes use the Lennard-Jones 12-6 function.



$$V(r_{ij}) = \epsilon\left[\left(\frac{R}{r_{ij}}\right)^{12} - 2\left(\frac{R}{r_{ij}}\right)^{6}\right]$$

Where ε and R are the energy and equilibrium minimum distance terms between two atoms of type i and j. If the atoms are of different type ($i \neq j$), then the following mixing rules would be applied, depending on the FF schemes:

For CHARMM and AMBER FF, ***half*** equilibrium distances are expressed in the data files.

$$R = R_i + R_j \qquad \varepsilon = \sqrt{\varepsilon_i \varepsilon_j} \quad \text{(For CHARMM and AMBER)}$$

For DREIDING, R is shown as the equilibrium minimum distance of diatomic *i*-*i*. The mixing rules for atom *i*-*j* of different types are therefore given as:

$$R = \frac{R_i + R_j}{2} \qquad \varepsilon = \sqrt{\varepsilon_i \varepsilon_j} \quad \text{(For DREIDING)}$$

In CHARMM, CHARMM19 and G54A7, in addition to intermolecular vdw interactions, some atom pairs can have other sets of Lennard-Jones parameters for the *intramolecular* 1-4 interactions. While the intermolecular vdw interactions will be defined under the ***vdw*** directive in the *dl_poly.FIELD* file, all intra 1-4 vdw interactions will be defined under the ***bond*** directive in the *dl_poly.FIELD* file as excluded atom lists. If the vdw 1-4 parameters are not explicitly defined (set to 0.0 in the *.par* file), then the intermolecular vdw parameters will be used instead.

In OPLS, vdw interactions are usually described in the equivalent 12-6 form of

$$V(r_{ij}) = \varepsilon\left[\left(\frac{\sigma}{r_{ij}}\right)^{12} - \left(\frac{\sigma}{r_{ij}}\right)^{6}\right] \qquad \text{(OPLS)}$$

Where σ is the steric parameter and geometric rules are used to obtain vdw parameters between atom *i* and *j*:

$$\sigma = \sqrt{\sigma_i \sigma_j} \qquad \varepsilon = \sqrt{\varepsilon_i \varepsilon_j} \qquad \text{(OPLS and CVFF)}$$

In the case of PCFF and COMPASS, the Lennard-Jones 9-3 form is used:

$$V(r_{ij}) = \varepsilon\left[2\left(\frac{R}{r_{ij}}\right)^{9} - 3\left(\frac{R}{r_{ij}}\right)^{6}\right] \text{ (PCFF, COMPASS)}$$

and use the following mixing rules (6th order combination rule) for atom *i-j* interaction:

$$R = \left[\frac{\left(R_i^6 + R_j^6\right)}{2}\right]^{\frac{1}{6}} \qquad \varepsilon = 2\sqrt{\varepsilon_i \varepsilon_j}\left(\frac{R_i^3 R_j^3}{R_i^6 + R_j^6}\right)$$

For G54A7, an equivalent, simplified LJ 12-6 version is used:

$$V(r_{ij}) = \frac{A_{ij}}{r_{ij}^{12}} - \frac{B_{ij}}{r_{ij}^{6}} \qquad \text{(G54A7)}$$

with geometric mixing rules to obtain parameters for the atom *i-j* interactions:

$$A_{ij} = \sqrt{A_i A_j} \qquad B_{ij} = \sqrt{B_i B_j} \qquad \text{(G54A7)}$$

When the option to calculate energy is switched on, no VDW energy will be calculated for any atom pair distance greater than the cutoff value specified. There is no long-range correction and therefore the energy values obtained are for references only.

The quantity $v_{1\text{-}4}$ is the 1-4 intra-vdw scaling factor. In CHARMM, CHARMM19, PCFF, CVFF, G54A7 and DREIDING, this is 1.0. For AMBER and OPLS, this is 1.0/2.0. However, AMBER's Glycam model sets $v_{1\text{-}4}$ = 1.0. Note that, for some force fields such as CHARMM and G54A7, the $v_{1\text{-}4}$ value is switched to 0.0. This is because, the 1-4 vdw pairs are explicitly defined, which can have different sets of vdw parameters than those of non-bonded ones. For TraPPE_EH and TraPPE_UA, $v_{1\text{-}4}$ = 0.0.

3.10.1 Lennard-Jones Mixing Rules in Multiple Potential Systems

If more than one force field schemes are specified (the multiple potentials scheme), user can specify the mixing equation scheme independently to derive the ε and σ for atom pairs *i* and *j* based on some type of equations expressed in terms of the vdw parameters for pure components *ii* and *jj.*

The available vdw mixing schemes are shown below:

Mixing rule no	Scheme name	Energy, ε_{ij}	Steric, σ_{ij}
1	Standard geometric	$\sqrt{\varepsilon_{ii}\varepsilon_{jj}}$	$\sqrt{\sigma_{ii}\sigma_{jj}}$
2	Standard arithmatic	$\frac{\varepsilon_{ii}+\varepsilon_{jj}}{2}$	$\frac{\sigma_{ii}+\sigma_{jj}}{2}$
3	Fender-Halsey	$\frac{2\epsilon_{ii}\epsilon_{jj}}{\epsilon_{ii}+\epsilon_{jj}}$	$\frac{\sigma_{ii}+\sigma_{jj}}{2}$
4	Halgren HHG	$4\frac{\varepsilon_{ii}\varepsilon_{jj}}{(\sqrt{\varepsilon_{ii}}+\sqrt{\varepsilon_{jj}})^2}$	$\frac{\sigma_{ii}^3+\sigma_{jj}^3}{\sigma_{ii}^2+\sigma_{jj}^2}$
5	Waldman-Hagler	$2\sqrt{\varepsilon_{ii}\varepsilon_{jj}}\frac{(\sigma_{ii}\sigma_{jj})^3}{\sigma_{ii}^6+\sigma_{jj}^6}$	$\left(\frac{\sigma_{ii}^6+\sigma_{jj}^6}{2}\right)^{\frac{1}{6}}$
6	Slater-Kirkwood (see below)	$B_{ij}=\frac{3}{2}\frac{\alpha_i\alpha_j}{\sqrt{\frac{\alpha_i}{N_i}+\frac{\alpha_j}{N_j}}}$	$A_{ij}=\frac{1}{2}B_{ij}(w_i+w_j)^6$

Lorentz-Berthelot – specify 1 for energy, 2 for steric

Hogervorst (good hope) – specify 1 for energy, 1 for steric

3.10.2 Slater-Kirkwood (SK) combination

Note: SK combination rule only applicable for multiple potential system and Option **12** (ATOM_KEY display format) in DL_FIELD control file must select to display DL_FIELD format (DL_F Notation). More specifically, it can be used to determine the vdw parameters between atom pairs from two different force field schemes.

The SK combination rule is particularly useful if there is no vdw parameter available in the library. The SK approach uses atomic data such as the atomic polarizability, α and electron number, *N*, to derive the dispersive parameter, *B*. Consider two atoms of type *i* and *j*, by using the Slater-Kirkwood approximation, the dispersive parameter for the atom pair is given as:

$$B_{ij} = \frac{3}{2}\frac{\alpha_i \alpha_j}{\sqrt{\frac{\alpha_i}{N_i}} + \sqrt{\frac{\alpha_j}{N_j}}}$$

Although *N* can simply be the number of valence electrons of the outer shells of atoms, which is the dominant contributions to the dispersive parameters, this choice is known to overestimate *B*. Kiselev *et. al.* [1] treated *N* as the complete numbers of electron. If atomic dispersive terms, B_{ii} and B_{jj} are known then the SK combination can be rewritten as

$$B_{ij} = \frac{2\alpha_i \alpha_j B_{ii} B_{jj}}{\alpha_i^2 B_{ii} + \alpha_j^2 B_{jj}}$$

where N_i and N_j are defined as the *effective* number of electrons [2].

$$N_i = \frac{16 B_{ii}^2}{9\alpha_i^3}$$

The repulsive parameter, *A*, can be obtained from the sum of atomic radii, *w*. For a given atom pair *i* and *j*,

$$A_{ij} = \frac{1}{2} B_{ij} \left(w_i + w_j\right)^6$$

To minimise the use of variable parameters and maximise the use of standard atomic properties, DL_FIELD will adopt the Kiselev formalism. These leaves α and *w* as the only adjustable parameters. DL_FIELD will search for the data included in the *dl_field.atom_data* file, which contained the atomic polarisabilities, electron numbers and atomic radii data, to determine the corresponding Lennard-Jones parameters.

Most of the atomic polarizability data were extracted from the CRC Handbook [3]. For atoms within a molecule, the polarisabilities of the individual atomic constituents are less well defined. Several sources will be used with the following assumptions taken into consideration [2]: (1) the polarisabilities of atomic constituents are additive to the molecular polarisability. (2) atoms in different chemical environment can share the same polarizability value.

In the case of the atomic radii, it is assumed to be independent of the types of molecules that contain this atom. For atoms and atoms in molecules, the van der Waals radii will be used. For ionic species, the bare ionic radii will be used.

To decide the values of α and *w*, DL_FIELD will first look for exact match of ATOM_KEYs (expressed in DL_F Notation) against the atom list in the library file. If there is no match, DL_FIELD will look for lists of default entries and, finally, a generic element (X*) that matches with the ATOM_KEY. For example, if there is no exact match for, say, a nitrogen atom within a molecule, the generic value for N* will be used. The values of α and *w* that were chosen for setting up vdw atom pairs will be shown in the *dl_field.output* file. The *dl_field.atom_data* file also contains comments, quoting the sources for each entry.

If any ATOM_KEYs cannot be matched with the data in the library, the vdw parameters involve the ATOM_KEYs in question will be set to zero.

Finally, please note that data listed in *dl_field.atom_data* library file cannot be overridden by the udff file. Users would need to change values or add new entries directly in the library file.

References
[1] A.V. Kiselev, A.A. Lopatkin and A.A. Shulga, *Zeolites* **5**, 261 (1985).
[2] T.A. Halgren, *J. Am. Chem. Soc.* **114**, 7827 (1992).
[3] CRC Handbook for Chemistry and Physics, 95th Edition (2014-2015).

3.11 *VDW_FIX* and *END VDW_FIX* directives

Enclosed within these directives are lists of atom pairs with their unique sets of vdw parameters. Usually, and many times by default, the vdw parameters of atom pairs are obtained automatically from some mixing rules (such as the Benoit rules). However, the vdw functional form, as well as the parameters of a pair of atoms, can be changed by using the ***VDW_FIX*** directive.

The syntax for using the directive are as follows:

VDW_FIX *potential_label*
atom_1 *atom_2* func1 p1 p2 p3
atom_3 *atom_4* func2 p4 p5 p6
…
END VDW_FIX

Where *atom_1*, *atom_2* and *atom_3*, *atom_4* are two sets of atom pairs of which the vdw parameters will be redefined. These atoms are the ATOM_KEYs expressed in the standard symbols of a force field scheme (not in DL_F Notation), shown as ATOM_KEYs in the *sf* files.

The value 'func' is the vdw functional form, which can be either the 12-6 Lennard-Jones, Morse or Buckingham (see the Inorganic section for the available functional forms). The values p1, p2, p3 and p4, p5, p6 are the parameter values of the functional forms.

The *potential_label* is the name of a potential scheme. Usually, this would be the same where the ***VDW_FIX*** directive is located in the *.par* library file. For example, to override the vdw parameters for the pair of atoms, *atom_1* and *atom_2* in the AMBER force field, the ***VDW_FIX*** directive is issued in the *.par* file, or the
udff file as follows:

```
POTENTIAL amber
UNIT kj/mol
…
…
VDW_FIX amber
C1  HO    lj    5.00   10.000
…
…
END VDW_FIX
```

When AMBER force field is used and assuming the system model contains the ATOM_KEYs C1 and OH, then instead of using the usual mixing rule (for instance, geometric rule for the ε parameter) to obtain the vdw parameters for the C1-OH pair, the parameters 5.0 and 10.0 will be chosen instead, and expressed them as the lj functional form in the *dl_poly.FIELD* file.

The ***VDW_FIX*** directive can also be used for models using multiple-potential force field schemes. Example below shows the use of the directive for a bio-inorganic model.

```
POTENTIAL inorganic_binary_oxide
UNIT eV
…
…
VDW_FIX charmm22_prot
C1   HO      lj     5.00    10.000
…
…
END VDW_FIX
…
…
END POTENTIAL
```

Here, the ***VDW_FIX*** is in the ***POTENTIAL*** inorganic_binary_oxide force field, with the potential label charmm22_prot. This means that for each atom pair that is listed within the ***VDW_FIX*** directive, one atom must belong to the inorganic_binary_oxide force field and the other atom must belong to the charmm22_prot force field.

A more generic potential label can also be used. For example, in the case of CHARMM force field, the force field is split into various components (protein, carbohydrates, etc). Example below expresses the potential label as charmm*, where the asterisk indicates any potential scheme begins with the label 'charmm'.

```
POTENTIAL charmm22_prot
UNIT kcal/mol
…
…
VDW_FIX charmm*
SOD  OC     buck    124.00    0.34     2.00
CT3  HA     mors    111.00    0.01     3.00
…
END VDW_FIX
```

The above example shows the use of the ***VDW_FIX*** in the multiple potential schemes, which is in fact the use of more than one CHARMM force field components in the system. One of the components must be the charmm22_prot, where the ***VDW_FIX*** directive is located. The label 'charmm*' means that DL_FIELD will look for any matching atom pairs if the second force field is of other CHARMM force field components.

Note that atom equivalence does not apply to the atoms listed in ***VDW_FIX***. In other words, DL_FIELD only matches exactly the atoms as shown within the ***VDW_FIX*** directives.

3.12 Coulombic Energy

The long-range Coulombic energy takes the charge parameters on the atoms. The charges are not listed in *.par* files. Atomic charge values are usually defined in the MOLECULE templates in *sf* files.

In the case of PCFF, CVFF, zeolite_HS and OPLS_2005, if the charge is not defined in the MOLECULE definition, then charges will be obtained from the special bond increments files (bci) listed in the *lib/* directory. DL_FIELD will determine the total charge on an atom by summing the partial charge contributions from all neighbouring atoms that are directly linked to the atom.

The Coulombic energy takes the following functional form:

$$U(r_{ij}) = k_e \frac{q_i q_j}{|r_{ij}|} \varepsilon_{1-4} \qquad k_e = \frac{1}{4\pi\ \varepsilon_0}$$

Where q_i and q_j are the charges of the ATOM *i* and ATOM *j*. The quantity k_e is the Coulombic constant and the exact value is dependent on value definitions for π and the electric constant ε_0.

The quantity $\varepsilon_{1\text{-}4}$ is the 1-4 scaling factor which is included for intra 1-4 Coulombic interactions. In CHARMM, PCFF and DREIDING, this is 1.0. For CHARMM19, this is 0.4. For AMBER, this is 1/1.2. For OPLSAA, this is 1/2.0. However, the AMBER's Glycam model sets $\varepsilon_{1\text{-}4}$ = 1.0. For TraPPE_EH and TraPPE_UA, this is set to 0.0.

3.13 *THREE-BODY* and *END THREE-BODY* directives

The three-body interactions are different from the three-body angular (see Section 3.4) interactions, where the latter describes the bonded 1-3 interactions. For three-body interactions, the atoms are not bonded.

These types of interactions are usually used in inorganic materials such as zeolites and glassy materials. An example is shown in the *DLPOLY_INORGANIC_glass.par* file.

3.14 *EQUIVALENCE* and *END EQUIVALENCE* directives

Enclosed within these directives contain lists of definitions of equivalent ATOM_KEYs for a given potential scheme. This feature is especially useful if one were to define a new ATOM_KEY that shares some common behaviour with the other ATOM_KEYs. Quite often, the new ATOM_KEY may share similar potential parameters of other ATOM_KEYs that are already defined in the parameter file. Instead of redefining a whole new set of parameters for the new ATOM_KEY, an ***EQUIVALENCE*** directive can be issued for the new ATOM_KEY. This directive informs DL_FIELD that, instead of looking for the new ATOM_KEY, the equivalent ATOM_KEY*s* should be used, when searching for the potential parameters that correspond to interactions that contain the new ATOM_KEY.

Different equivalent ATOM_KEYs can be assigned for different types of interactions using *interaction components*. The various *interaction components* are shown in the table as follows:

Interaction type	***Interaction components***
Bond interactions	***bond_***
Angle interaction	***angle_***
Dihedral interactions	***dihedral_***
Inversion	***inv_***
Improper	***imp_***
Shell	***shell_***
Van-der-Waals	***vdw_***
Three-body interactions	***tbp_***

The format to define an equivalent atom is as follows:

new_atom_key > ***component1_***ATOM_KEY*1* ***component2_***ATOM_KEY*2* ...

For instance, in DREIDING force field, a generic sp3 hybridised oxygen atom is defined with an ATOM_KEY *O_3* (see *DLPOLY_DREIDING.sf* file). Suppose a user wish to define a new oxygen atom exclusively for molecules containing the -OH alcohol group and assign a new ATOM_KEY *OA*. The new key will share the same potential interactions as that of *O_3,* except the vdw interaction which has a different set of parameters.

The new atom can be defined in the user-define force field (*udff*) file as follows:

```
POTENTIAL DREIDING

ATOM_TYPE
...
...
O_alcohol    OA       15.99994       New atom for alcohol -OH only
...
...
END ATOM_TYPE
```

Instead of duplicating all interactions involving *O_3* for *OA* in the file, the following ***EQUIVALENCE*** directive can be issued in a *udff* file:

```
EQUIVALENCE
...
...
OA > bond_O_3  angle_O_3  dihedral_O_3  inv_O_3
...
...
END EQUIVALENCE


VDW
...
OA     3.5000          0.0957    * New vdw value for OA.
...
END VDW
```

The ***EQUIVALENCE*** statement shown in the example:

OA > bond_O_3 angle_O_3 dihedral_O_3 inv_O_3

means any bond, angle, dihedral and inversion components that contain *OA* will be treated as *O_3*.

For instance, the angle *H-OA-C2* is equivalent to *H-O_3-C2,* DL_FIELD will search for the potential parameters correspond to the latter set of angles. Since no equivalence is set for the vdw interactions, DL_FIELD will therefore look for vdw parameters correspond to *OA*.

DL_FIELD also allows multiple equivalence definitions such as shown below:

OA > bond_O_3 angle_C_2 ...

Which means the *OA atom key* is equivalent to *O_3* for bond interactions but is equivalent to *C_2* for angle interactions. For instance, the bond *OA - H* is equivalent to *O_3 - H*, while the angle *H - OA - C_2* is equivalent to *H - C_2 - C_2*. All other interactions (such as vdw and dihedral) involving *OA*, must be explicitly defined.

Once an equivalent ATOM_KEY is defined for a given interaction, DL_FIELD will look for data containing the equivalent ATOM_KEY, ignoring potential parameters involving the original ATOM_KEY even if they were included in the file.

Note that, the equivalence mentioned so far is also called the *first-tier* equivalence setup. During the conversion processes, DL_FIELD will look for potential parameters based on the ATOM_KEYs with the equivalence atom if applicable. If the program fails to locate the parameters, then DL_FIELD will determine the second set of equivalence atoms, if applicable, and using the modified set of atoms to look for the parameters the second time. This is shown below.

3.15 Second-tier Equivalence

This essentially work the same way as that of ***EQUIVALENCE*** directive mentioned above and allows user to define two different sets of equivalence atoms. When DL_FIELD fails to find a suitable parameter sets during the first search attempt, it will make a second attempt by using an alternative (second) set of ***EQUIVALENCE*** atoms, if this is defined. DL_FIELD will only then report an error if the second attempt also fails to locate a suitable set of parameters. If DL_FIELD able to locate the parameters in the first attempt, then the second-tier equivalence set will be ignored.

The second-tier equivalence set must be defined within the ***EQUIVALENCE*** and ***END EQUIVALENCE*** directives and distinguish from the *first-tier* equivalence set by having the '2' suffix on the interaction components.

Example uses of second-tier equivalence can be found in the OPLS2005 parameter file. For example, consider the equivalence statements below, showing several possible equivalence scenarios:

```
EQUIVALENCE
CDX    > dihedral_CD
CML    > angle_CM imp_CM vdw_CM angle2_CM imp2_CM
C5BB   > angle2_CA dihedral2_CA imp2_CA
CTNC   > bond_CT angle_CT  imp_CT  vdw_CT  bond2_CT angle2_CT imp2_CT dihedral2_CT
END EQUIVALENCE
```

First, the *dihedral* component for ATOM_KEY CDX is made equivalent to CD and there is no second-tier equivalence being defined. This means that, during the first search attempt, DL_FIELD will look for the equivalent dihedral parameter sets involving CD, instead of CDX. If this fails, DL_FIELD will make a second attempt, looking for the dihedral parameters involving the CDX ATOM_KEY itself.

In the second equivalence statement, the CML ATOM_KEY contains the first-tier equivalence to CM ATOM_KEY for the *angle*, *imp* and *vdw* components. The statement also defines the second-tier equivalence to CM ATOM_KEY, but only for the *angle* and *imp* components. This means that, when DL_FIELD make the second search attempts, the equivalence only applies to *angle* and *imp*, but not the *vdw*.

The third equivalence statement only defines second-tier equivalence set for C5BB ATOM_KEY. This means that, during the first search attempt, DL_FIELD will look for interaction components involving the original C5BB key itself. When this fails, DL_FIELD will only then use the equivalent atom CA in the second attempt, for *angle*, *dihedral* and *imp* components.

The fourth equivalence statement stipulates that CTNC is made equivalent to CT in both the first and second search attempts. The exception being the *dihedral* component, which set equivalence to CT only in the second search attempt.

4 Control File

The *control* file is read by DL_FIELD to determine how FF output files would be defined. All the available control settings and the number of options available for each setting are shown in the control file.

```
This is the title line. Put what you want. Reads only 80 columns
A 1                 * Construct DL_POLY output file.
B none              * Additional output files.
C 0                 * Write user udff model file.
1  opls2005          * Type of force field require.
2 kcal/mol           * Energy unit: kcal/mol, kJ/mol, eV or default
3  normal            * Conversion criteria (strict, normal, loose)
4  1                 * Bond type (0=default, 1=harmonic , 2=Morse)
5  2                 * Angle type (0=default, 1=harmonic, 2=harmonic cos)
6  none              * Include user-defined information. Put 'none' or a .udff filename
7  0                 * Verbosity mode: 1 = on, 0 = off
8  example.pdb       * Filename for the user's atomic configuration.
9  None              * Output file in PDB. Put 'none' if not needed.
10 0  1.05 g/cm^3  1.9  * Solution Maker: on/off, density, unit, cutoff)
11 0                 * Optimise FIELD output size, if possible? 1=yes  2=no
12 2                  * Atom display: 1 = DL_FIELD format. 2 = Standard format
13 1                 * Vdw display format: 1 = 12-6 format   2 = LJ format
14 Default           * Epsilon mixing rule (organic FF only) : 1 = geometric   2 = arithmatic or default
15 Default           * Sigma mixing rule (organic FF only) : 1 = geometric   2 = arithmatic or default
16 1                 * Epsilon mixing rule (inorganic FF only) : 1 = geometric   2 = arithmatic
17 2                 * Sigma mixing rule (inorganic FF only) : 1 = geometric   2 = arithmatic
18 1                 * Epsilon mixing rule (between different FF) : 1 = geometric   2 = arithmatic or default
19 1                 * Sigma mixing rule (between different FF) : 1 = geometric   2 = arithmatic or default
20 1                 * Display additional info. for protein 1=Yes  0=No
21 1                 * Freeze atoms? 1 = Yes (see below)  0 = No
22 0                 * Tether atoms? 1 = Yes (see below)  0 = No
23 0                 * Constrain bonds? 1 = Yes (see below) 0 = No
24 0                 * Apply rigid body? 1 = Yes (see below) 0 = No
25 auto              * Periodic condition ? 0=no, other number = type of box (see below)
26 50.00  0.0  0.0   * Cell vector a (x, y, z)
   0.00  50.0  0.0   * Cell vector b (x, y, z)
   0.00  0.0  50.0   * Cell vector c (x, y, z)
27 default           * 1-4 scaling for coulombic (put default or x for scaling=x)
28 default           * 1-4 scaling for vdw (put default or x for scaling=x)
29 0   300          * Include velocity? 1=yes, 0=no and scaling temperature.
30 1                 * Position solute at origin 1 = yes, 0 = no
31 tip3p_e  1.7 default  *  Solvate model? none or specify solvent (see below) and distance criteria.
32 0  20.0           * Add counter ions? 1=yes, 0=no,  minimum distance from solute
33 0                  * Not use
34 10.0               * Not use
35 10.0               * Not use

#############################################################
FREEZE   A   all_backbone
FREEZE   B   all_backbone

TETHER       C   all_backbone  2390.0
CONSTRAIN  F   H-bond

RIGID     E

#############################################################
```

The order of the control settings is fixed and must not be removed nor rearranged. Do not remove the comments alongside the control settings.

The first line is the title line up to 80 columns in length. The numbers in **red** are the option numbers. They are shown only for illustration purposes in this manual and do not play any role in influencing the outcome of the results output.

Option **A** must always switch on. Option **B** is used to produce other files for third party software. This option takes the following values:

(a) *ChemShell*. This is for pyChemShell interface QM/MM setup. For more details, please consult ChemShell manual.
(b) *Gromacs*. This option instructs DL_FIELD to produce force field files for GROMACS simulation package. For more details, please consult Chapter 14.
(c) *Lammps*. This option instructs DL_FIELD to produce force field files for LAMMPS simulation package. For more details, please consult Chapter 15.
(d) *None*. This means no additional file would be produced, apart from DL_POLY files.

Option **C** instructs DL_FIELD to create a user *udff* model file, which contains all FF information of the system configuration. For more information, see Chapter 5:3.

The input options are described in more details as follows:

1. Force Field (FF) Scheme. This determines the conversion scheme and the corresponding type of potential parameters for the molecular model. The available choices are shown in the previous page.

2. Energy unit. This is the unit that will be used for the FIELD file. If this unit is different from what is defined in the *.par* file or the *udff* file of the potential scheme, DL_FIELD will do internal unit conversions and change the parameter values accordingly. The available options as as follows: *kcal/mol*, *kJ/mol*, *eV* and *K*.

3. Conversion criteria. DL_FIELD determines the bond connections of a molecule based on atomic distance criteria. A bond is identified if two atoms are within a certain minimum distance defined in the header file *dl_field.h*. User can control the extent of this distance criteria, where the 'strict' option means the minimum distance is reduced by -0.01 Å. The 'loose' option will increase the distance criteria by 0.01 Å. The 'normal' is the default option and the pre-set minimum distances set in the *dl_field.h* will remain unchanged.

4. Bond type. If DREIDING potential is selected, users must choose either harmonic bond (value = 1) or Morse bond (value = 2). All other potential schemes will automatically revert to value 0 (default bond type).

5. Angle type. If DREIDING potential is selected, users must choose either harmonic angle (value = 1) or harmonic cos angle (value = 2). All other potential schemes will automatically revert to value 0 (default angle type).

6. Include user-defined information. Once this is activated, DL_FIELD will read the file with the *.udff* extension specified by the user.

7. Verbosity mode. This setting determines the amount of information to be displayed in *dl_field.output* file. When this is switched on (1), DL_FIELD will display detailed conversion process, including the converted atom types.

8. Input filename for the user's atomic configuration. Currently, DL_FIELD only recognises the configuration file in strict PDB, *xyz* and mol2 formats.

9. Output file in PDB format. Atom sequence will be rearranged, and atom labels rewritten according to how they are defined in the MOLECULEs. In other words, the atom sequence will be the same as the *dl_poly.CONFIG* and *dl_poly.FIELD* files produced by DL_FIELD. The output file can be used as a PDB model template, along with the HISTORY files produced by the subsequent DL_POLY runs, in DL_ANALYSER.

If the output file is not needed, put 'none'. Otherwise, DL_FIELD will write the PDB structure in that name.

10. Solution Maker. Options to setup disordered system. See Chapter 8 for more details.

11. FIELD file optimisation (for DL_POLY only). This feature is useful for large molecular models that contain many similar molecules that have split into different Molecular Groups.

For example, water molecules are usually split into several Molecular Groups in PDB files due to large number of the molecules. This is so done because the maximum residue number permitted is 9999. Turning this feature on will enable DL_FIELD to lump all these water molecules and define under a single Molecular Group in the *dl_poly.FIELD* file.

12. ATOM_KEY display format. This setting allows user to select the display format for the ATOM_KEYs in FF files. This setting can take two options: DL_FIELD format or the standard format according to the type of potential scheme selected. For DL_FIELD format, ATOM_KEYs for most of the FF schemes can be expressed in DL_F Notation, which is obtained *in-situ* from the corresponding ATOM_TYPEs.

13. Vdw display format. Except for a few schemes such as PCFF and COMPASS, most potential schemes use 12-6 Lennard-Jones for the vdw interactions.

$$V(r_{ij}) = \epsilon\left[\left(\frac{R}{r_{ij}}\right)^{12} - 2\left(\frac{R}{r_{ij}}\right)^{6}\right]$$

Where R is the distance at which V is minimum and is related to the steric parameter, σ, as

$$R = \sqrt[6]{2}\sigma$$

In DL_POLY, the Lennard-Jones parameters can be displayed in two alternative formats: the 12-6 form and the 'LJ' form with the corresponding equation as shown below:

$$V(r_{ij}) = \frac{A}{r_{ij}^{12}} - \frac{B}{r_{ij}^{6}}$$

The 12-6 form

$$V(r_{ij}) = \varepsilon\left[\left(\frac{\sigma}{r_{ij}}\right)^{12} - \left(\frac{\sigma}{r_{ij}}\right)^{6}\right]$$

The LJ form

Therefore, the display format can be changed from one to the other using the following relationships:

$$A = 4\varepsilon\sigma^{12} \qquad B = 4\varepsilon\sigma^{6}$$

The A and B parameters will be shown if the vdw display option is set to 1. Otherwise, ε and σ are shown when the vdw display option is set to 2. This option does not apply to the FF schemes that do not use 12-6 Lennard-Jones.

Note that in certain older versions of DL_POLY, such as version 2.20 or before, will give an error message of 'unrecognised directives'. This may be due to older DL_POLY 2 versions not able to recognise the expression of 'LJ' form. If that is the case, use 12-6 form instead. Better still, please use DL_POLY_4 or later!

14. Apply to ORGANIC force fields only. Specify mixing rule for the parameter epsilon (ε, potential well-depth) for LJ12-6 between two atoms. This can be either geometric or arithmetic. The recommended setting is 'default' which means DL_FIELD will automatically set the correct rule depending on the force field scheme.

15. For ORGANIC force fields only. Specify mixing rule for the parameter sigma (σ, steric parameter) for LJ12-6 between two atoms. This can be either geometric or arithmetic. The recommended setting is 'default' which means DL_FIELD will automatically set the correct rule depending on the force field scheme.

16. For INORGANIC force fields only. Specify mixing rule for the parameter epsilon (ε, potential well-depth) for LJ12-6 between two atoms. This can be either geometric or arithmetic.

17. For INORGANIC force fields only. Specify mixing rule for the parameter sigma (σ, steric parameter) for LJ12-6 between two atoms. This can be either geometric or arithmetic.

18. Applicable to multiple potential systems only. Specify the mixing rule for the parameter epsilon (ε, potential well-depth) for LJ12-6 between two atoms from two different potential schemes. Usually, DL_FIELD will not determine the mixed parameters and leave blanks if the vdw atom pair involves one atom from an ORGANIC force field and the other from an INORGANIC force field. The vdw parameters for INORGANIC force fields are defined explicitly for a pair of atoms, *A*-*B*. However, if one of the atoms is a generic, *A*-*X*, (see *INORGANIC_clay.par* file) then DL_FIELD will carry out the calculation to derive the mixed parameter for the atom pair. Alternatively, ***VDW_FIX*** directive can be used to explicitly define both the functional forms and the vdw parameters for any pairs of atoms.

19. For multiple potential systems only. Specify the mixing rule for the parameter sigma (σ, steric parameter) for LJ12-6 between two atoms from two different potential schemes. Usually, DL_FIELD will not determine the mixed parameters and leave blanks if the vdw atom pair involves one atom from an ORGANIC force field and the other from an INORGANIC force field. The vdw parameters for INORGANIC force fields are defined explicitly for a pair of atoms, *A*-*B*. However, if one of the atoms is a generic, *A*-*X*, (see *INORGANIC_clay.par* file) then DL_FIELD will carry out the calculation to derive the mixed parameter for the atom pair. Alternatively, ***VDW_FIX*** directive can be used to explicitly define both the functional forms and the vdw parameters for a pair of atoms.

20. Display additional information (for DL_POLY FF files only). This setting allows the option to display additional information in a PDB file in *dl_poly.CONFIG* and *dl_poly.FIELD* files. Information that can display are the amino acid residue sequence obtained from the user's protein PDB file.

21. Freeze atoms switch. This option, when switched on (1), will read the [***FREEZE***] directives to decide which atoms to be frozen. This directive can appear anywhere in the *dl_field.control* file after the option settings.

The [***FREEZE***] directive takes the following data in the following order:

FREEZE *Molecular_Group* *atom_filter*

The *Molecular_Group* takes the label of the protein chain ID or protein molecular label. Only one [***FREEZE***] directive is allowed for each Molecular Group.

The *atom_filter* key determines which atoms to be frozen within a Molecular Group. The available keys are shown in the table below.

Atom_filter key	**Selection effects**
All_backbone	All backbone atoms of amino acids (α-carbon, amide carbon and nitrogen
c-alpha	(α-carbon on in amino acids)
all	All atoms in a Molecular Group
User-specified atom key	only affect a particular type of atoms

Note that there is no GROMACS equivalent for [***FREEZE***]. However, the [***TETHER***] directive can be used to produce a 'freeze' system, as described below.

22. Tether atoms switch. This is equivalent to *position_restraints* in the GROMACS context. This option, when switched on (1), will read [***TETHER***] directives to decide which atoms to be tethered. The current version of DL_FIELD only allows harmonic as the constraint potential form for a tethered atom. This directive can appear anywhere in the *dl_field.control* file after the option settings.

The [***TETHER***] directive takes the following data in the following order:

TETHER *Molecular_Group* *atom_filter* *spring_constant*

In a PDB file, the *Molecular_Group* takes the label of the protein chain ID or protein molecular label. In an xyz file, the Molecular_Group is defined as a directive. Only one [***TETHER***] directive is allowed for each Molecular Group.

The *atom_filter* key determines which atoms to be tethered within a Molecular Group. The available keys are shown in the table above. The *spring_constant* takes a value with the same unit as defined in the potential energy scheme.

Note that the *atom_filter* keys **all_backbone** and **c_alpha** are only applicable to molecules containing amino acids such as protein molecules. Molecules that do not contain amino acid will not be affected even if it contains similar types of atoms.

While DL_FIELD allows both [***FREEZE***] and [***TETHER***] directives to be defined at the same time for a given molecular group, it is up to the user to ensure both directives do not apply to same type of atoms. The example *control* file switched on the setting for frozen atom (Option **21**) and off for tethered atom. Consequently, the [***FREEZE***] directive will be read by DL_FIELD, while ignoring the [***TETHER***] directive.

Note that to emulate a 'frozen' atom in GROMACS, the spring constant of the atom can be set to a large value to effectively 'freeze' the atom in place.

23. Constrained bonds switch. This option, when it is switched on (1), will read [***CONSTRAIN***] directives to decide which atoms to be constrained. This option is different from the [***FREEZE***] directive. In the former case, the atom's position is fixed in space, whereas [***CONSTRAIN***] restricts the position of an atom relative to another atom. This directive is used to constrain bond distances, for instance, to reduce the degrees of freedom and increase the simulation time step. When running simulations, the ***shake*** directive in the DL_POLY's *CONTROL* file must be used.

The [***CONSTRAIN***] directive reads the parameters in the following order:

CONSTRAIN Molecular_Group atom_filter

The *molecular_group* takes the label of the protein chain ID or protein molecular label. Only one [***CONSTRAIN***] directive is allowed for each molecular group. The *atom_filter* key determines which bonds were to be constrained. The available options are shown in the table below.

Atom_filter key	**Selection effects**
h-bond	only affect bonds that contained H atoms
rigid_water	(For water molecules only). Constrain structure become a rigid-like body as a whole
all	All atoms in a Molecular Group

Note that when applying [***CONSTRAIN***] directives with *atom_filter* keys as **h-bond** or **all** to water models, the two O-H bonds of a water molecule will be constrained BUT not the angle (H-O-H). The water molecules are still somewhat flexible.

To make a rigid water molecule, use **rigid_water** *atom_filter* key. This is a special command applicable to water models only. By using this command, the H-H, in addition to O-H bonds, will be constrained. The **rigid_water** *atom_filter* key must always apply to some water models such as the original TIP3P model because there is no LJ parameter on the hydrogen atom and no force constant on the H-O-H

angle. Failure to do so will result in wrong water structure and the simulation may crash.

Section 8.1 lists all the available water models.

Note: Do not confuse **rigid_water** *atom_filter* key with rigid body definition for water (using the [***RIGID***] directive. See Option **24** below). In the former case, constrain methods such as SHAKE algorithm are used to maintain the bond lengths in an iterative manner. For this reason, bond constrain methods are usually computationally slower than rigid body calculations. If your water model is not equilibrated, apply **rigid_water** to the water model in the first instance, setting SHAKE tolerance limit to 10^{-5} or below. Once the model is equilibrated, change the water model to rigid body.

24. Rigid body switch (applicable to DL_POLY FF files only, and certain water models for GROMACS). This option when it is switched on will read [***RIGID***] directives in the *control* file to decide which Molecular Groups where rigid bodies should be applied and defined in the *dl_poly.FIELD* file. For example, the following [***RIGID***] directives state that rigid bodies will be applied to molecules belong to Molecular Groups Grp_A and Grp_C only.

```
...
...
#############################################################
FREEZE   Grp_D   all_backbone

CONSTRAIN  Grp_B h_bond

RIGID    Grp_A
RIGID    Grp_C

#############################################################
```

Do not confuse with the [***RIGID***] directives defined within a MOLECULE. In this case, the [***RIGID***] directives specify how a rigid body should be defined *within* a MOLECULE. Note that, in the above example, rigid bodies only assign to MOLECULEs belong to either Grp_A or Grp_C but not to any other MOLECULEs belong to other molecular groups, even though [***RIGID***] directives are defined in those MOLECULEs. Similarly, no rigid body will be applied if DL_FIELD cannot locate any [***RIGID***] directive in MOLECULEs belong to Grp_A or Grp_C.

25. Periodic boundary condition. Defines the type of periodic boundary conditions for the user system model. This is essential for systems that contain molecules that wrap around the periodic box.

This option takes the following values: 0 (no periodic boundary), 1 (cubic box), 2 (orthorhombic), 3 (parallelepiped) and 6 (slab). These values are the same as that of *imcon* value as described in the DL_POLY manual.

If the value is set to 0, then DL_FIELD will ignore the cell vectors define below or cell parameters define in the user's configuration file. Otherwise, DL_FIELD will read the cell vectors below to determine the boundary conditions.

If the value is set to 'auto', DL_FIELD will automatically determine the simulation box dimensions and assign the appropriate *imcon* value for the DL_POLY's CONFIG

file. In this case, DL_FIELD will ignore the cell vectors define below. Instead, it will look for the unit cell parameters (from the CRYST statement) in the configuration file and convert them to the corresponding cell vectors. If this information is not available, then DL_FIELD will return an error.

26. Cell vectors. Determine simulation box size and shape, centred at (0,0,0). If a periodic boundary condition is defined in Option **25**, then these cell vectors will be used to determine how the boundary condition is applied. DL_FIELD will insert these values into *dl_poly.CONFIG* file.

The cell vectors consist of three sets of numbers, referring to vectors **a**, **b** and **c**:

a = i**x** + j**y** + k**z**, **b** = p**x** + q**y** + r**z**, **c** = u**x** + v**y** + w**z**

where i, j, k, etc, define the simulation box dimension and size. DL_FIELD can inter-convert between cell parameters and cell vectors. For instance, if Option **25** is switched to 'auto', DL_FIELD will look cell parameters in the user configuration file such as the PDB file and convert this to the corresponding cell vectors for *dl_field.CONFIG* file. On the other hand, for GROMACS, this information will be written in the *gromacs.gro* file.

Note: For DL_POLY, the center of the box is (0,0,0), whereas, for GROMACS, this is 0.5(**a** + **b** + **c**). If Option **30** is switched on, DL_POLY will adjust this accordingly.

27. 1-4 scaling factor for coulombic interaction. User can redefine the scaling factor and override the default value for all potential schemes. To retain the recommended values for a potential scheme, just put 'default'. Any other number will be used as the scaling factor instead. For example, if the value is *b*, then this would be the new scaling factor and DL_POLY will scale the 1-4 intra-interactions as: V(1-4) x *b.*

Section 3.6 describes the default scaling factors for ORGANIC force fields.

28. 1-4 scaling factor for vdw interaction. User can redefine the scaling factor and override the default value for all potential schemes. To retain the recommended values for a particular potential scheme, just put 'default'. Any other number will be used as the scaling factor instead. For example, if the value is *b*, then this would be the new scaling factor and DL_POLY will scale the 1-4 intra-interactions as: V(1-4) x *b.*

Section 3.6 describes the default 1-4 scaling factors for ORGANIC force fields.

29. Assign initial Gaussian random velocities, scaled to a temperature, to every ATOM in *dl_poly.CONFIG* file. The random number generator is seeded with system time. Therefore, different velocities are generated each time when DL_FIELD is run.

30. Option to translate user's configuration such that the centre of gravity of the whole system is centred at the origin (0, 0, 0) for DL_POLY; 0.5(**a** + **b** + **c**) for GROMACS.

31. Solvate model. This option instructs DL_FIELD to add solvent molecules to the user's configuration (solute). The number of solvent molecules added will depend on the dimension of the simulation box size specified in Options **25** and **26**, which must be either a cubic (1) or an orthorhombic (2).

Addition of solvent molecules involves the use of appropriate pre-equilibrated solvent template box located in the */solvent* directory. The minimum cell size must not be less than 20 Å and there is no maximum size limit. Ideally, the simulation box size should be in the multiple of 20 Å. Otherwise, the solvent template box will be truncated at the edge of the simulation box.

If Option **30** is switched on, then the whole user's configuration is re-positioned before the solvation is taken place. Distance criteria sets the limit within which no solvent molecule from the solute will be located.

DL_FIELD will skip the solvation step if the keyword 'none' is specified. Otherwise, DL_FIELD will solvate the user's configuration based on the type of solvent specified. The range of solvents available are listed in the file *solvent_list* located in the */solvent* directory, of which solvents labels are the filenames.

In addition to the type of solvents, users can also specify the FF scheme for the solvent. For instance, the example control file shows the following command:

```
tip3p_e  1.7 default
```

This means TIP3P water model for Ewald sum is used as the solvent and the water molecules must be at least 1.7 Å away from the solute and the FF model for the solvent is located in the default FF setting, which is OPLS2005, as specified in Option **1**.

If the FF scheme specified is different from Option **1**, then DL_FIELD will automatically change to a multiple potential scenario and setup the FF model accordingly. For example, the command

```
etoh  1.7 amber16_gaff
```

instructs DL_FIELD to solvate the system with ethanol (EtOH), using Amber GAFF as the FF model for the solvent. The vdw parameters for atoms between the solute and solvent components will be derived according to the mixing rules defined in Options **18** and **19** (for multiple potential).

Warning: solvating a system using a solvent with a different FF scheme is not recommended without validation. This feature is made available in DL_FIELD to solvate inorganic systems, or to 'borrow' a standard FF model such as those of water (TIP3P, TIP4P, etc) that are available in other schemes.

32. Add counter ions. If the total charge in the system is not balanced then set this option to 'on' will cause DL_FIELD to add either chloride (charge value -1.0) or sodium ions (charge value +1.0), depending on the net charge of the system. Charge neutrality cannot be achieved if there is a partial net charge in the system.
33. No longer use.
34. No longer use.
35. No longer use.

Options **33** to **35** are no longer applicable. They were used to carry out MM calculation, which can be more conveniently obtained by running the calculations in packages as shown below.

4.1 Running DL_POLY Program

DL_FIELD *control* file also contains a *DL_POLY control* section. It instructs DL_FIELD how to run DL_POLY via a fork process (for 1 processor). For multi-processor run, a *mpirun* command will be issued. Note that the *DL_POLY control* feature is only applicable to Unix/Linux (including the Window Subsystem for Linux, WSL), or within the Cygwin environment for Window machines. In addition, a pre-compiled DL_POLY program must also reside in the same machine.

The *DL_POLY control* feature can carry out two independent tasks: single-point calculation and multi-tier equilibrations. The feature forms part of the integrated workflow environment within DL_FIELD. Once DL_FIELD has setup the force field model, it will then automatically create the necessary sets of DL_POLY files: CONFIG (copied from *dl_poly.CONFIG*), FIELD (copied from *dl_poly.FIELD*) and CONTROL files (created according to the user-selected task). These DL_POLY files will be automatically copied into the directory where DL_POLY program is located. The output results will also locate here.

Below shows a *DL_POLY control* section.

```
************* DL_POLY control ******************
1         * Run DL_POLY program
DLPOLY.Z   * DL_POLY executable filename
/home/user/dl_field_4.10/output * absolute path to DL_POLY program
4          * Number of processors (mpirun)
1         * MM calculation 1=on  0=off
0   1000   * Structural Relaxation level (0 - off, 1,2 or 3). Total timestep
9.0        * cutoff (vdw and electrostatic)
1000       * Time limit for DL_POLY run (in seconds)
```

The correct DL_POLY path must be specified (highlighted in **bold**). By default, the executable filename for DL_POLY is *DLPOLY.Z*. Simulation time can be controlled by adjusting the MD time step and the non-bonded cut off value. This may be crucial, especially if the user system model is large, as the simulation is done on a local machine (rather than an HPC!).

Number of processors

This option allows user to specify the number of processors use for DL_POLY runs. This only works for DL_POLY program that is precompiled with parallelisation.

Single point calculation

DL_FIELD will create a simple CONTROL file with zero-time step. This means no dynamics will be carried out. Instead, DL_POLY exits once the configuration energy of the initial system is calculated.

Equilibration Timestep

Strictly speaking, DL_FIELD does not carry out the normal equilibration process, per se. Rather, the CONTROL files contain sets of DL_POLY *directives* to suppress run-away system energies and to disentangle any locked, high-energy conformations, perhaps due to poor initial configuration setup. Therefore, the equilibration features as mentioned here is by no means a substitution to the proper equilibration processes. The final CONFIG file, which contained the relaxed structure, can then be transferred to an HPC system to carry out further equilibration processes.

Up to three levels of equilibrations can be specified. The number refer to the successive DL_POLY runs, each with a different CONTROL file that contain directives to eventually relax the system model. Level 1 is the least restrictive process and run DL_POLY once. Level 2 will run DL_POLY twice with two different CONTROL files, and so on.

The value of timestep is specified in unit ps.

Cutoff

The cutoff values for non-bonded vdw and electrostatic (real space) calculations in DL_POLY.

Time limit

The maximum time limit for simulation run, in seconds.

4.2 Running GROMACS Program

This option only works if GROMACS program is installed in the computer and the *gromacs* keyword is used for Option **B**, to produce force field files for GROMACS. The *GROMACS Control* section is shown below:

```
************* Gromacs control ******************
1           * Run Gromacs
gmx           * Gromacs executable filename
/usr/bin/ * absolute path to Gromacs
1           * MM single-point calc.
```

The correct GROMACS path must be specified (highlighted in **bold**). By default, the executable filename for GROMACS is *gmx*. The only option to run GROMACS is a single-point MM calculation, and this option must always switch on.

Before running GROMACS, DL_FIELD will setup a *gromacs_sp.tpr* file by calling the GROMACS *grompp* as follows:

gmx grompp -f gromacs.mdp -c gromacs.gro -p gromacs.top -o gromacs_sp.tpr

where *gromacs.mdp* (parameters adjusted for single-point run), *gromacs.gro* and *gromacs.top* are the matching files produced from DL_FIELD conversion prior to GROMACS run. A *tpr* file is a portable binary run input file that combines the starting structure, force field topology and control file into a single binary file. A successful production of a *tpr* file means the syntax and input simulation files are matched.

After that, DL_FIELLD runs GROMACS using the following command:

gmx mdrun -v -deffnm gromacs_sp -ntmpi 1

This instructs *gmx* to read *gromacs_sp.tpr* for simulation run. After a successful run, GROMACS will produce the following files: *gromacs_sp.edr* and *gromacs_sp.log*.

All GROMACS output files will be located at the *output/* folder, as indicated in the *dl_f_path* file.

4.3 Running LAMMPS Program

This option only works if LAMMPS program is installed in the computer and the *lammps* keyword is used for Option **B**, to produce force field files for LAMMPS. The *LAMMPS Control* section is shown below:

```
************ Lammps control ******************
0          * Run Lammps
lmp         * Lammps executable filename
/usr/bin/ * absolute path to Lammps
1          * MM single-point calc.
```

The correct LAMMPS path must be specified (highlighted in **bold**). By default, the executable filename for LAMMPS is *lmp*. The only option to run LAMMPS is a single-point MM calculation, and this option must always switch on.

After setting up the input files, DL_FIELD runs LAMMPS using the following command:

lmp -in lammps.in

where *lammps.in* is the input file generated by DL_FIELD which contains details about number of data files to read. After simulations, all LAMMPS output files, including the log file, *log.lammps*, will be located at the *output/* folder, as indicated in the *dl_f_path*.

5 User-defined Force Field (*udff*)

While new force field information can be added to the standard library files, **.sf* and **.par*, such practices can become inconvenient and error-prone when user-defined models were to migrate to latest standard library files whenever a new DL_FIELD version is released.

However, new parameters and molecular templates can be defined in the user-defined force field (*udff*) file, without having to tamper with the standard library files. Users can simply migrate *udff* files to future DL_FIELD releases that contain the latest update of the standard library files.

There are two *udff* files included as examples, the alcohol*.udff* and the *vdw_fix.udff*. To use these files, just insert the appropriate path and the filename in Option **6** in the *dl_field.control* file. Only one *udff* file can be used at any one time. Users can create as many *udff* files as they like but the files created must always end with the extension *.udff*.

5.1 File Format

The *udff* file can be treated as a combination of both *.sf* and *.par* files. In other words, all *DIRECTIVES* from the *.sf* and *.par* files can be included in the *udff* file.

The order of a ***DIRECTIVE*** is not important, so long it is enclosed with the appropriate ***END DIRECTIVE*** statement. For instance, the potential parameters definition can be sandwiched somewhere between two ***MOLECULE*** definitions. Such flexibility allows user to arrange information in the file according to personal tastes.

Same as the standard library files, a *udff* file must include the directives ***POTENTIAL*** *potential_scheme* and ***UNIT*** *energy_unit*. DL_FIELD allows multiple ***POTENTIAL*** definitions in the *udff* file. All information belongs to the relevant force field scheme must be enclosed within the appropriate ***POTENTIAL*** and ***END POTENTIAL*** directives. If no *udff* information can be located for the force field scheme specified in the *control* file (Option **1**), DL_FIELD will notify this to the user but **will not** treat this as an error.

5.2 Functionality

When an *udff* file is specified in Option **6** of the *control* file, DL_FIELD will set up a molecular model database in the computer memory by first reading information from the *udff* file before it proceeds to read the standard library file of the selected potential scheme (Option **1**) in the *control* file.

MOLECULEs in the standard library files can be overridden by redefining them in the *udff* file. To override an existing MOLECULE, redefine it in the *udff* file with the same MOLECULE_TYPE. For instance, consider the *alcohol.udff* file in the *Examples/* directory, the MOLECULE methanol is redefined in the *udff* file. The only changes made are the charges on some of the ATOMs (compare this with methanol defined in *DLPOLY_CHARMM22_prot.sf*).

By switching the Verbosity mode Option **7** to 'on', DL_FIELD will display which MOLECULEs have been overridden.

Potential parameters can also be overridden if ATOM_KEYs are matched. This can be either matched with ATOM_KEYs appear in the same sequence or with the alternative but equivalent sequence. For instance, an angle interaction is defined for ATOM_KEYs A-B-C

in a *.par* file. DL_FIELD will subsequently override the corresponding potential parameters even if it is defined as C-B-A in the *udff* file.

The *udff* file cannot override an existing ATOM_TYPE defined in the standard *.sf* file. This is because any change to the ATOM_KEY will lead to invalid potential sets defined in the standard *.par* file. However, a new ATOM_TYPE can still be defined, either with a completely new ATOM_KEY or with an existing ATOM_KEY from the library file. In other words, the ATOM_TYPE is unique while the ATOM_KEYs are not.

Each ***POTENTIAL*** block in *udff* file must contain its own ***UNIT***. If potential parameters are also specified, DL_FIELD will assume the values are described in the same energy unit. DL_FIELD will do automatic conversion if the unit is different from what is specified in the control file (Option **2**), which is the final unit in the *dl_poly.FIELD* file.

5.3 UDFF User Model Files

When Option **C** is selected to ON (1), DL_FIELD will create a new file called *dl_field.udff*. This is a model-specific *udff* model file. It is called the *udff* user model file. It contains all FF information of an input system configuration, including the coordinates of the configuration itself.

It can be used for purposes of model reproducibility and FF repertoire. Of note is that the file contents can be very different from the 'normal' *udff* file as mentioned earlier. Below shows a section of *dl_field.udff* content.

```
#-------------------------------------------------------
#   DL_FIELD user udff model file (atom set mode)
#        Generated from DL_FIELD version 4.13
#          Created: 2026-05-13 11:14:40
#-------------------------------------------------------
…
…
UDFF_MODE = ALL_INCLUSIVE

POTENTIAL opls2005

UNIT kcal/mol

MOLECULAR_GROUP 2
1. CYC
2. PHE
END MOLECULAR_GROUP
…
…
…
ANGLE_SET               0.5K    theta0
MOLECULAR_GROUP CYC
2       1       46          90.000000     108.567000
2       1       50          90.000000     108.567000
2       1       78          45.000000     112.011000
46      1       50          92.600000     111.550000
46      1       78          35.000000     109.500000
50      1       78          35.000000     109.500000
1       2       3           75.000000     109.430000
1       2       43          90.000000     111.587000
…
```

What tells DL_FIELD apart from normal *udff* files is the special directive

UDFF_MODE = ALL_INCLUSIVE

This instructs DL_FIELD to obtain all FF information from *dl_field.udff* to produce the identical FF model files, without the need of referencing to FF library files, nor the supply of the input configuration. DL_FIELD will also ignore most options in the control file. The type of output files produce depends on selection of Option **B** (either *gromacs*, *lammps*, or *none*), but DL_POLY FF files will always produce.

Note that this is a new feature first implemented in Version 4.13. Not all system characteristics will be recorded in the file. For instance, *dl_field.udff* does not include information about rigid body, atom states, and bond constrain, although these states are still defined in FF files in normal DL_FIELD runs. More features will be included in subsequent versions.

To read back the *udff* model file, insert the file in Option **6**. Obviously, *dl_field.udff* can be renamed to anything that ends with the extension *udff*, so long the ALL_INCLUSIVE directive is retained.

5.3.1 File format

Unlike the normal *udff* files, *udff* user model files have their own sets of directives. Mixing directives from two different *udff* formats is not allowed.

DL_FIELD obtained FF information and energy unit from the file, rather than the *control* file.

After that DL_FIELD read the Molecular Group information, which summarises how the molecular system is organised. DL_FIELD will use the information to setup the FF data files.

```
MOLECULAR_GROUP 2
1. CYC
2. PHE
END MOLECULAR_GROUP
```

Each Molecular Group will have its own set of interaction components as shown below:

```
…
…
ANGLE_SET                0.5K    theta0
MOLECULAR_GROUP CYC
2       1       46          90.000000     108.567000
2       1       50          90.000000     108.567000
2       1       78          45.000000     112.011000
46      1       50          92.600000     111.550000
46      1       78          35.000000     109.500000
50      1       78          35.000000     109.500000
1       2       3          75.000000     109.430000
1       2       43          90.000000     111.587000
END ANGLE_SET
…
…
ANGLE_SET                0.5K    theta0
MOLECULAR_GROUP PHE
2       1       3       138.072000 107.800000
2       1       4       138.072000 107.800000
2       1       5       156.900000 110.700000
3       1       4       138.072000 107.800000
END ANGLE_SET
```

In this example, angle parameter sets are separately listed for Molecular Groups CYC and PHE, respectively.

Below shows a *udff* user model file for zonisamide, a small drug molecule, to illustrate the input configuration is saved in the *dl_field.udff* model file.

```
# User configuration.
# Source: test_structures/zonisamide.xyz
# ID   Type                    x          y          z         charge  imp_flag neighbour_number - members
CONFIG
MOLECULAR_GROUP XYZ
1    C6_benzisoxazole          -3.287000    1.859000   -0.035000 -0.115000 1 3    - 2    6    15
2    C5_benzisoxazole          -3.244000    0.515000    0.017000 -0.115000 1 3    - 1    3    16
3    C4_benzisoxazole          -2.069000   -0.138000    0.054000 -0.115000 1 3    - 2    4    17
4    C3a_benzisoxazole         -0.954000    0.601000    0.037000 -0.205200 1 3    - 3    5    7
5    C7a_benzisoxazole         -0.991000    1.937000   -0.018000  0.282200 1 3    - 4    6    9
6    C7_benzisoxazole          -2.159000    2.591000   -0.053000 -0.115000 1 3    - 1    5    18
7    C3_benzisoxazole           0.347000    0.291000    0.065000  0.343000 1 3    - 4    8    10
8    N2_benzisoxazole           0.976000    1.389000    0.028000 -0.413000 0 2    - 7    9
9    O1_benzisoxazole           0.142000    2.414000   -0.024000 -0.122000 0 2    - 5    8
10   CS_sulphonamide            1.002000   -1.058000    0.130000 -0.220500 0 4    - 7    11   19   20
11   S_sulphonamide             2.762000   -1.032000    0.028000  0.934900 0 4    - 10   12   13   14
12   Np_sulphonamide            3.520000   -2.554000    0.097000 -0.767800 1 3    - 11   21   22
13   O_sulphonamide             3.324000   -0.867000   -1.312000 -0.468000 0 1    - 11
14   O_sulphonamide             3.473000   -0.673000    1.255000 -0.468000 0 1    - 11
15   HC_aromatic               -4.264000    2.371000   -0.064000  0.115000 0 1    - 1
16   HC_aromatic               -4.184000   -0.061000    0.030000  0.115000 0 1    - 2
17   HC_aromatic               -2.024000   -1.237000    0.099000  0.115000 0 1    - 3
18   HC_aromatic               -2.191000    3.691000   -0.094000  0.115000 0 1    - 6
19   HC_alkane                  0.711000   -1.548000    1.087000  0.195000 0 1    - 10
20   HC_alkane                  0.614000   -1.683000   -0.707000  0.195000 0 1    - 10
21   HN_sulphonamide            4.195000   -2.571000    0.861000  0.357200 0 1    - 12
22   HN_sulphonamide            4.060000   -2.712000   -0.752000  0.357200 0 1    - 12
END CONFIG
```

6 User's Atomic Configuration File Format

DL_FIELD recognises user's configuration file in the following formats: PDB, *mol2* and simple *xyz*. Note that not all DL_FIELD features are made available to all file formats. This is especially true for the *mol2* format, since it is the newest format being introduced into DL_FIELD framework.

PDB file format is applicable to all force field schemes implemented in DL_FIELD.

6.1 The PDB Format

Although PDB format is a popular format for proteins and other bio molecules, other molecular models such as metals and inorganic molecules can also be described in the PDB format.

DL_FIELD adheres to strict PDB standard format and different information must contain within the appropriate columns in a PDB file. However, not all information will be required by DL_FIELD and only those relevant to the conversion process will be discussed here.

Each ATOM definition that precedes with the PDB statement 'ATOM' and/or 'HETATM' will be read by DL_FIELD. Table below lists data that will be read by DL_FIELD for each ATOM.

DL_FIELD also recognises the CRYST1 statement which contains the unit cell parameters of the system. By setting the periodic boundary condition to *auto* (Option **25**) in the *control* file, the program will look for the cell parameter information and convert it to the corresponding cell vectors.

Note that DL_FIELD will process a PDB file until the END statement is encountered. **Any atoms after these statements will be ignored**. In addition, DL_FIELD will also ignore the 'TER' statement.

Column range (inclusive)	**Data**	**Remark**
13-16	*Atom label*	If *element symbol* is not defined, then this data will be treated as the element symbol for the atom. Any numerical characters will be ignored.
18-20	*Residue names*	Residue names such as those of amino acids. Equivalent to the MOLECULE_KEYs.
21	*Reserves for residue names*	Some residue names such as those of carbohydrates may contain more than three characters. The fourth one will be located here.
22	*Chain ID*	If *Molecular Group name* is not defined, this information will be used as the Molecular Group for the atom.
23-26	*Residue sequence*	Numerical sequence for molecules such as amino acids or carbohydrates.
29-56	*X,y,z coordinates*	Location of the atom in x,y,z coordinates.
70-76	*Molecular group name*	Molecular group name definition. This will override *Chain ID* if it is defined.
77-79	*Element symbol*	The element symbol of the atom. If this is not defined, then *Atom label* will be used to determine the element symbol of the atom.

6.1.1 Pre-processing

Most of the PDB files for bio molecules were constructed from X-ray crystallography data. These 'raw PBD' files must be pre-processed by the user using some other third party software to produce a 'readable format' for DL_FIELD. The following steps are needed to be taken by the users.

(1) Decide which molecular residues to delete that involve multiple occupancies in crystalline structures.

(2) Usually, hydrogen atoms are not visible to X-rays and they are not assigned in the PDB file. The molecules must therefore be pre-filled with hydrogen atoms by using other third-party packages. The terminal residues must also be appropriately terminated.

(3) In most cases, DL_FIELD defines a bond between two atoms according to distance criteria. The connectivity of atoms within a molecule must be within a reasonable range and not deviate significantly from the equilibrium length. The distance criteria can be relaxed or tightened somewhat using Option **3**, the Conversion Criteria in the *control* file.

(4) For protein systems, decide the charge states of some amino acids, such as histidine (HIS), lysine (LYS), glutamic acid (GLU), etc., by adding or deleting the appropriate hydrogen atoms. Force field schemes such as CHARMM, AMBER, etc will have different three-letter notations for amino acids of different charge states. However, standard notation such as HIS (for histidine for instance) can be left unchanged and DL_FIELD will automatically determine the charge state based on the positions of the hydrogen atoms and assign the appropriate potential parameters accordingly.

(5) For cysteine (CYS) molecules with disulphide bonds, thiolate hydrogen atoms must be removed. DL_FIELD will automatically define a disulphide bonds between the molecules, provided the S-S distance is within a 'sensible' value.

6.1.2 Conversion Processes

The conversion of a PDB file into *dl_poly.CONFIG* and *dl_poly.FIELD* files involve several stages. First, the PDB file is scanned once to determine the number of Molecular Group available. Then, ATOMs and their interactions will be grouped according to different molecular groups in the *dl_poly.FIELD* file.

A large multi-component molecular system can be grouped into several Molecular Groups so that each Molecular Group contain a component of the molecular system. For instance, group a protein molecule into a Molecular Group and define all water molecules with another Molecular Group. This not only saves computational resources on memory usage but also produces a more compact FIELD file layout.

Note that DL_FIELD assumes no covalent bond between ATOMs from any two different Molecular Groups.

After that, DL_FIELD will read the PDB file in more details, one residue at a time. The residue (equivalent to the MOLECULE_KEY) can either be a single distinct molecule or a linked molecule (like an amino acid residue in a protein). Firstly, DL_FIELD will look for any MOLECULE templates in the *udff* file. After that, it will search the *.sf* library file, looking for a match with the residue structure.

It is considered a match if the residue label is matched with a MOLECULE_KEY in the library file. However, DL_FIELD will still flag up an error if the matched MOLECULE template is

topologically not the same as that of residue structure (by matching with the connectivity matrix).

The next step is to rearrange the atoms within the residue according to the sequence defined in the MOLECULE template. All ATOMs will be written out in the same sequence to the *dl_poly.CONFIG* and *dl_poly.FIELD* files. The molecular interactions are then defined base on the connectivity information of the MOLECULE template.

6.1.3 Linked Residues in Proteins

For protein molecules, the residues are automatically converted from a three-letter notation, ABC, to -ABC- to indicate linkages exist between molecules and DL_FIELD will define potential parameters involving these linkages. However, the three-letter notation is retained (without the hyphens) if it is an isolated molecule.

User can use standard three-letter notations for amino acids with various charge states. For instance, use HIS rather than explicitly define HSD or HSE for histidine residues in CHARMM. DL_FIELD will automatically determine the nature of charge state and assign the potential parameters accordingly.

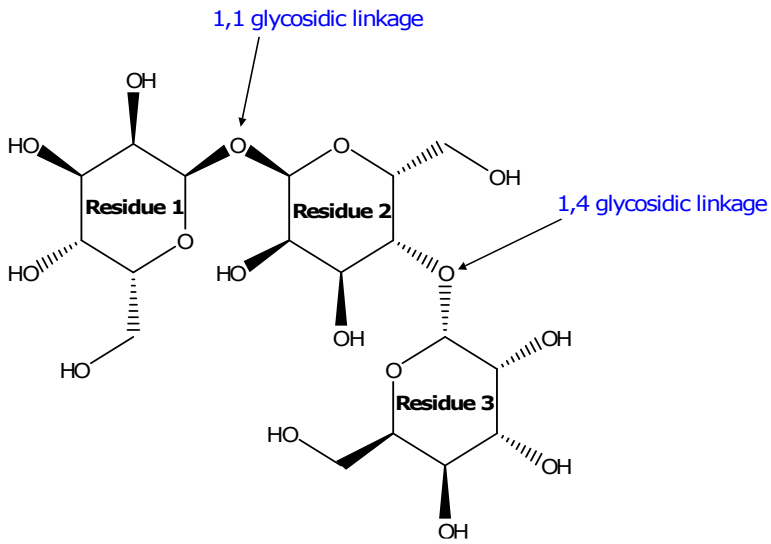
The standard three-letter notations for terminal amino acids can also be used. DL_FIELD will automatically determine the type of terminal states, such as normal ammonium or carboxylate terminals or even acetylated N-terminus or methylated C-terminus etc.

6.1.4 Linked Residues in Carbohydrates

For carbohydrates molecules in CHARMM (note: This section does not apply to CHARMM36_carb): The simplest carbohydrate molecules exist as monomers called monosaccharides. It can form polymeric structures via glycosidic oxygen linkages to give disaccharides, oligosaccharides and polysaccharides. The basic monosaccharides are usually consisted of six-membered pyranose sugars such as glucose and their anomeric counterparts (the α- and β- anomers). Other members of pyranose monosaccharides also exist, with the -OH groups adopting various orientations. Examples of these are altrose, allose, galactose, etc.

In CHARMM force field, all pyranose molecules are based on α-glucose (AGLC). The program will identify various keywords in CHARMM, referring to different pyranose members as AGLC. If the user is unsure of the identity of a pyranose member, just define the keyword 'AGLC' in the *Residue names* column (18-21) in the PDB file.

For example, consider a carbohydrate molecule consists of three pyranose units connected via 1,1 and 1,4 glycosidic linkages as shown in the diagram below. The corresponding PDB file (*glucose3_1.pdb*) is available in the *Examples/* directory in the DL_FIELD package.



Another carbohydrate structure, *glucose3_2.pdb*, is also contained in the *Examples/* directory. Both structures are identical with identical geometrical structure. The only difference is the assignment of oxygen linkage. For instance, the 1,1 glycosidic oxygen is assigned to residue id = 2 in *glucose3_1.pdb* whereas, the same oxygen is assigned to residue id = 1 in *glucose3_2.pdb*. DL_FIELD will be able to convert both structures to give similar potential interactions.

The exact way of grouping of the glycosidic oxygen into a residue is not important. However, inner pyranose units must always contain one gylcosidic oxygen (Residue 2 in this example).

The Glycam potential is a stand-alone potential force field that is compatible with AMBER force field. It has a complete set of all anomeric and enantiomeric structures of sugars, and some of the derivatives such as amino sugars. The corresponding potential parameters use case-sensitive atom types to distinguish them from other standard AMBER potentials. For instance, the aliphatic carbon is defined as 'Cg' in Glycam. In the case of DL_FIELD, a prefix 'G_' is added to indicate such distinction. Therefore, Cg in the standard Glycam potential is equivalent to G_Cg in DL_FIELD.

The complexity and variety of glyco-molecules can be systematically classified based on notations suggested by Woods group (M. L. DeMarco and R. J. Woods, 'Structural glycobiology: A game of snakes and ladders' *Glycobiology*, **18**, 426-440 (2008)). DL_FIELD fully adopts this notation. In the case of L-sugars, the notation is expressed as XxX, where the middle character is a small letter x. DL_FIELD will automatically reassign it as a corresponding D-sugar with capital 'X'. This should not change the potential assignment since chemically both D- and L-sugars are identical with the same set of atomic partial charges.

All Glycam structures are defined as individual unit links. For this reason, DL_FIELD will automatically reassign a notation XXX as -XXX- internally during the FF setup processes.

6.1.5 Nuclei Acids

Nuclei acids consist of three components: a 5-carbon sugar unit, a phosphate group, and a nucleobase. For the sugar unit, it can be either ribose or the deoxyribose derivative, of which the hydroxyl group at C2 is replaced with a hydrogen for the derivative form. For

this reason, nucleic acids can be classified as either RNA (R) or DNA (D), which correspond to nucleic acids with ribose or deoxyribose units, respectively.

There are altogether five different nucleobases: adenine (A), cytosine (C), guanine (G), thymine (T) and uracil (U). However, uracil can only be found in RNA, while thymine can only be found in DNA. Nuclei acids are therefore named after the nucleobases.

Nuclei acids formed phosphate links at the 3-and 5-position of the sugar units to form a nucleic acid sequence of polymeric chain, adopting a helical strand. RNA consists of one such strand, whereas DNA consists of two helical strands that bind to each other via hydrogen bonds between two nucleobases – one from each stand. The specific order of these nucleobase-pairs enables storing and transmitting coded instruction as genes.

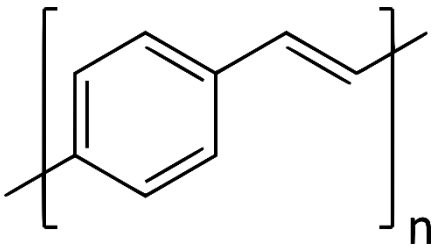
In DL_FIELD, the following standard notation for nucleic acids are used: the first character is always either 'R' or 'D', referring to RNA or DNA, respectively. The second character represent the nucleobases, as shown above. For example, for adenine in RNA, the residue notation is -RA-. However, only the residue label RA, without hyphen, is needed in PDB. DL_FIELD will automatically add the hyphens if it is a nucleic acid residue that forms phosphate links. Otherwise, it would be treated as a stand-alone nucleotide with no link and the original label, RA, is retained.

The chain termination occurs at the 3- and 5- positions of the sugar units, which can be hydroxyl phosphate (P), or hydroxyl (O). DL_FIELD will automatically detect the terminal type and assign suitable parameters accordingly. For example, for RNA adenine with hydroxyl phosphate terminal at the 5-position, the residue label with be changed from RA to -RA5P-. For DNA guanine with hydroxyl termination at the 3-position, the residue label will be changed from DG to -DG3O-.

6.1.6 Polymeric materials

A polymer is a macro molecule consists of many repeating units, forming a chain and usually terminates with terminal units at both ends. Although there are various ways to setup a polymeric system for DL_FIELD, one way is to use MOLECULE templates and input structures in PDB format to setup polymeric systems of any length.

Consider setting up force field models for poly(*p*-phenylene vinylene) with the following structural formula:



Three MOLECULE templates can be setup in a *udff* file: one terminal unit with CH3 termination at the benzene ring, one for the repeating unit and the other terminal unit with the vinyl termination. By using CHARMM36_CGenFF as the FF scheme, the templates can be constructed in a *udff* file as follows:

```
MOLECULE_TYPE
p_phenylene_vinylene_unit -PV-  102.140    ! repeating unit for Poly(p-phenylene vinylene).
p_phenylene_vinylene_t1  -PV1-  103.148    ! Terminal unit for Poly(p-phenylene vinylene)
p_phenylene_vinylene_t2  -PV2-  103.148    ! Second terminal unit for Poly(p-phenylene
vinylene)
END MOLECULE_TYPE
```

```
MOLECULE p_phenylene_vinylene_t1 15 -0.25
C7  CL_alkene  -0.400
H7  HCL_alkene  0.150
C8  CL_alkene  -0.400
H81 HCL_alkene  0.150
C1  C_benzene   0.250
C2  C_benzene  -0.115
H2  HC_benzene  0.115
C3  C_benzene  -0.115
H3  HC_benzene  0.115
C4  C_benzene  -0.115
H4  HC_benzene  0.115
C5  C_benzene  -0.115
H5  HC_benzene  0.115
C6  C_benzene  -0.115
H6  HC_benzene  0.115
CONNECT C7  > H7 C8 C1
CONNECT H7  > C7
CONNECT C8  > H81 C7
CONNECT H81 > C8
CONNECT C1  > C7 C2 C6
CONNECT C2  > H2 C1 C3
CONNECT H2  > C2
CONNECT C3  > C2 C4 H3
CONNECT H3  > C3
CONNECT C4  > C3 C5 H4
CONNECT H4  > C4
CONNECT C5  > C4 C6 H5
CONNECT H5  > C5
CONNECT C6  > C1 C5 H6
CONNECT H6  > C6
END MOLECULE
```

```
MOLECULE p_phenylene_vinylene_t2 15 0.25
C7  CL_alkene  -0.400
H7  HCL_alkene  0.150
C8  CE_alkene  -0.300
H81 HCL_alkene  0.150
H82 HCL_alkene  0.150
C1  C_benzene   0.250
C2  C_benzene  -0.115
H2  HC_benzene  0.115
C3  C_benzene  -0.115
H3  HC_benzene  0.115
C4  C_benzene   0.250
C5  C_benzene  -0.115
H5  HC_benzene  0.115
C6  C_benzene  -0.115
H6  HC_benzene  0.115
CONNECT C7  > H7 C8 C1
CONNECT H7  > C7
CONNECT C8  > H81 H82 C7
CONNECT H81 > C8
CONNECT H82 > C8
CONNECT C1  > C7 C2 C6
CONNECT C2  > H2 C1 C3
CONNECT H2  > C2
CONNECT C3  > C2 C4 H3
CONNECT H3  > C3
CONNECT C4  > C3 C5
CONNECT C5  > C4 C6 H5
CONNECT H5  > C5
CONNECT C6  > C1 C5 H6
CONNECT H6  > C6
END MOLECULE
```

```
MOLECULE p_phenylene_vinylene_unit 14 0.0
C7  CL_alkene  -0.400
H7  HCL_alkene  0.150
C8  CL_alkene  -0.400
H81 HCL_alkene  0.150
C1  C_benzene   0.250
C2  C_benzene  -0.115
H2  HC_benzene  0.115
C3  C_benzene  -0.115
H3  HC_benzene  0.115
C4  C_benzene   0.250
C5  C_benzene  -0.115
H5  HC_benzene  0.115
C6  C_benzene  -0.115
H6  HC_benzene  0.115
CONNECT C7  > H7 C8 C1
CONNECT H7  > C7
CONNECT C8  > H81 C7
CONNECT H81 > C8
CONNECT C1  > C7 C2 C6
CONNECT C2  > H2 C1 C3
CONNECT H2  > C2
CONNECT C3  > C2 C4 H3
CONNECT H3  > C3
CONNECT C4  > C3 C5
CONNECT C5  > C4 C6 H5
CONNECT H5  > C5
CONNECT C6  > C1 C5 H6
CONNECT H6  > C6
END MOLECULE
```

H7 H81
C7===C8—(link)
C1
H2-C2 C6-H6
H3-C3 C5-H5
C4

H7 H81
C7==C8-H82
C1
H2-C2 C6-H6
H3-C3 C5-H5
C4
(link)

H7 H81
C7==C8-(link)
C1 repeating unit
H2-C2 C6-H6
H3-C3 C5-H5
C4
(link)

Charges are obtained from the CHARMM MATCH web server. The residue labels, or MOLECULE_KEYs for terminal units are -PV1- and -PV2-, respectively, while the repeating unit is -PV-. The hyphens indicate these residues are linking units.

However, in PDB file, these are expressed as PV, PV1, etc. DL_FIELD will automatically detect links and internally reassign to -PV- and -PV1-, respectively. Below shows a section of the PDB file for the polymer:

```
REMARK  Poly(p-phenylene vinylene), n = 60
ATOM     1  C   PV1    1     -2.365  -0.144   0.216  1.00  0.00         C
ATOM     2  C   PV1    1     -1.542  -1.221  -0.117  1.00  0.00         C
ATOM     3  C   PV1    1     -0.158  -1.044  -0.222  1.00  0.00         C
ATOM     4  C   PV1    1      0.419   0.224   0.007  1.00  0.00         C
ATOM     5  C   PV1    1      1.880   0.482  -0.087  1.00  0.00         C
ATOM     6  C   PV1    1      2.800  -0.434  -0.394  1.00  0.00         C
ATOM     7  C   PV1    1     -0.426   1.299   0.341  1.00  0.00         C
ATOM     8  C   PV1    1     -1.808   1.114   0.444  1.00  0.00         C
ATOM     9  H   PV1    1     -1.976  -2.197  -0.294  1.00  0.00         H
ATOM    10  H   PV1    1      0.440  -1.904  -0.481  1.00  0.00         H
ATOM    11  H   PV1    1      2.232   1.489   0.105  1.00  0.00         H
ATOM    12  H   PV1    1      2.558  -1.464  -0.603  1.00  0.00         H
ATOM    13  H   PV1    1     -0.015   2.284   0.523  1.00  0.00         H
ATOM    14  H   PV1    1     -2.446   1.949   0.702  1.00  0.00         H
ATOM    15  H   PV1    1     -3.447  -0.287   0.297  1.00  0.00         H
ATOM    16  C   PV     2      4.245  -0.037  -0.455  1.00  0.00         C
ATOM    17  C   PV     2      5.214  -0.979  -0.804  1.00  0.00         C
ATOM    18  C   PV     2      6.563  -0.613  -0.863  1.00  0.00         C
ATOM    19  C   PV     2      6.958   0.711  -0.569  1.00  0.00         C
ATOM    20  C   PV     2      8.372   1.166  -0.612  1.00  0.00         C
ATOM    21  C   PV     2      9.415   0.395  -0.926  1.00  0.00         C
ATOM    22  C   PV     2      5.968   1.647  -0.219  1.00  0.00         C
ATOM    23  C   PV     2      4.622   1.274  -0.162  1.00  0.00         C
ATOM    24  H   PV     2      4.920  -1.996  -1.030  1.00  0.00         H
ATOM    25  H   PV     2      7.278  -1.374  -1.137  1.00  0.00         H
ATOM    26  H   PV     2      8.580   2.203  -0.371  1.00  0.00         H
ATOM    27  H   PV     2      9.320  -0.649  -1.181  1.00  0.00         H
ATOM    28  H   PV     2      6.237   2.671   0.013  1.00  0.00         H
ATOM    29  H   PV     2      3.870   2.005   0.109  1.00  0.00         H
ATOM    30  C   PV     3     10.794   0.985  -0.930  1.00  0.00         C
ATOM    31  C   PV     3     11.890   0.198  -1.288  1.00  0.00         C
ATOM    32  C   PV     3     13.178   0.743  -1.293  1.00  0.00         C
ATOM    33  C   PV     3     13.382   2.095  -0.938  1.00  0.00         C
ATOM    34  C   PV     3     14.723   2.738  -0.923  1.00  0.00         C
ATOM    35  C   PV     3     15.867   2.127  -1.236  1.00  0.00         C
…
…
ATOM   833  C   PV2   60     55.807  60.967  70.188  1.00  0.00         C
ATOM   834  C   PV2   60     55.989  57.463  68.907  1.00  0.00         C
ATOM   835  C   PV2   60     56.799  56.553  68.223  1.00  0.00         C
ATOM   836  H   PV2   60     59.498  58.537  67.659  1.00  0.00         H
ATOM   837  H   PV2   60     58.105  60.148  68.850  1.00  0.00         H
ATOM   838  H   PV2   60     54.559  59.319  70.199  1.00  0.00         H
ATOM   839  H   PV2   60     56.738  61.443  69.922  1.00  0.00         H
ATOM   840  H   PV2   60     55.012  57.141  69.247  1.00  0.00         H
ATOM   841  H   PV2   60     56.447  55.545  68.041  1.00  0.00         H
ATOM   842  H   PV2   60     55.071  61.566  70.732  1.00  0.00         H
END
```

6.2 The XYZ Format (*.xyz*)

Unlike the PDB file format, the *xyz* format contains much fewer information and is free-formatted. DL_FIELD only reads the element symbols and the corresponding coordinates of the structure. Below shows the general form of an *xyz* file.

```
[Number of atoms]
[SPACE, or CRYST1 statement or some remark or title]
Element_symbol      x1   y1   z1
Element_symbol      x2   y2   z2
Element_symbol      x3   y3   z3
.....
```

Remark statements can be placed anywhere in the file, with a symbol '#' at the beginning of each line.

The *xyz* file format is applicable to all FF schemes, including those of inorganic. For ORGANIC FF, template-based FF schemes such as CHARMM and AMBER, DL_FIELD will attempt to identify the types and number of molecules in a Molecular Group. After that, DL_FIELD will search through the library and look for correct templates based on number of atoms in a molecule and its corresponding molecular weight as matching criteria. For ORGANIC FF schemes that allow topological analysis approach, DL_FIELD will bypass any attempt to look for MOLECULE templates. Instead, DL_FIELD will carry out fully automatic assignment of ATOM_TYPEs and determine the corresponding potential parameters accordingly, including any improper and inversion potentials. For example: OPLS, CVFF and PCFF all allow topological-based molecular analysis.

For INORGANIC force fields, the ***MOLECULE_KEY*** directive must be used (see below).

6.2.1 The xyz features

The various directives that can be used in xyz files are described in more details as follows:

(a) CRYST1 statement

This is same as the CRYST1 statement in a PDB file, to specify the crystal structure and cell parameters of a structure. DL_FIELD recognises such statement if it appears at the 'title' section of the file, which may look something like this:

```
451
CRYST1   83.908   20.7302253   17.9859383759  90.00  90.00  90.00 (P 1)
C   10.000000  1.390000  0.000000
C   10.000000  0.695000  1.204000
C   10.000000  -0.695000  1.204000
…
…
```

DL_FIELD will read the CRYST1 statement and convert the cell parameters to the corresponding cell vectors for the *dl_poly.CONFIG* file. To use this feature, Option **25** in the control file must set to *auto*.

(b) ***MOLECULAR_GROUP*** directive

Atoms can be grouped by using the ***MOLECULAR_GROUP*** directive, which will also appear in the *dl_poly.FIELD* file. For example, consider a *xyz* molecular system consists of a cyclodextrin molecule and a phenylalanine molecule:

```
170
Cyclodextrin with PHE.
# MOLECULAR_GROUP CYC
C  -6.049000  0.945000  0.569000
C  -6.341000  2.134000  -0.313000
C  -5.113000  3.000000  -0.453000
C  -4.574000  3.388000  0.915000
…
…
# MOLECULAR_GROUP PHE
C  -0.830000  0.683000  2.322000
C  -0.114000  -0.185000  3.052000
C  0.873000  -0.882000  2.472000
C  1.110000  -0.714000  1.164000
C  0.405000  0.151000  0.413000
…
…
```

The 170-atom system is divided into two Molecular Groups: CYC and PHE. The TETHER and FREEZE features (Options **21** and **22** in the *control* file) can also be applied on atoms belong to respective Molecular Groups. Note that to apply bond constrains (Option **23**) in *xyz* systems, only the atom filter **h-bond** can be used, which applies constrains to all hydrogen-containing bonds within a Molecular Group.

For ORGANIC force fields, if no ***MOLECULAR_GROUP*** is specified, a default Molecular Group will be assigned, called 'XYZ', to the whole system.

For INORGANIC force fields, the directive ***MOLECULAR_GROUP*** must always be specified.

(c) The ***MOLECULE_KEY*** directive

This directive must be provided for INORGANIC *xyz* systems. It must be specified along with a ***MOLECULAR_GROUP*** and only one ***MOLECULE_KEY*** can be specified for each ***MOLECULAR_GROUP***. Consider a *xyz* configuration file as follows:

```
1056
CRYST1   19.952   24.790   79.262  90.00  90.00  78.39 P1          0
# MOLECULE_GROUP A1   MOLECULE_KEY CC1
O  4.337000  8.034000  23.538000
O  4.337000  16.128000  23.538000
O  5.639000  3.987000  23.538000
O  19.301000  8.034000  23.538000
O  19.301000  16.128000  23.538000
C  20.603000  3.987000  23.538000
O  14.313000  8.034000  23.538000
…
…
```

In this example, the Molecular Group A1 is defined, along with the MOLECULE_KEY CC1, which refers to a calcium carbonate model (see lib/*inorganic_ternary_oxide.sf*

file). The force field inorganic_ternary_oxide must also be specified in the *control* file (Option **1**).

(d) The ***POTENTIAL*** directive
This directive specifies the type of force field scheme, and it is only used in multiple potential models. The position of the directive indicates the range of application of the force field to atoms. Any atoms immediately after the directive would apply to the same force field type until another ***POTENTIAL*** directive is encountered. The ***POTENTIAL*** directives are automatically activated if the keyword 'multiple' is used in the *control* file (Option **1**). Otherwise, these directives will be ignored if a force field scheme is specified in the *control* file.

6.2.2 Special Note on xyz format

(a) For protein and DNA models based on CHARMM and AMBER FFs, users are advised to use the *PDB* format. There is no guarantee DL_FIELD can convert these structures correctly in the *xyz* format. This is because protein molecules are usually fitted to a specific set of parameters. For this reason, amino acid residues are still defined as MOLECULE templates.

(b) When noble gases are included in the *xyz* molecular systems, DL_FIELD assumes there is no bonding with rest of the atomic systems. The non-bonded interactions (vdw and Coulombic) are assumed to be the only interactions for noble gases.

(c) Models that are solvated with charged ions are considered to be non-bonded species. User must specify the charge state in the *xyz* file. For instance, a chloride anion will be expressed as '*Cl-*', rather than just '*Cl*' in the *xyz* file. In this case, DL_FIELD will automatically assign it as a chloride anion, rather than a normal chlorine atom. DL_FIELD will only consider the non-bonded interactions for charged ions, just like point (b).

Do not specify the charge state if a charged atom is expected to form bonds with other atoms. For example, specify an ammonium nitrogen simply as *N*, rather than *N+* in the *xyz* file.

(d) DL_FIELD can detect water molecules in user configuration files and will assign them as the TIP3P model by default. To solvate the system with other water models, use Option **31** in the *control* file.

6.2.3 Auto-identification of template-based FF systems

DL_FIELD allows user to express molecular system using the simple *xyz* format for template-based FF models. Traditionally, a PDB format would be needed, to specify explicitly the residue labels, to indicate which MOLECULE templates to use.

For instance, consider the following *xyz* structure:

```
21

H  0.721000  5.934000  0.953000
C  0.185000  5.739000  0.000000
O  -0.158000  4.366000  0.000000
H  0.666000  3.872000  0.000000
H  0.721000  5.934000  -0.953000
H  -0.770000  10.307000  0.000000
H  -0.770000  6.307000  0.000000
H  0.721000  9.934000  0.953000
C  0.185000  9.739000  0.000000
O  -0.158000  8.366000  0.000000
H  0.666000  7.872000  0.000000
H  0.721   9.934  -0.953
C   0.995   0.329  -0.000
H   1.844  -0.392   0.000
H   1.096   0.975  -0.902
H   1.096   0.976   0.901
C  -0.340  -0.404   0.000
H  -0.456  -1.038   0.907
H  -0.457  -1.039  -0.907
H  -2.235   0.073   0.000
O  -1.394   0.538  -0.000
```

By using, say, CHARMM36_CGENFF (CHARMM General Force Field), which is a template-based FF scheme, DL_FIELD will automatically group and identify the molecules in the file. After that, DL_FIELD will search the library (and *udff*) files to identify the suitable MOLECULE template. In this example, two types of molecules were detected: methanol and ethanol, and there are two methanol molecules and one ethanol molecule. This information is shown in the *dl_field.output* file as follows:

```
…
…
>>>>Reading coordinates in Molecular Group XYZ

The molecule has no cyclic structure.
ATOM_TYPEs mapping (DL_F notation): See output/dlf_notation.output file.

Total different type of molecules: 2

Molecule type 1
Total number: 2
Num of atoms in the molecule: 6
Molecular mass: 32.042780
Composition: C1 H4 O1

Molecule type 2
Total number: 1
Num of atoms in the molecule: 9
Molecular mass: 46.070220
Composition: C2 H6 O1

The following type of molecules have been identified:
Molecule type 1: methanol (MEOH)
Molecule type 2: ethanol (ETOH)
…
…
```

After that, usual FF conversions will be carried out, based on the template matching procedure as would be the case for a PDB file. If a suitable MOLECULE template cannot be located, then an error will be generated.

6.3 The Tripos *mol2* Format

Tripos mol2 file is a complete, portable representation of a SYBYL (SLN) molecule. Unlike the *PDB* format, *mol2* files are free format and can contain detailed information of molecular structures.

The *mol2* format is only applicable to the following ORGANIC force fields: OPLS2005, CVFF, PCFF and AMBER16_GAFF.

A typical *mol2* file consists of a series of *data records* classified according to the *Record Type Indicator* (*RTI*). Essentially, the *RTI*s divide a mol2 file into several sections. A section begins with a *RTI* and ends at the next *RTI* or the end of the file. All *RTI*s must always begin with the '@' at column 1. There are many *RTI*s but DL_FIELD only recognises a small set of 'essential' *RTI*s for the purposes of force field conversions. These *RTI*s are shown below:

@<TRIPOS>MOLECULE - information about a molecule.

@<TRIPOS>ATOM - information about an atom within a molecule.

@<TRIPOS>BOND - bond connectivity information of a molecule.

@<TRIPOS>CRYSIN - crystal cell parameters

```
@<TRIPOS>MOLECULE
aniline              (molecule name)
14 14 1 0 0          (number of atom, number of bonds, number of residues, 0, 0)
SMALL                (molecule type: SMALL, BIOPOLYMER, PROTEIN, etc)
USER_CHARGES  (charge type, or NO_CHARGES)
                     (leave blank!)
This is an example       (Comment, can leave blank)
@<TRIPOS>ATOM
    1 C       -1.1604   0.7227   -0.0729 C.ar     1    ****   0.0000  (atom index, atom label, x,y,z coordinates, atom_key,
    2 C       -1.1598  -0.6198   -0.0771 C.ar     1    ****   0.0000   residue id, residue key, charge values)
    3 C       -0.0005  -1.2919   -0.0018 C.ar     1    ****   0.0000
    4 C        1.1589  -0.6205    0.0779 C.ar     1    ****   0.0000
    5 C        1.1597   0.7220    0.0823 C.ar     1    ****   0.0000
    6 C       -0.0003   1.3949    0.0069 C.ar     1    ****   0.0000
    7 H       -2.1205   1.2629   -0.1353 H        1    ****   0.0000
    8 H       -2.1126  -1.1716   -0.1426 H        1    ****   0.0000
    9 H       -0.0007  -2.3948   -0.0053 H        1    ****   0.0000
   10 H        2.1116  -1.1729    0.1399 H        1    ****   0.0000
   11 H        2.1199   1.2616    0.1482 H        1    ****   0.0000
   12 N       -0.0001   2.6626    0.0110 N.3      1    ****   0.0000
   13 H        0.9052   3.1895    0.0732 H        1    ****   0.0000
   14 H       -0.9054   3.1901   -0.0479 H        1    ****   0.0000
@<TRIPOS>BOND
   1    1    2    2    (bond index, atom pair that formed a bond, bond type: 1=single, 2=double, 3=triple, am=amide, ar=aromatic)
   2    1    6    1
   3    1    7    1
   4    2    3    1
   5    2    8    1
   6    3    4    2
   7    3    9    1
   8    4    5    1
   9    4   10    1
  10    5    6    2
  11    5   11    1
  12    6   12    1
  13   12   13    1
  14   12   14    1
@<TRIPOS>CRYSIN
12.0  33.0  44.0   90.0  90.0  90.0    space_group   setting  (cell parameters: a, b, c, alpha, beta, gamma)
```

Example above shows a simple *mol2* file for an aniline molecule, indicating the meaning of each section (in red).

Special Note on mol2 format

(a) Unlike the *xyz* format, DL_FIELD does not recognise the directives such as ***MOLECULAR_GROUP***, ***MOLECULE_KEY*** and ***POTENTIAL***.

(b) If the atom keys in the *mol2* file match (by making references to the appropriate *.par* file) with that of the potential scheme specified in the DL_FIELD *control* file, then these will be treated as the ATOM_KEYs straight away without carrying out any procedure to determine the ATOM_TYPEs. The corresponding charge values will also be taken from the *mol2* file, if these were specified.

(c) If DL_FIELD fails to match the atom keys, then it will read the atom label in the *mol2* file and setup the force field model and carry out the auto-assignment of ATOM_TYPEs similar to that of the *xyz* format. In this case, the charge values will be determined in the usual way and will NOT extract from the *mol2* file. Note that DL_FIELD will make best guess to convert 'Atom label' to element symbols.

Beware: The matching of atom keys as mentioned in (c) and (d) is not fool-proof. For instance, PCFF and Amber16_GAFF share similar ATOM_KEYs for some of the ATOM_TYPEs. DL_FIELD may be fooled into thinking the atom keys in a *mol2* file is of PCFF which in fact is of Amber GAFF or *vice versa*.

7 The Output File (*dl_field.output*)

This is the output file where the conversion process of a user's atomic configuration is recorded. The output file is separated into several sections and the level of reporting details can be selected by switching the verbosity Option **7** in the *control* file to 1 (on) or 0 (off):

(1) CONTROL STATUS REPORT

Display options that have been selected in the *control* file.

(2) FORCE FIELD INITIALISATION

In this section, DL_FIELD reports the reading status of the *udff* (if requires), *.sf* and *.par* files. After that, computer memory is allocated to set up model structure databases.

(3) CONVERSION STATUS REPORT

This is the main section of *dl_field.output* which reports the conversion process of user configuration model into DL_POLY and GROMACS files.

Below shows a section of the conversion process of a protein (with Verbosity mode switch on):

```
*** Mapping 1. HIS ***
...
***** Need patching ******
Residue -HSE- of id = 1
*****Patch normal N-termination****

User config neighbour list:
1 --2  3  4  5
2 --1
3 --1
4 --1
5 --1  6  7  18
6 --5
…
17 --16
18 --5  19
19 --18
All atoms matched:  User atom index > matched atom labels
1 > n
2 > hn1
3 > hn2
4 > hn3
5 > ca
6 > ha
...
17 > h4
18 > c
19 > o
Total charge for residue 1 (-HSE-) is 1.000000.
```

In this example, the first residue is a histidine molecule. DL_FIELD will determine this charged species to be a HSE (CHARMM notation for a histidine protonated at the N3 position) with N-termination (-NH_3).

The 'User config neighbour list' shows the connectivity of all atoms where the numbers refer to the atom index number. For example, atom 1 is connected to atoms 2, 3, 4 and 5

The 'All atoms matched:' list identifies the user atoms with atom labels defined for the histidine molecule in the *.sf* file. In this example, it turns out that atom 1 is 'n'; atoms 2, 3 and 4 are 'hn' and atom 5 is 'ca'. The atom list will be reordered and assigned with the corresponding *atom keys*.

The neighbour list and information on matching atoms will not be displayed when the Verbosity mode is switched off.

Another example is given below. Here, PCFF potential scheme is used to convert a chlorobenzene (CBEN) structure.

```
*** Mapping 1. CBEN ***
User config neighbour list:
1 --7
2 --6
3 --4
4 --3 5 7
5 --4 6 12
6 --2 5 8
7 --1 4 9
8 --6 9 10
9 --7 8 11
10 --8
11 --9
12 --5

*** For Residue 1. CBEN
Following atoms are mismatched:
c2      h2      c6      h6

Might be due to symmetrical close-loop structure.
Attempt to break symmetry with nn=1
Reassign: 1 > h3
Reassign: 2 > h6
Reassign: 3 > h2
Reassign: 10 > h5
Reassign: 11 > h4

Okay, everything is fine now.

All atoms matched: User atom index > matched atom labels
1 > h3
2 > h6
3 > h2
4 > c2
5 > c1
6 > c6
7 > c3
8 > c5
9 > c4
10 > h5
11 > h4
12 > cl
```

In this example, DL_FIELD detects mismatch of atoms due to symmetrical nature of the molecule (the benzene group). Depending on the initial arrangement of atoms in user's configuration file, this may sometimes occur where atoms are assigned to correct ATOM_TYPEs but in the wrong positions. A second conversion attempt is made by introducing dummy atoms to break the symmetry, ensuring all atoms are correctly matched and in the right positions.

(4) LINKAGE INFORMATION BETWEEN RESIDUES

At this stage, all atoms within every residue have been matched and the force field parameters have been assigned. In this section, the DL_FIELD determines the linkage information between two neighbouring residues and this information is useful to check for any undiscovered/undesirable bonds between two atoms from different residues.

Once this information is obtained, DL_FIELD will determine force field parameters that describe the interactions between two connecting residues. Below shows a section of the output data (for a protein molecule):

```
...
Link pair: -thr- (3844) and -glu- (3846) of c-n  distance=1.331446
Link pair: -glu- (3859) and -cys- (3861) of c-n  distance=1.330719
Link pair: -glu- (3857) and -asp- (4011) of od1-oc2  distance=0.730184
Warning: od1-oc2 pair may not be a valid link for protein amino acid. It is not considered as a bond.
Link pair: -cys- (3869) and -cys- (3871) of c-n  distance=1.328854
Link pair: -cys- (3868) and -cys- (3977) of s-s  distance=2.025060
Link pair: -cys- (3879) and -hse- (3881) of c-n  distance=1.326833
...
```

In most cases, DL_FIELD will pick up the usual C-N amide linkages but occasionally the disulphide S-S linkages as well. The latter information is useful to indicate the **possible** S-S bond and, consequently, may have to delete the thiolate hydrogens in the user's PDB file and rerun DL_FIELD.

Occasionally, unintended atom distances may also be flagged up as a warning to indicate the potential source of problem during equilibration. For instance, the example above shows two oxygen atoms are too close to each other in -glu- (3857) and -asp- (4011). This warning only applies to protein molecules.

NOTE:

(1) Even if a S-S linkage is listed, DL_FIELD does not guarantee a bond is established. Though it is rare, this **will** happen if the macro definition S_S in the *dl_field.h* include file is smaller than the actual distance measured between the S-S linkage. If this happens, locate the program statement

#define S_S 2.135

in the *dl_field.h* file and increase the maximum bond length for S_S (2.135). Recompile DL_FIELD and rerun the conversion. Alternatively, select the Conversion criteria (Option **3**) in the *control* file to '*loose'* and run DL_FIELD again.

(2) DL_FIELD allows more than one linkage between a same pair of residues. However, conversion will fail if an atom from a residue has more than one link to other atoms from the other residues. This can be avoided by defining the two residues as a single MOLECULE.

8 Solution Maker

The Solution Maker allows users to setup a disordered system, be it a liquid or a solution (two-component system consists of solute and solvent molecules) model, based on a single molecule template input file. DL_FIELD will then setup the force field model in a single step.

The Solution Maker is located at Option **10** in the *control* file. The required parameters are shown below as an example:

*0 0.94 g/cm^3 1.9 * Solution Maker: on/off, density, unit, cut off)*

The first value is the switch which takes two values: off (*0*) and on (*1*). The second parameter dictates the number of molecules in the system, depending on the unit use in the third parameter and the simulation box size specified (it must be an orthogonal system). The fourth parameter, in unit Ångström, indicates the minimum distance between the molecules.

The third parameter allows the following units - density: *g/cm^3* or *kg/m^3*, concentration: *mol/dm^3*, or *molecule*, which simply indicates the number of molecules require in the system.

For example, to setup an ethanoic acid liquid force field model, an input molecular structure (either in the PDB or xyz format) contained only a single ethanoic acid molecule is needed which acts as a template. Then, the following options may be used for the Solution Maker. For instance, the following parameters were used:

1 1.05 g/cm^3 1.8

DL_FIELD will then automatically duplicate a number of ethanoic acid molecules require base on the input unit (g/cm^3) and cell vectors (Option **26**). The molecules are systematically added by randomly orientate the molecular template before it is placed in the system. The centre of gravity of each molecule is located approximately around some imaginary grid points in succession. These grid points were pre-determined based on the simulation box size and the size of the molecular template. After that, DL_FIELD will set up the force field model and generate the require output files, *dl_poly.FIELD* and *dl_poly.CONFIG*.

If DL_FIELD fails to insert all the molecules, reduce the fourth parameter (1.8) to some lower value and try again. This enable DL_FIELD to pack more molecules in the system, as the critical distance between molecules is reduced. Obviously, doing so would create a high-energy system which would need careful equilibration. Alternatively, if the input molecular structure is flexible, perhaps try to use different conformations.

The Solution Maker can be used to create a two-component system, such as a solution model. For example, to setup an ethanoic acid solution of concentration 0.1 M with water as solvent, the following command can be used:

1 0.1 mol/dm^3 1.9

A suitable solvent type, such as a water model, can be selected, in Option **31** in the *control* file.

User can also specify a fixed number of molecules to be inserted. For example:

1 15 molecules 6.4

This instructs DL_FIELD to duplicate and insert 15 molecules, each is at least 6.4 Å apart from one another in the system. The extent of molecular separation can be increased by increasing the cut off value. Obviously, this is only feasible for dilute systems. Highly concentrated systems or pure liquid systems would naturally need a smaller cut off value to pack more molecules in the simulation box.

To create a truly disordered system, further equilibration is needed and can be achieved in the MD program.

8.1 Water Models

The following water models are available in most *.sf* files and the structure templates are located in the *solvent/* directory.

(1) Original TIP3P. Key: *TP3O* (*water_tip3p*)
W. L. Jorgensen, J. Chandrasekhar and J. D. Madura, 'Comparison of simple potential functions for simulating liquid water', *J. Chem. Phys*. **79**, 926-935 (1983)

(2) Modified TIP3P for CHARMM. Key: *TIP3* (*water_tip3p_c*)
This is fitted using CHARMM-type atom-based, force-shifted for long-range electrostatic calculations. Can be implemented as both constrained (use **rigid_water** *filter_atom* key) or flexible models.
Note: The CHARMM molecular models were fitted to the rigid version of the modified TIP3P model.

(3) Modified TIP3P with particle mesh ewald (PME) summation. Key: *TP3E* (*water_tip3p_pme*)
This is fitted using PME long-range electrostatic calculations that does not include Lennard-Jones parameters on the hydrogen atoms. The version DL_FIELD uses is derived from Model B of D.J. Price and C. L. Brooks III, *J. Chem. Phys.* **121**, 10096 (2004).
Can only be implemented as a constrained (use **rigid_water** *filter_atom* key) model or as a rigid model using the [***RIGID***] directive.

(4) Original SPC water (*SPC*) and SPC/E water (*SPCE*)
Three-site water model and SPC/E includes corrections for self-polarization and improved structural and diffusion properties.
W. L. Jorgensen, J. Chandrasekhar and J. D. Madura, 'Comparison of simple potential functions for simulating liquid water', *J. Chem. Phys*. **79**, 926-935 (1983)
H.J.C. Berendsen, J. R. Grigera and T.P. Straatsma, *J. Phys. Chem.* **91**, 6269 (1987)

(5) TIP4P water (*TIP4*)
A four-site water model with an off-center point charge for oxygen. Can only be implemented as rigid model using the [***RIGID***] directive.
W. L. Jorgensen, J. Chandrasekhar and J. D. Madura, 'Comparison of simple potential functions for simulating liquid water', *J. Chem. Phys*. **79**, 926-935 (1983)

(6) TIP5P water (TIP5)
A five-site water model with two pseudo points representing the lone pairs on the oxygen atom. Can only be implemented as a rigid model using the [***RIGID***] directive.
M.W. Mahoney and W.L. Jorgensen, *J. Chem. Phys.* **112**, 8910 (2000).

(7) TIP5P water for PME calculations (TP5E)
Slight modification to the original TIP5P for PME calculations.

S. W. Rick, *J. Chem. Phys.* **120**, 6085 (2004)

Note that use of modified TIP3P (*TIP3*) for CHARMM is usually recommended. However, inclusion of other water models in the *.sf* file does not mean the endorsement of DL_FIELD's author to use other water models. They are included as alternative choices at the user's discretion.

However, there are some works being carried out to investigate the structural and dynamical effects on molecules using different water models. User is recommended to consult these papers:

P. Mark and L. Nilsson, 'Molecular dynamics simulations of the Ala-Pro dipeptide in water: Conformational dynamics of trans and cis isomers using different water models', *J. Phys. Chem. B* **105**, 8028-8035 (2001)

P. Mark and L. Nilsson, 'A molecular dynamics study of tryptophan in water', *J. Phys. Chem. B* **106**, 9440-9445 (2002)

D. C. Glass, M. Krishnan, D. R. Nutt and J. C. Smith, 'Temperature dependence of protein dynamics simulated with three different water models', *J. Chem. Theory Comput.* **6**, 1390-1400 (2010)

9 Inorganic Force Fields

Despite the inherent differences between organic and inorganic materials in terms of their chemical nature, the force field libraries for inorganic systems have the same format as those of organic systems. In other words, all standard libraries share the same ***DIRECTIVE***s.

The connectivity for all ATOMs in inorganic MOLECULEs must be defined as either the self-CONNECT type or the auto-CONNECT type. The normal CONNECT type is not available in the INORGANIC force field. Example below shows the definition of the core-shell model of a magnesium oxide MOLECULE.

```
ATOM_TYPE       key    element  mass      remark
…
Mg2+#6c_binary_oxide1  Mg1c    Mg      24.3050   Ref. 1  Mg core
Mg2+#6s_binary_oxide1  Mg1s    Mg       0.0000   Ref. 1  Mg shell
O2-#8c_binary_oxide1   O1c     O       15.9994   Ref. 1  O core, on Li2O, Na2O etc
O2-#8s_binary_oxide1   O1s     O        0.0000   Ref. 1  O shell on Li2O, Na2O etc
…
END ATOM_TYPE

MOLECULE_TYPE
…
magnesium_oxide1        MO1     40.3044   Ref. 1, MgO cubic
…
END MOLECULE_TYPE

MOLECULE magnesium_oxide1 4 0.0        Ref. 1
Mg1c  Mg2+#6c_binary_oxide1    1.580
Mg1s  Mg2+#6s_binary_oxide1    0.420
O2c   O2-#6c_binary_oxide1     0.513
O2s   O2-#6s_binary_oxide1    -2.513
CONNECT Mg1c > Mg1c
CONNECT Mg1s > Mg1s
CONNECT O2c  > O2c
CONNECT O2s  > O2s
SHELL Mg1c  Mg1s
SHELL O2c   O2s
END MOLECULE
```

The self-CONNECT type means that the atom labels in the PDB file must also be the same as those of ATOM_KEYs defined in the MOLECULE. Note that, unlike the MOLECULEs defined for the ORGANIC force fields, the number of ATOMs in the PDB file does not have to correspond to that defined in the ***MOLECULE*** directive. An example PDB file for the MgO MOLECULE is shown below.

```
HETATM    1 Mg1c MO1     1       0.000   0.000   0.000                      Mg
HETATM    1 Mg1s MO1     1       0.000   0.000   0.000                      Mg
HETATM    2 O2c  MO1     1       0.000   2.104   0.000                      O
HETATM    3 Mg1c MO1     1       0.000   4.208   0.000                      Mg
HETATM    3 Mg1s MO1     1       0.000   4.208   0.000                      Mg
HETATM    4 O2c  MO1     1       0.000   6.312   0.000                      O
HETATM    4 O2s  MO1     1       0.000   6.312   0.000                      O
HETATM    5 O2c  MO1     1       2.104   0.000   0.0                        O
HETATM    5 O2s  MO1     1       2.104   0.000   0.0                        O
HETATM    6 Mg1c MO1     1       2.104   2.104   0.0                        Mg
HETATM    6 Mg1s MO1     1       2.104   2.104   0.0                        Mg
HETATM    7 O2c  MO1     1       2.104   4.208   0.0                        O
…
…
```

The core and shell components of a polarisable atom is usually located very close to each other and both can be exactly at the same location in the initial model set up. DL_FIELD will automatically match up these components, provided the core-shell distances are less than 0.5 Å. The MgO shell model example is in the *Examples/* directory with the filename *mgo_shell.pdb*.

If no shell or core components are given, DL_FIELD will automatically add and define the missing components in DL_POLY and GROMACS files (see Section 9.2).

Unlike ORGANIC force fields, which uses LJ functions by default, the INORGANIC force fields can use different functional forms for the vdw interactions. The vdw parameters in the INORGANIC force fields are defined in pairs of ATOM_KEYs as follows:

KEY*1* KEY*2* *functional_form* *parameter_1* *parameter_2* *paremeter_3* *Remark*

The available *functional form*s are as follows:

Lennard-Jones 12-6: $V(r_{ij}) = \frac{A}{r_{ij}^{12}} - \frac{B}{r_{ij}^{6}}$

Morse: $V(r_{ij}) = D_e[1 - e^{-\alpha(r_{ij}-R)}]^2$

Buckingham: $E(r_{ij}) = A_{ij}\exp(-\frac{r_{ij}}{\rho}) - \frac{C_{ij}}{r_{ij}^6}$

General LJ n-m: $V(r_{ij}) = \frac{\varepsilon}{n-m}\left[m\left(\frac{R}{r_{ij}}\right)^n - n\left(\frac{R}{r_{ij}}\right)^m\right]$

The vdw *parameters* must be defined in the same order as that of DL_POLY. For *functional form*s with only two parameters such as the Lennard-Jones 12-6, *parameter_3* must be set to zero (0.0). Below shows an example list vdw parameters, extracted from the *DLPOLY_INORGANIC.par* file.

```
VDW        type        parameters
Li1  O1s   buckingham   426.480   0.300   0.0    ! Ref. 1 Li2O
Na1  O1s   buckingham   1271.504  0.300   0.0    ! Ref. 1 Na2O
K1   O1s   buckingham   3587.570  0.300   0.0    ! Ref. 1 K2O
Mg1s O2s   buckingham   2457.243  0.261   0.0    ! Ref. 1 Mg shell - O shell
Mg2s O5s   buckingham   821.600   0.3242  0.0    ! Ref. 2 set 2, Table 3
Mg3  O10   buckingham   1152.000  0.3065  0.0    ! Ref. 7 Mg-O rigid ion
…
END VDW
```

All inorganic force field models are classified according to the type of materials in different library files. This allows user to locate and identify the appropriate models to use more efficiently. The available material classifications are shown below.

(a) Binary oxides (*INORGANIC_binary_oxides*)
This class of materials consist exclusively two types of elements, one of which must be the oxygen. The class has a general formula of M_xO_y. Examples of binary oxides including mineral oxides, metal oxides and gas oxides such as carbon dioxide, sulphur dioxide, etc.

(b) Binary halides (*INORGANIC_binary_halides*)
This class of materials consist exclusively two types of elements, one of which must be a halide (F, Cl, Br or I). The class has a general formula of M_xX_y, where X is a halide.

(c) Ternary oxides (INORGANIC_ternary_oxides)
This class of materials consist exclusively three types of elements, one of which must be the oxygen. The class has a general formula of $M_xX_yO_z$. Example of such including carbonate minerals such as aragonite ($CaCO_3$).

(d) Clay minerals (***INORGANIC_clay***)
The inorganic force field including CLAYFF for modelling clay minerals. This is defined in ***MOLECULE*** general_CLAYFF with the corresponding MOLECULE_KEY of CLYF. All ATOMs are defined as auto-CONNECT that allows variable number of neighbours for most ATOM_KEYs.

ANGLE ONLY and ***BOND ONLY*** directives are used to selectively define the necessary bonds and angles depending on the neighbouring atoms in the system.

To use this force field, the CLYF residue label must be inserted in the PDB file or using the directive ***MOLECULE_KEY*** CLYF in the *xyz* file, and all atoms must be labelled with the correct ATOM_KEYs.

The force field scheme also contains a set of generic Lennard-Jones vdw force field, which can be mixed with the vdw parameters from other force field schemes including those of ORGANIC types.

(e) Glassy materials (**INORGANIC_glass**)
These materials consist of a random mixture of ionic species such as metal ions, silicates and phosphates, etc. Some of these models also use three-body interactions.

(f) Zeolites (**INORGANIC_zeolite**)
These are highly ordered rigid materials, consist of pores and channels of various sizes. Some of these models also use three-body interactions.

(g) Hill-Sauer force field for alumino-silicates (INORGANIC_zeolite_hs)
This is a consistent force field for protonated aluminosilicates including zeolites, hydrated silica and aluminasilicic acids.

Inorganic systems can also be expressed in the *xyz* format. In this case, Molecular Groups and the MOLECULE_KEYs can be defined by inserting the following directives in the configuration file:

MOLECULAR_GROUP something ***MOLECULE_KEY*** MO3

For more details about inorganic systems in *xyz* format, please see Section 6.2.

9.1 Auto setup of core-shell systems.

DL_FIELD can automatically detect the type of MOLECULE template and insert core (or shell) component if needed. To use this feature, a 'clean' configuration file would be needed, which only contains the usual atomic species with no or incomplete core or shell element.

When the program is run, DL_FIELD will look for the ***MOLECULE_KEY*** and decide if it is a rigid ion or a core-shell model. If the latter is used, DL_FIELD will automatically identify

the missing components and automatically add these into *dl_poly.CONFIG* and *dl_poly.FIELD* files.

10 Multiple Potential Systems

DL_FIELD allows different potential schemes apply on different parts of a system model. This type of model is called the multiple potential system. The multiple potential feature can be implemented in both PDB and *xyz* files.

The word 'multiple' must be used in Option **1** of the *control* file, to instruct DL_FIELD to look for the potential scheme definitions, which is indicated by the ***POTENTIAL*** directive in the configuration file.

10.1 Implementation for PDB files

The example below illustrates the use of three different potential schemes on a system model consists purely of methanol molecules (extracted from the *multiple_potential_pdb_1.pdb* file from the *Examples/* directory).

```
REMARK  POTENTIAL AMBER
HETATM   1 C   MEOH   1     0.185   0.739  -0.000          GRP1 C
HETATM   2 H   MEOH   1    -0.770   1.307   0.000          GRP1 H
HETATM   3 H   MEOH   1     0.721   0.934  -0.953          GRP1 H
HETATM   4 H   MEOH   1     0.721   0.934   0.953          GRP1 H
HETATM   5 O   MEOH   1    -0.158  -0.634  -0.000          GRP1 O
HETATM   6 H   MEOH   1     0.666  -1.128  -0.000          GRP1 H
REMARK POTENTIAL OPLS2005
HETATM   8 H   MEOH   1    -0.770   6.307   0.000          GRP2 H
HETATM  10 H   MEOH   1     0.721   5.934   0.953          GRP2 H
HETATM   7 C   MEOH   1     0.185   5.739  -0.000          GRP2 C
HETATM  11 O   MEOH   1    -0.158   4.366  -0.000          GRP2 O
HETATM  12 H   MEOH   1     0.666   3.872  -0.000          GRP2 H
HETATM   9 H   MEOH   1     0.721   5.934  -0.953          GRP2 H
HETATM   8 H   MEOH   2    -0.770  10.307   0.000          GRP2 H
HETATM  10 H   MEOH   2     0.721   9.934   0.953          GRP2 H
HETATM   7 C   MEOH   2     0.185   9.739  -0.000          GRP2 C
HETATM  11 O   MEOH   2    -0.158   8.366  -0.000          GRP2 O
HETATM  12 H   MEOH   2     0.666   7.872  -0.000          GRP2 H
HETATM   9 H   MEOH   2     0.721   9.934  -0.953          GRP2 H
HETATM   1 C   MEOH   3     0.185  -3.261  -0.000          GRP3 C
HETATM   2 H   MEOH   3    -0.770  -2.693   0.000          GRP3 H
HETATM   3 H   MEOH   3     0.721  -3.066  -0.953          GRP3 H
HETATM   4 H   MEOH   3     0.721  -3.066   0.953          GRP3 H
HETATM   5 O   MEOH   3    -0.158  -4.634  -0.000          GRP3 O
HETATM   6 H   MEOH   3     0.666  -5.128  -0.000          GRP3 H
REMARK  POTENTIAL CHARMM22_prot
HETATM   8 H   MEOH   1     3.230   6.307   0.000          GRP4 H
HETATM  10 H   MEOH   1     4.721   5.934   0.953          GRP4 H
HETATM   7 C   MEOH   1     4.185   5.739  -0.000          GRP4 C
HETATM  11 O   MEOH   1     3.842   4.366  -0.000          GRP4 O
HETATM  12 H   MEOH   1     4.666   3.872  -0.000          GRP4 H
HETATM   9 H   MEOH   1     4.721   5.934  -0.953          GRP4 H
END
```

The example model basically consists of five methanol molecules. The first methanol is assigned to the AMBER FF; the following three to the OPLS2005 FF and the last molecule is assigned to the CHARMM22_prot FF.

Note that the example given above is just an illustration. Notice the ease and flexibility of applying multiple-potential schemes in a molecular model. Such a mixed FF model is obviously nonsensical!

The location of the ***POTENTIAL*** directive determines the extent of the FF application. For instance, if the statement ***POTENTIAL*** OPLS2005 is removed, then the first four molecules will be assigned to the AMBER FF and the last molecule to the CHARMM22_prot.

It is advisable to place the ***POTENTIAL*** directive statements within the PDB's REMARK statement. This is to ensure the directive statements do not interfere with the PDB file structure and can still be viewed using some standard molecular viewer programs.

Note that a ***POTENTIAL*** directive statement can only be placed **at the beginning** of a Molecular Group. For instance, the Molecular Group GRP2 as shown above contains two methanol molecules. Placing the ***POTENTIAL*** OPLS2005 statement somewhere in the middle of GRP2 scope will generate an error.

In a multiple potential scheme, all structures will be converted and assigned with the appropriate parameters according to the rules specific to the force field scheme. To distinguish atoms that belong to different force field schemes, a unique suffix will be appended to the ATOM_KEYs as shown below:

X_$

The X is the ATOM_KEY and '$' is the unique label assign to a given FF scheme according to the order of appearance of each scheme in a user model. This label basically consists of alphabets 'A', 'B', 'C', etc.

For instance, from the given previous example, atoms that belong to ***POTENTIAL*** AMBER, OPLS2005 and CHARMM22_prot will be assigned with the _A, _B and _C suffixes respectively. In this case, the primary alkane carbon in AMBER is described as CT_A, where CT is the standard AMBER key. For CHARMM22_prot, the same carbon atom is described as CT3_C, where CT3 is the standard CHARMM key.

10.2 Implementation for *xyz* Files

Molecular Groups can naturally be defined within a PDB file within the appropriate columns (See Section 6.1). For *xyz* file, the ***MOLECULAR_GROUP*** directive would be needed. For example, consider the *multiple_potential_3.xyz* file in the *Examples/* directory, of which the relevant parts are shown below:

The example below shows several methanol molecules classified into four groups, GRP1, GRP2, GRP3 and GRP4. If the 'multiple' keyword is used in the control file, DL_FIELD will setup a multiple potential model that contains CVFF and OPLS2005 force fields. Within the CVFF force field, atoms will be classified into GRP1, GRP2 and GRP3, while the ATOM_KEYs will be assigned with a suffix _A. The OPLS2005 force field will be applied to the last molecule with a suffix _B and is associated with the Molecular Group GRP4.

```
30
Multiple potential schemes for methanol molecules
# MOLECULAR_GROUP GRP1   POTENTIAL cvff
C  0.185000  0.739000  0.000000
H  -0.770000  1.307000  0.000000
H  0.721000  0.934000  -0.953000
H  0.721000  0.934000  0.953000
O  -0.158000  -0.634000  0.000000
H  0.666000  -1.128000  0.000000
# MOLECULAR_GROUP GRP2
H  -0.770000  6.307000  0.000000
H  0.721000  5.934000  0.953000
…
…
…
O  -0.158000  8.366000  0.000000
H  0.666000  7.872000  0.000000
H  0.721000  9.934000  -0.953000
# MOLECULAR_GROUP GRP3
C  0.185000  -3.261000  0.000000
H  -0.770000  -2.693000  0.000000
H  0.721000  -3.066000  -0.953000
H  0.721000  -3.066000  0.953000
O  -0.158000  -4.634000  0.000000
H  0.666000  -5.128000  0.000000
#  POTENTIAL opls2005 MOLECULAR_GROUP GRP4
H  3.230000  6.307000  0.000000
H  4.721000  5.934000  0.953000
C  4.185000  5.739000  0.000000
O  3.842000  4.366000  0.000000
H  4.666000  3.872000  0.000000
H  4.721000  5.934000  -0.953000
```

For INORGANIC force fields, both ***MOLECULAR_GROUP*** and ***MOLECULE_KEY*** directives (see Section 6.2) must be defined on the same statement line in the *xyz* file. For example, consider a mixed organic-inorganic system such as the *multiple_potential_4.xyz* file in the *Examples/* directory:

```
33
This is title section
# POTENTIAL OPLS2005  MOLECULAR_GROUP A1
C   0.995000  0.329000  5.000000
H    1.844000  -0.392000  5.000000
H    1.096000  0.975000  4.098000
H    1.096000  0.976000  5.901000
C   -0.340000  -0.404000  5.000000
H    -0.456000  -1.038000  5.907000
H    -0.457000  -1.039000  4.093000
H    -2.235000  0.073000  5.000000
O   -1.394000  0.538000  5.000000
# POTENTIAL inorganic_binary_oxide   MOLECULE_KEY MO3  MOLECULAR_GROUP A2
O    0.000000  6.312000  0.000000
Mg   0.000000  0.000000  0.000000
O    0.000000  2.104000  0.000000
Mg   0.000000  4.208000  0.000000
O    2.104000  0.000000  0.000000
Mg   2.104000  2.104000  0.000000
O    2.104000  4.208000  0.000000
…
…
```

The example above shows a simple system consists of an ethanol molecule (organic) and magnesium oxide (inorganic). For the organic part, the ***MOLECULE_KEY*** directive is not required. DL_FIELD will automatically determine the chemical structure base on the element symbols and assign the force field parameters automatically for the OPLS2005 force field.

For the inorganic part, the ***MOLECULE_KEY*** directive must be defined, along with the ***MOLECULAR_GROUP***. The ***MOLECULE_KEY*** directive indicates which MOLECULE template to look for by the DL_FIELD program. Unlike the organic systems, only one ***MOLECULE_KEY*** would be allowed for a Molecular Group. In contrast to the PDB format, normal element symbols are allowed and DL_FIELD will attempt to determine the corresponding ATOM_KEYs. Alternatively, DL_FIELD can also recognise the actual ATOM_KEYs, instead of just element symbols. For instance, consider a clay mineral with an organic molecule (benzene) as shown below:

In this example, the organic part would still be determined by DL_FIELD and assign CVFF

```
451
CRYST1   83.908   20.7302253   17.9859383759  90.00  90.00  90.00 (P 1)
# POTENTIAL CVFF  MOLECULAR_GROUP ORG1
C   10.000000  1.390000  0.000000
C   10.000000  0.695000  1.204000
C   10.000000  -0.695000  1.204000
C   10.000000  -1.391000  0.000000
C   10.000000  -0.695000  -1.204000
C   10.000000  0.695000  -1.204000
H   10.000000  2.450000  0.000000
H   10.000000  1.225000  2.123000
H   10.000000  -1.226000  2.122000
H   10.000000  -2.451000  0.000000
H   10.010000  -1.226000  -2.122000
H   10.000000  1.225000  -2.122000
# POTENTIAL inorganic_clay   MOLECULAR_GROUP CLY   MOLECULE_KEY CLYF
st  -2.675000  -6.024000  4.703000
st  -2.700000  -6.039000  -4.313000
st  -2.776000  -3.444000  -8.761000
ob  -3.282000  -7.370000  -7.946000
ob  -3.236000  -7.348000  0.934000
ob  -3.229000  -2.154000  0.953000
…
…
```

parameters. For the inorganic part, the actual ATOM_KEYs are used for CLAYFF (st, ob, etc). DL_FIELD recognises this automatically and assign the parameters accordingly.

10.3 Mixing Lennard-Jones 12-6 Potentials

DL_FIELD can mix 12-6 Lennard-Jones potentials for a given non-bonded atom pair belong to different FF schemes. The mixing rule for epsilon and sigma can be independently specified in Option **18** and Option **19** respectively. See Section 3.10.1 for the available mixing schemes.

The mixing of LJ parameters can be applied for FF schemes that use the 12-6 functional forms. This means most of the ORGANIC FF schemes and a few INORGANIC ones such as the CLAYFF (clay force field).

If a non-LJ 12-6 form is used, then DL_FIELD can transform this to the LJ 12-6 form as follows:

10.4 Mixing Lennard-Jones 9-6 Potentials

PCFF and COMPASS use the LJ 9-3 form. In this case, if PCFF and COMPASS are present in the system, the vdw parameters will be mixed according to the 6th order Waldman-Hagler mixing rules*:

$$\varepsilon = 2\sqrt{\varepsilon_i \varepsilon_j}\left(\frac{R_i^3 R_j^3}{R_i^6 + R_j^6}\right) \qquad R = \left[\frac{\left(R_i^6 + R_j^6\right)}{2}\right]^{\frac{1}{6}}$$

Where R_i and R_j are the minimum equilibrium distances for atom *i* and atom *j*, respectively with one from PCFF and the other from COMPASS. Note: The vdw mixing rules in Option **18** and Option **19** only applies to 12-6 functional forms.

* M. Waldman and A.T. *Hagler, J. Comput. Chem.* **14**, 1077 (1993)

10.5 Mixing Lennard-Jones 9-6 and 12-6 Potentials

To blend between LJ 9-6 and LJ 12-6 in a multiple potential setting, DL_FIELD can transform LJ 9-6 to LJ 12-6 approximate form. From such, the vdw parameters for mixed potential schemes can be derived based on the mixing rules specify in Option **18** and Option **19**. Obviously, this sort of transformation is only an approximation.

Same as the 12-6 form, the 9-6 form can be expressed in terms of equilibrium distance, *R* or the steric parameter, σ.

$$V(r_{ij}) = \varepsilon\left[2\left(\frac{R}{r_{ij}}\right)^9 - 3\left(\frac{R}{r_{ij}}\right)^6\right]$$

$$V(r_{ij}) = 6.75\varepsilon\left[\left(\frac{\sigma}{r_{ij}}\right)^9 - \left(\frac{\sigma}{r_{ij}}\right)^6\right]$$

with the following relationship:

$$R = 1.5^{\frac{1}{3}}\sigma \approx 1.144714\sigma$$

The corresponding LJ 12-6 forms would be expressed as follows:

$$V(r_{ij}) = \epsilon\left[\left(\frac{R}{r_{ij}}\right)^{12} - 2\left(\frac{R}{r_{ij}}\right)^6\right]$$

$$V(r_{ij}) = 4\varepsilon\left[\left(\frac{\sigma}{r_{ij}}\right)^{12} - \left(\frac{\sigma}{r_{ij}}\right)^6\right]$$

With the following relationship:

$$R = 2^{\frac{1}{6}}\sigma \approx 1.122462\sigma$$

If the values of *R* and ε are transferred, as is, from LJ 9-6 to LJ 12-6, this will underestimate the density and surface energy would decrease. This is because the 9-6 form is softer on

the repulsive part and more attractive at the dispersive part when compare with the 12-6 form.

By starting from a given set of 9-6 parameters, DL_FIELD will fit numerically to derive the equivalent 12-6 parameters base on same area between the functions, over the interatomic distance from σ to 15 Å. The justification to do so is described below.

For example, consider the aromatic carbon in COMPASS FF of atom type c3a with R = 3.915 Å and ε = 0.068 kcal/mol for LJ 9-6 functions. Graph below shows the corresponding LJ functions for the 9-6, fitted 12-6 and unfitted 12-6 (using the 9-6 parameters, as is).

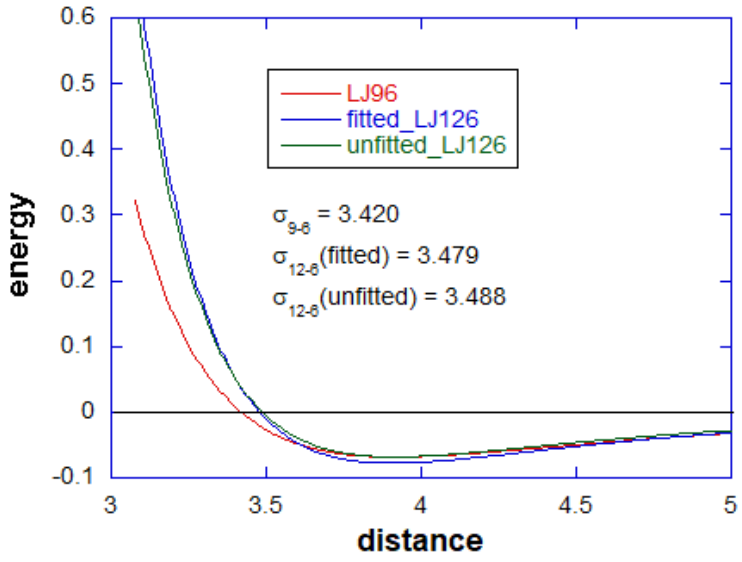


Table below shows the coefficient of determinations (goodness of fits) of LJ functions. The unfitted LJ 12-6* (shown in graph above) used similar R and ε of the LJ 9-6, while the unfitted LJ 12-6** used the identical σ and ε values of LJ 9-6.

	Fitted LJ 12-6	Unfitted LJ 12-6*	Unfitted LJ 12-6**
Short range (0.9σ to σ)	-3.134618	-2.030590	0.639511
Near equilibrium (σ to 2σ)	0.923565	0.915493	0.941755
Long range (2σ to 15 Å)	0.948703	0.884641	0.805844
Overall (σ to 15 Å)	0.969042	0.965565	0.976322

As expected, the LJ 12-6 functions do not fit well relative to LJ 9-6 at the repulsive range, due to softness of the latter function. The fitting at the short range is vastly improved if the LJ 12-6 is forced to adopt the identical σ value (LJ 12-6**). However, this results in the change of value of R due to differing relationships for LJ 9-6 and LJ 12-6 as shown above. For this reason, this would sacrifice the accuracy over longer range, with the fitting coefficient reduces to about 0.8058.

For the fitted LJ 12-6 parameters, ε is 0.076 kcal/mol, which is slightly stronger to compensate for the more repulsive part of the 12-6 potential. At the same time, the fitted R value is 3.905 Å which is quite similar to that of LJ 9-6 (3.915 Å). For this reason, it gives better fit to the dispersive part over longer range (2σ to 15 Å).

The fitting outcomes will be shown in *dl_field.output* if the multiple potential systems involve the mixing of the LJ 9-6 and 12-6 functions and Option **7** (verbosity mode) is switched on.

Note that DL_FIELD only provides a mean to automatically refit the parameters. It is still up to the user to test the validity of such mixing, as would be the case for all vdw mixing in a mixed potential system.

10.5 Mixing Morse and LJ 12-6 Potentials

The Morse potential is a better approximation for covalent bonds than those of harmonics, where the Morse function can account for the anharmonicity of a real bond. However, the Morse potentials have also been used to model non-bonded interactions.

The Morse function is expressed as:

$$V(r) = D_e\left(1 - e^{-\lambda}\right)^2$$

$$\text{Where } \lambda = \alpha(r - R)$$

Here, D_e is the well-depth, R is the equilibrium distance and α is a stretching constant that associates with the width of the potential. The expression above shows that $V(r) = 0$ at R, or zero potential energy at minimum. For this reason, Morse potentials are usually redefined with a negative well-depth D_e

$$V_{mor}(r) = D_e\left(1 - e^{-\lambda}\right)^2 - D_e = D_e(e^{-2\lambda} - 2e^{-\lambda})$$

The steric constant, σ_{mor}, where $V_{mor}(r)$ changes sign, is given as

$$\sigma_{mor} = \frac{1}{\alpha}\ln\left(\frac{1}{2}\right) + R$$

If R is equivalent for both Morse and LJ 12-6, and $\sigma_{mor} = \sigma_{LJ}$ then this gives the relationship between α and R as follows:

$$\alpha R = \frac{2^{\frac{1}{6}}\ln\left(\frac{1}{2}\right)}{\left(1 - 2^{\frac{1}{6}}\right)} \approx 6.3532$$

Furthermore, if both Morse and LJ 12-6 have similar well-depth with $D_e = \varepsilon = D$ at R, then, at the minimum,

$$\left(\frac{\partial V_{mor}}{\partial r}\right)_{r=R} = \left(\frac{\partial V_{LJ}}{\partial r}\right)_{r=R} = 0$$

and the second derivatives of these functions are given as follows:

$$\frac{\partial^2 V_{mor}}{\partial r^2} = 2D\alpha^2(2e^{-2\lambda} - e^{-\lambda})$$

$$\frac{\partial^2 V_{LJ}}{\partial r^2} = D\left(\frac{156R^{12}}{r^{14}} - \frac{84R^6}{r^8}\right)$$

By assuming the curvature behaviour is the same at the minimum, then

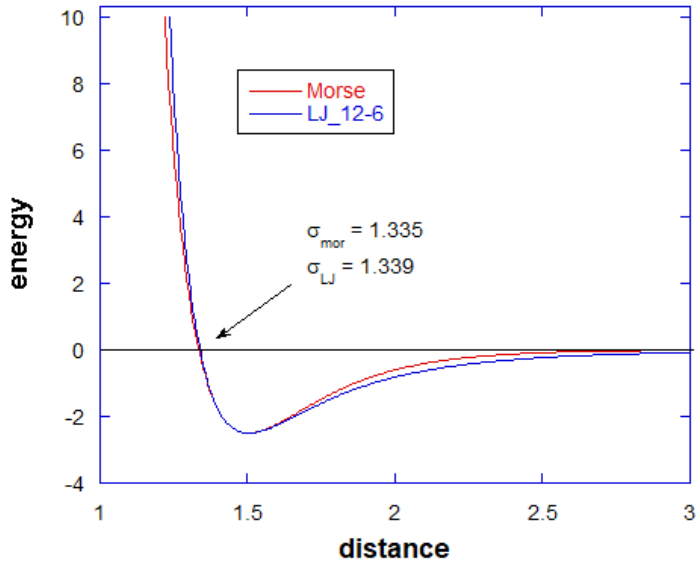
$$\left(\frac{\partial^2 V_{mor}}{\partial r^2}\right)_{r=R} = \left(\frac{\partial^2 V_{LJ}}{\partial r^2}\right)_{r=R}$$

Which then gives another relationship between α and R as follows:

$$\alpha R = 6$$

This means it is possible for a Morse potential to obtain a good fit to a corresponding LJ 12-6 at small r by using the parameters R and D_e, as they are, if the relationships between α and R as shown above are approximately fulfilled.

For example, consider the following parameters for a Morse function: R = 1.503 Å, D_e=2.5 kcal/mol and α = 4.125. This gives αR = 6.2. Graph below shows the plots for the Morse function and the corresponding LJ 126 function.



As expected, the LJ 12-6 fits quite well near to the equilibrium down to around σ. The notable difference is that, at the longer range, the LJ 12-6 over-estimates the attractive part of the energy. Fitting to these ranges would entails the change in well-depth and even R, which may result in a much poorer fit to the short distances.

Obviously, there is no 'perfect fit' between the functions. A warning will be given by DL_FIELD if the quantity αR is not within the range $5.8 \leq \alpha R \leq 6.5$. Once again, goodness of fits coefficients will be provided as a gauge to the quality of the LJ 12-6 function obtained.

Table below shows the goodness of fits for the above graph over several regions (with respect to σ_{mor}).

	LJ 12-6
Short range (0.95σ to σ)	0.728044
Around equilibrium (σ to 1.2R)	0.981387
Intermediate range (1.2R to 2.5σ)	0.759713

Overall (σ to 15 Å)	0.980892

The parameter α in a Morse function determines the shape of the curve: the smaller the quantity α, the broader the Morse curve and larger α results in a narrower curve.

If the quantity αR falls beyond the 'acceptable' range as mentioned above, then changing the value of α can improve the fit (keeping R and well depth the same for Lennard-Jones). For example, if αR is small, then increasing α will give a narrower Morse curve that gives a better fit to the LJ curve at the intermediate distance. At the same time, the quantity σ_{mor} also becomes larger and gives increasingly better fit to the equilibrium region until αR approaches 6.

Note that, changing R (as well as D_e) is thought to result in a more dramatic change to the overall structure. Hence, should DL_FIELD reports an unsatisfactory fit, users are recommended to change the Morse parameter α, perhaps in a *udff* file. Rerun DL_FIELD and see if the goodness of fit improves.

However, if the shape of the curve is more desirable, then changing R, rather than α, can also improve the fitting, so long $\alpha R \sim 6$. Due to the variation how the curve fittings can be improved, DL_FIELD left to the user's discretion how this can be achieved.

10.6 Using VDW_MIX

DL_FIELD permits the setting up of bio-inorganic models but make no attempt to mix the vdw parameters between organic and inorganic atoms. Such parameters can be inserted manually in the *dl_poly.FIELD* file, or by making use of the ***VDW_FIX*** directive.

Example below shows the latter part of the DL_POLY's FIELD file, produced from the *multiple_potential_2.pdb* file from the *Examples/* directory.

```
…
Molecule name not_define
nummols 1
atoms 24
Mg3_B       0.00000     2.00000  12   0
O10_B       0.00000    -2.00000  12   0
finish
vdw 28
CT3_A   CT3_A   lj       0.0800      3.6705
HA_A    CT3_A   lj       0.0420      3.0112
CT2_A   CT3_A   lj       0.0663      3.7730
OH1_A   CT3_A   lj       0.1103      3.4121
H_A     CT3_A   lj       0.0607      2.0353
HA_A    HA_A    lj       0.0220      2.3520
CT2_A   HA_A    lj       0.0348      3.1137
OH1_A   HA_A    lj       0.0578      2.7529
H_A     HA_A    lj       0.0318      1.3760
CT2_A   CT2_A   lj       0.0550      3.8754
OH1_A   CT2_A   lj       0.0915      3.5146
H_A     CT2_A   lj       0.0503      2.1377
OH1_A   OH1_A   lj       0.1521      3.1538
H_A     OH1_A   lj       0.0836      1.7769
H_A     H_A     lj       0.0460      0.4000
Mg3_B   Mg3_B   buck      0.0000      0.0000      0.0000
O10_B   Mg3_B   buck   26566.2720      0.3065      0.0000
O10_B   O10_B   buck   524868.3600      0.1490    667.8466
CT3_A   Mg3_B   ????  xxx  xxx
CT3_A   O10_B   ????  xxx  xxx
HA_A    Mg3_B   ????  xxx  xxx
HA_A    O10_B   ????  xxx  xxx
CT2_A   Mg3_B   ????  xxx  xxx
CT2_A   O10_B   ????  xxx  xxx
OH1_A   Mg3_B   ????  xxx  xxx
OH1_A   O10_B   ????  xxx  xxx
H_A     Mg3_B   ????  xxx  xxx
H_A     O10_B   ????  xxx  xxx
close
```

The example model consists of ethanol molecules and magnesium oxide. The organic molecule is associated with the Molecular Group 'ETOH'. For magnesium oxide, Molecular Group is not defined in the PDB file (see *multiple_potential_2.pdb*) and DL_FIELD assigns the Molecular Group to the inorganic materials as 'not_define'.

The ethanol molecule is assigned to the CHARMM22_prot FF, while a rigid-ion INORGANIC_binary_oxide FF model is assigned to magnesium oxide. The ATOM_KEYs were appended with the suffixes '_A' and '_B', respectively to distinguish the atoms to which FF schemes they belong.

The vdw LJ mixing rules are applied to organic atom pairs according to the CHARMM's mixing rules, while the vdw interactions for magnesium oxide are described by a set of Buckingham (buck) potentials. Since DL_FIELD cannot provide Buckingham to LJ12-6 transformation, all such cross combinations are listed as question marks and crosses. It is up to the user to manually insert the suitable parameters for these atom pairs.

As mentioned earlier, the ***VDW_FIX*** directive can be used in DL_FIELD to insert these mixed vdw parameter sets automatically, for instance, in an *udff* file as shown below:

```
POTENTIAL inorganic_binary_oxide

UNIT kcal/mol

VDW_FIX  charmm22_prot
CT3  Mg3    mors   2.40   3.50   0.52
Mg3  H      lj     0.06   2.88   0.0
END VDW_FIX

END POTENTIAL
```

By requesting DL_FIELD to read the *udff* file, the following FIELD file will be produced.

```
…
H_A     CT3_A   lj       0.0607      2.0353
HA_A    HA_A    lj       0.0220      2.3520
CT2_A   HA_A    lj       0.0348      3.1137
OH1_A   HA_A    lj       0.0578      2.7529
H_A     HA_A    lj       0.0318      1.3760
CT2_A   CT2_A   lj       0.0550      3.8754
OH1_A   CT2_A   lj       0.0915      3.5146
H_A     CT2_A   lj       0.0503      2.1377
OH1_A   OH1_A   lj       0.1521      3.1538
H_A     OH1_A   lj       0.0836      1.7769
H_A     H_A     lj       0.0460      0.4000
Mg3_B   Mg3_B   buck      0.0000      0.0000      0.0000
O10_B   Mg3_B   buck   26566.2720      0.3065      0.0000
O10_B   O10_B   buck   524868.3600      0.1490    667.8466
CT3_A   Mg3_B   mors      2.4000      3.5000      0.5200
CT3_A   O10_B   ????  xxx  xxx
HA_A    Mg3_B   ????  xxx  xxx
HA_A    O10_B   ????  xxx  xxx
CT2_A   Mg3_B   ????  xxx  xxx
CT2_A   O10_B   ????  xxx  xxx
OH1_A   Mg3_B   ????  xxx  xxx
OH1_A   O10_B   ????  xxx  xxx
H_A     Mg3_B   lj       0.0600      2.8800
H_A     O10_B   ????  xxx  xxx
close
```

New parameters are now inserted in the vdw section, highlighted in bold, in different functional forms such as the Lennard-Jones 12-6 (lj) function and the Morse (mors) potential function. Please note that these are arbitrary values and do not copy them for your own model setup and calculations!

Note to GROMACS users: there are some restrictions in GROMACS in handling mixed functional forms for vdw interactions. For example, at present, it does not permit mixing of LJ and Buckingham forms although DL_FIELD will still be able to generate a GROMACS file to that effect. Please see Section 14.3 for details about GROMACS restrictions.

11 Beyond the Standard (Potential) Models

This chapter describes some of the features available in DL_FIELD that can be used to modify the standard potential schemes according to user's needs. Note that features in this Chapter do not apply to *xyz* files since they are only used within a MOLECULE definition.

11.1 Pseudo Atoms

In DL_FIELD, a pseudo atom is an imaginary particle that can take any atomic mass and charge and can optionally include inter-molecular interactions (vdw and coulombic). Its position is fixed relative to other atom or group of atoms within a MOLECULE. In other words, [***RIGID***] directives that involve all pseudo atoms must be issued within the ***MOLECULE*** definition. Otherwise, DL_FIELD will flag up an error.

However, pseudo element symbols only take a certain subset of two-character format in the form of Q*X*, where *X* = 0 to 9 and A to V (case insensitive). This is how DL_FIELD identify an *atom_type* as a pseudo atom. For instance, Q3, QD are arbitrary pseudo atom symbols. QW, QX, QY and QZ are pre-defined for DL_FIELD internal use only.

Pseudo atom is a useful feature to modify the existing standard potential model for a particular user's model. For instance, inclusion of a pseudo atom into -N=N- to model the polarisability of the bond. There are some restrictions when using a pseudo atom within a MOLECULE:

(a) The pseudo element symbol and its corresponding ATOM_KEY and ATOM_LABEL must be exactly matched with identical characters of the form Q*X*.

(b) The pseudo ATOM_TYPE is unique within a MOLECULE. In other words, if there are more than one pseudo atoms in a MOLECULE, each individual pseudo ATOM must be assigned with different pseudo element symbols. However, similar pseudo ATOMs can be re-used for different MOLECULEs.

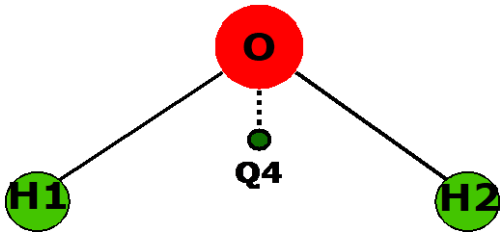
(c) A [***RIGID***] directive that contains a pseudo ATOM must always be issued within the MOLECULE. The directive must also contain at least one normal ATOM.

(d) Pseudo ATOMs must not be connected to any other atoms within the MOLECULE. In order words, pseudo ATOMs are not physically or covalently connected with any other atoms. Its positions relative to the others are held fixed by issuing a [***RIGID***] directive as mentioned above.

(e) For the reason given in (d), a self-CONNECT directive must be issued for a pseudo ATOM:

CONNECT Q*X* > Q*X*

An example usage of pseudo ATOM is shown in MOLECULE water_tip4p, which is the standard TIP4P water model available in the *DLPOLY_CHARMM22_prot.sf* file. A pseudo ATOM is defined with ATOM_KEY *Q4*. The pseudo ATOM is used in the TIP4P model as a pseudo point charge with no vdw interaction:



The definition of TIP4P MOLECULE in DL_FIELD are as follows:

```
...
ATOM_TYPE      key     element     mass     remark
...
pseudo_tip4p    Q4      Q4          0.000    TIP4P pseudo point
...
END ATOM_TYPE


MOLECULE_TYPE    key      mw          remark
...
water_tip4p              TIP4    18.015    Standard TIP4P
...
END MOLECULE_TYPE

MOLECULE water_tip4p 4 0.0
O      O_tip4p          0.00
H1     H_tip4p          0.52
H2     H_tip4p          0.52
Q4     pseudo_tip4p  -1.04
CONNECT O   > H1 H2
CONNECT H1  > O
CONNECT H2  > O
CONNECT Q4  > Q4
RIGID O H1 H2 Q4
END MOLECULE TIP4P
...
...
```

Note that Q4 does not have to confine for use in TIP4P water only. The same pseudo point can be reused for other purposes in the other MOLECULE definitions.

11.2 Core-shell Models

The core-shell models that are commonly encountered in inorganic systems can also be applied for ORGANIC force fields in DL_FIELD. An example is given in the */Examples* directory for the core-shell version of ethanol molecules for CHARMM force field (*ethanol_shell.pdb*). The corresponding *udff* file (*alcohol.udff*) contains the definition for the core-shell version of the ethanol MOLECULE, *ethanol_sh*:

```
MOLECULE ethanol_sh 10 0.0      Ethanol (ethyl alcohol) - shell model
C1   Cp_alkane       -0.27
H11  HC_alkane        0.09      H11  H21
H12  HC_alkane        0.09       |   |
H13  HC_alkane        0.09   H12--C1--C2--O--H
C2   Cs_alkane        0.05       |   |    \
H21  HC_alkane        0.09      H13  H22   OZ
H22  HC_alkane        0.09
O    O_hydroxyl_core  0.10
OZ   O_hydroxyl_shl  -0.76
H    HO_alcohol       0.43
CONNECT C1   > H11 H12 H13 C2
CONNECT H11  > C1
CONNECT H12  > C1
CONNECT H13  > C1
CONNECT C2   > C1 H21 H22 O
CONNECT H21  > C2
CONNECT H22  > C2
CONNECT O    > C2 H
CONNECT OZ   > OZ
CONNECT H    > O
SHELL O  OZ
END MOLECULE
```

Note that the ATOM_LABEL O is used for the core-component. Whereas, the OZ is in fact the ATOM_KEY for the shell component of the oxygen atom, which must be defined as a self-CONNECT ATOM.

For the ORGANIC force fields, instead of the core, the shell part of an atom will be defined in the FIELD file for the 1-4 vdw interactions. The 1-4 vdw interactions for the core part will always be zero. For this reason, DL_FIELD will look for the 1-4 vdw parameters for the shell-part of an atom. For the 1-4 Coulombic interactions, the DL_POLY directive ***exclude*** should be used in the CONTROL file to consider all possible combinations of 1-4 core-shell interactions. This directive is automatically included in DL_FIELD's *dl_poly.CONTROL* file.

Example below shows the vdw definition for the MOLECULE *ethanol_sh* (extracted from *alcohol.udff*).

```
VDW     Eps     Rmin/2   Eps_1-4  Rmin/2,1-4  Remark
O_cr    0.00    0.00       0.0      0.0        !
OZ     -0.1521  1.7700     0.0      0.0        !
END VDW
```

Note that the vdw parameters are defined for the shell since the 'chemistry' is occurred at the shell part of the atoms and the core must be set to zero. This will ensure that there is no vdw interaction between the core part of the atom with all other ATOMs.

For CHARMM and CHARMM19 potential schemes, if *Eps_1-4* and *Rmin/2,1-4* are set to some finite values, then these values will be used for the 1-4 vdw interactions instead.

12 The DL_F Notation

The DL_F Notation is a universal atom typing notation that is uniquely implemented in DL_FIELD. It is a consistent atom typing scheme that is contiguous across different force field schemes. Assignment of types of atoms, and therefore the appropriate potential parameters, in a molecular system is usually a non-trivial task. This is because different potential schemes usually have different sets of complicated rules how types of atoms are being assigned. The task is further complicated by the fact that the descriptions of these atom types are frequently cryptic in nature. Take an example, a methyl carbon atom is represented as 'CT3' and c3 in CHARMM and PCFF, respectively. Whereas OPLSAA just expresses it as 'CT'.

The introduction of the DL_F Notation allows DL_FIELD to assign ATOM_TYPEs based on a single conversion scheme that works across different FF schemes. This smoothens data transitions and minimises learning curves on user's side when converting molecular models among different FF schemes.

The DL_F notation applies to most FF schemes including OPLS2005, PCFF, CVFF, AMBER GAFF, OPLS CL&P and all CHARMM components. By using these FFs, DL_FIELD can convert structures in all formats (PDB, *xyz*, and *mol2* with some exceptions). Upon successful conversion, DL_FIELD will also generate an output file called *dlf_notation.output* in the *output/* directory. The file essentially contains atomic structures listed in DL_F Notation.

The Notation avoids the use of cryptic symbols and all ATOM_TYPEs are described in self-explanatory, easy to understand form, while at the same time can precisely indicate the chemical nature of each atom in a molecule.

Originally, DL_F Notation was only applicable to covalent molecules, such as general organic molecules and even specific classes of biomolecules such as sugars and proteins. This has now been extended to include inorganic molecules (See Section 12.8)

For more information about the DL_F Notation and how the conversion works please refer to the following references:

C. W. Yong, *J. Chem. Inf. Model.* **56**, 1405-1409 (2016)

C. W. Yong, SSRN (2022), http://dx.doi.org/10.2139/ssrn.4254942

The use of the DL_F Notation to annotate molecular interactions at atomistic scales was demonstrated in the following reference:

C. W. Yong and I. T. Todorov, *Molecules* **23**, 36 (2018)

12.1 The DL_F Atoms

The DL_F notation only recognises a number of atoms in the Periodic Table of the Elements. They include elements that are commonly encountered in organic molecules: hydrogen (H), carbon (C), oxygen (O), nitrogen (N), sulphur (S), phosphorus, boron (B), silicon (Si), alkaline metals, alkaline earth metals and all halides and noble gases.

12.2 The Chemical Group (CG) and Chemical Group Index (CGI)

The term Chemical Group (*CG*) refers broadly to the functional group in chemistry. The *CG*s are similar to functional groups in organic chemistry but within the context of chemical elements for molecular simulations. A *CG* consists of a specific group of *DL_F Atoms* and bonds within a molecular structure that exhibits a characteristic chemical behaviour of the molecule. Every *CG* is also represented by a unique *CG* index number, *CGI*, ranged between 1-9999.

Some examples of *CG* are *alkane*, *aldehyde*, *alcohol*, *acylhalide,* etc, with the corresponding unique *CGI* of 1, 17, 15, 18, respectively. For a complete list please refer to *DLF_notation* file in the *lib/* directory.

In some cases, a *CG* refers to a special group of molecules. For instance, *cyclopropyl* and *cyclobutyl* are *CG*s that contained saturated carbon atom in a 3-member and 4-member ring, respectively. These molecules usually have different structural behaviour and therefore assigned with different sets of potential parameters. On the other hand, the atoms from other cycloalkanes such as cyclopentane and cyclohexane are generally assigned to the *cycloalkane CG*. DL_FIELD also recognises certain complex *CG*s. For instance, *pyrene* (581), *hydantoin* (600) and *xanthine* (706).

12.3 Primary DL_F Tokens

These are supporting tokens that can be expressed optionally along with a *DL_F Atom*. They are used to locate or to distinguish one *DL_F Atoms* from another within a *CG*. The available primary tokens are shown below:

p, s, t, q – refer to the degrees of substitution on either carbon (*C*) and nitrogen (*N*) atoms, depending on the number of connected hydrogen (H) atoms. The C and N atoms can be classified into primary (*p*), secondary (*s*), tertiary (*t*) and quaternary (*q*) atoms.

For carbon atoms, this can be *Cp*, *Cs*, *Ct* and *Cq*, which refer to carbon atoms with three, two, one and zero hydrogen (H) atoms attach to a carbon atom.

For *N* atoms, *Np*, *Ns* and *Nt* refer to two, one and zero H atoms attach to an N atom. In some special cases such as the *ammonium CG*, which contain four bonds, *Np*, *Ns*, *Nt* and *Nq* refer to nitrogen atoms with three, two, one and zero hydrogen (H) atoms attach to a nitrogen atom.

'n – refer to the degree of general substitution, where n refers to the degree of substitution with a range of 1 to 3. It is similar to the above with the exception that the degree of substitutions depends on the number of substituted atoms connected to the atom. These substituted atoms are often belonged to other *CG*s. For instance, *C'2* in the *alkane CG* means secondary (second degree) substitution where two of the hydrogen atoms were substituted with some other atoms such as nitrogen, oxygen, etc, **but not** with another *C* atom belong to the *alkane*.

E – refers to a terminal position within a *CG* (the 'end' atom).

L – linked position within a *CG* (the 'link' atom).

The positive (+) and negative (-) signs refer to the charge state of a *DL_F Atom.*

1, *2*, *3*, etc – numerical values. These tokens usually apply to aromatic or pyranose cyclic compounds.

12.4 Secondary DL_F Tokens

This is a group of optional supporting tokens that can be expressed along with a *DL_F Atom*, to indicate the way or the type of atom to which it is connected to. The available tokens are shown below:

R - next to an aryl carbon atom.

U - next to an unsaturated carbon atom (such as alkene, alkyne type).

A - alpha, refers to the first atom that connect directly to an atom within a *CG*.

B - beta, refers to the second atom that connect directly to an atom within a *CG*.

G – gamma, refers to the third atom that connect directly to an atom within a *CG*.

Element symbols of the neighbouring *DL_F Atom* itself.

12.5 The Notation Rules

The following lists the rules in using the DL_F Notation.

(a) Every *DL_F Atom* in a system model will have an *ATOM_TYPE* that associates with a *CG* tag to which it belongs. This means that the corresponding *DL_F ATOM_KEY* will associate with the corresponding *CGI*.

(b) Every *DL_F Atom* can only associate with not more than one token from ***each group*** of the *DL_F token*. This means that there can be only one token, either from the *Primary DL_F Token* or the *Secondary DL_F Token* that associates with a *DL_F Atom*. However, two tokens are permissible only if each is originated from both *Primary* and *Secondary Token* groups.

(c) Single-valence *DL_F Atoms* such as hydrogen and halogens must always express along with a *Secondary DL_F Token*. This is usually the element symbol of a neighbouring *DL_F Atom*.

12.6 The Notation Expressions

In a DL_F expression, each ATOM_TYPE is referenced to an ATOM_KEY. The DL_F ATOM_TYPEs are expressed as follows:

A[t]_CGname[#n]

Where *A* is the element symbol of a *DL_F Atom*, *[t]* is the optional *DL_F token* and *CGname* is the name of the Chemical Group to which the atom *A* belongs.

The corresponding DL_F ATOM_KEYs are expressed as follows:

ACGI[t][#n]

Where *A* is the element symbol of a DL_F Atom, *CGI* is the Chemical Group index number and [t] is the optional *DL_F token*.

The optional expression, *#n*, is called the version number. If there are more than one version of identical ATOM_TYPEs, but may share different parameters, then *#n* would be

needed, where *n* is an arbitrary integer. Different values of *n* differentiate one version of ATOM_TYPE from the others. The version labelling would be needed, especially for MISC_FF scheme, which include a range of FF models from different sources.

For example, a primary alkyl carbon atom has the DL_F ATOM_TYPE of *Cp_alkane* with the corresponding ATOM_KEY of *C1p*. In this example, C is the carbon atom, *p* is a primary token refers to *primary* and *alkane* is the *CG,* which has the unique *CGI* of 1.

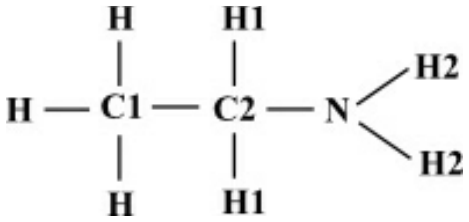
The alkyl hydrogen atoms are expressed as *HC_alkane* with the ATOM_KEY *H1C*. In this case, *C* is a *Secondary DL_F Token* which is the neighbouring *DL_F Atom* to which the hydrogen atom is connected to.

If there are, say, two versions of identical ATOM_TYPEs, then this can be distinguished from each other's as *Cp_alkane#1*, *Cp_alkane#2* with the corresponding ATOM_KEYs of *C1p#1* and *C1p#2*, respectively.

12.7 Organic Molecules Examples

The following lists several example structures together with the assignment of ATOM_TYPEs in the DL_F Notation.

(a) Ethylamine



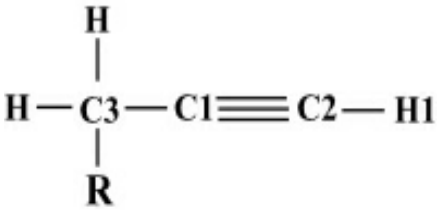
This is a primary amine and DL_FIELD will assign the ATOM_TYPEs (ATOM_KEYs in bracket) for each atom as follows:

C1 = *Cp_alkane* (*C1p*)
C2 = *Cs_alkane* (*C1s*)
H = *HC_alkane* (*H1C*)
H1 = *HC_alkane* (*H1C*)
N = *Np_amine* (*N45p*)
H2 = *HN_amine* (*H45N*)

Note that DL_FIELD will automatically search for any conditional modification based on the chosen FF scheme and reassign an ATOM_TYPE accordingly. For example, the C2 atom can be reassigned as *CA_amine* with the ATOM_KEY *C45A,* implicating the fact that the C2 will be assigned to a different set of potential parameters due to the presence of the neighbouring nitrogen atom. Since H1 atoms are connected to C2, which is now an alpha atom amine, they will also reassign to become *HA_amine* (*H45A*).

Note that such conditional modifications will only apply if needed depending on the nature and the type of FF chosen. Even then, users can still make out cognitively the identity of the atom despite the change of the ATOM_TYPE expression.

(b) Propyne



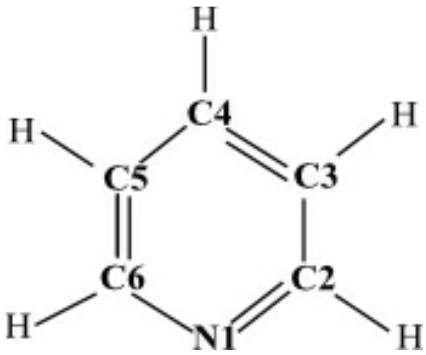
This is an alkyne and DL_FIELD will assign the ATOM_TYPEs (ATOM_KEYs in bracket) for each atom as follows:

C1 = CL_alkyne (C3L)
C2 = *CE_alkyne* (*C3E*) - terminal alkyne carbon
H1 = *HC_alkyne* (*H3C*)
C3 = *Cs_alkane* (*C1s*)
H = *HC_alkane* (*H1C*)

Once again, DL_FIELD will automatically search for any necessary reassignment. For instance, in OPLS2005, the C1 atom can be assigned with different parameter sets depending on the neighbouring atom, which is the C3, a secondary alkane carbon atom. DL_FIELD can reassign C1 as *CCs_alkyne* (*C3Cs*). If R = H, then the C3 would be *Cp_alkane* (*C1s*) and the C1 will be reassigned as *CCp_alkyne* (*C3Cp*).

Note that Rule (2) as described in Section 12.5 still holds where the '***C***' in *C3**C**s* is the Secondary DL_F Token and the '***s***' in *C3C**s*** is the Primary DL_F Token.

(c) Pyridine



The DL_F notation apply specific rules for most aromatic cyclic compounds. The aromatic carbon atoms are distinguished from one another by a numerical token assigned serially around the ring. In the case of hydrogen atoms, they are assigned with the generic ATOM_TYPE *HC_aromatic*. Take example for a pyridine molecule, DL_FIELD will assign the atoms as follows:

N1 = *N1_pyridine* (*N501-1*)
C2 = *C2_pyridine* (*C501-2*)
C2 = *C3_pyridine* (*C501-3*)
C2 = *C4_pyridine* (*C501-4*)
C2 = *C5_pyridine* (*C501-5*)
C2 = *C6_pyridine* (*C501-6*)
H = *HC_aromatic* (*H11C*)

Where *CGI* of 501 and 11 refer to pyridine and the generic aromatic *CG*s respectively. Note that the numerical tokens are expressed as *-1*, *-2*, etc, to distinguish them from the *CGI*.

The *Examples/* directory contains a few *xyz* structures to illustrate the use of the DL_F Notation. To do this, the value 1 must be selected for Option **12** in the *control* file, to instruct DL_FIELD to write out atoms in DL_F Notation.

If the user configuration file is in the *xyz* or *mol2* formats, DL_FIELD will produce a user-structure file, called the *dlf_notation.output* in the *output/* directory. The file contains a list of ATOM_TYPEs expressed in DL_F Notation that match with the atoms in the user configuration.

12.8 DL_F Notation for Inorganic Systems

Atoms in organic molecules are usually identifiable based on their chemical characteristics that is almost always depending on the connectivity to its surroundings. For inorganic system, similar chemical characteristics can have different parameters depending on the types of inorganic materials, atomic environments, and their coordination numbers. For this reason, while topologically speaking, the chemical structures of inorganic materials are generally less complex than those of organic molecules, annotating inorganic species are much less straight forward.

12.8.1 ATOM_TYPE format

Similar to the DL_F expression for organic atoms, each ATOM_TYPE is referenced to an ATOM_KEY. The DL_F ATOM_TYPEs for inorganic species are expressed as follows:

E[X][Y]#*y*_CG*Q*

E is the element symbol of the atom type.

[X] is the formal charge value, including the charge sign. For instance, 2+, 2- or simply + or – for unit charge. It can be omitted if it is a neutral species (leave blank).

[Y] contain additional, optional neighbours and connectivity information relate to E. For instance, neighbouring atoms and next nearest neighbour atoms to E. Up to two neighbour atoms and two next neighbour atoms can be specified. The general notation is as follows:

N1N2-N3N4

It means N1 and N2 atoms are two of the nearest neighbour atoms to E, and N3 and N4 are the next nearest neighbour to E, which are connected to N2.

is the (usual) coordination geometry or state of an atom and takes the following values for *y*:

2 = (coordinated to two atoms) linear

3 = (coordinated to three atoms), trigonal

4 = (coordinated to four atoms), tetrahedral, square planar

5 = (coordinated to five atoms), trigonal bipyramid, square pyramid.

6 = (coordinated to six atoms), octahedral, trigonal prismatic.

7 = (coordinated to seven atoms), octahedral, trigonal prismatic

a = amorphous

l = liquid state, for example, ions in solution

g = gas phase

c = core part of an atom.

s = shell part of an atom.

Note that the value of *y* is only an indication of how many neighbouring atoms are to be expected. The program does not necessarily flag up an error should the neighbour criteria is not met. For instance, surface atoms have less coordination than those in the bulk. Even then, the Notation allows one to create ATOM_TYPEs, and hence, distinguish these atoms from one another.

CG is the same as the organic counterparts, which is the Chemical Group for the inorganic systems. Examples are *binary_oxide*, *binary_halide*, *glass*, etc.

Q is the numerical suffix to distinguish different versions of force field models to which E belongs. For instance, the value *Q* is used to indicate a particular type of Fe^{3+} from many different versions of potential models for iron(III) oxide, developed by different workers. In the standard library files, *Q* is usually referred to the reference source to which the parameters were derived from.

For example, the ATOM_TYPE for Fe^{3+} ion core in iron(III) oxide is described as

Fe3+#6c_binary_oxide1

In this case, the suffix '*1*' refers to the particular set of potential models to which the Fe^{3+} cation belongs. Note that since it is the core part of the cation, it means there must also be a shell part of the cation:

Fe3+#6s_binary_oxide1

If the cation is a rigid ion model, then it can be expressed as

Fe3+#6_binary_oxide2

Here, the suffix '2' distinguishes the cation from all other Fe^{3+} ions that belong to different force field models.

12.8.2 ATOM_KEY Format

The corresponding ATOM_KEYs can be generated *in-situ* from the DL_F ATOM_TYPEs. This means that there is no need to keep a separate reference list in the library files. The corresponding DL_F keys for inorganic systems are expressed in almost the same sequence as follows:

E[X][Y]#[y]_*CGI*

The *CGI* is called the Chemical Group Index, which is the unique value corresponds to a Chemical Group (see the file *DLF_notation* in the *lib/* folder). A *CGI* value of > 5000 refers to an inorganic *CG*.

However, the ATOM_KEY must be shortened to accommodate the restriction of up to the maximum permissible of 8 characters for atom labels in DL_POLY files. For this reason,

the abbreviated version of ATOM_KEYs will be adopted with the following information being omitted (or abbreviated):

(1) Model version, *Q*, are omitted.

(2) Abbreviation symbols – A, D, E, G, etc, to represent E[X][Y]#[*y*], one symbol for a different atom type. Note, only a selection of alphabet symbols are used to distinguish them from the standard symbols in the Periodic Table of the Elements.

The ATOM_KEY is now reduced to the following format:

A_*CGI* where A = E[X][Y]#[y]

Example 1: sodium oxide crystal adopts an antifluorite structure. The oxygen core anion is described as

O2-#8c_binary_oxide1

The corresponding ATOM_KEY would be

A_5001

Where *A* = *O2-#8c*.

Example 2: the oxygen atom that links with neighbouring silicon atoms in a clay framework is

O2-Si#2_clay1

The corresponding ATOM_KEY would be

A_5010

Where A = *O2-Si#2* and *5010* is the CGI for *clay* CG.

Example 3: the shell part of the oxygen atom in a silicalite framework can be shown as:

O2-#2s_zeolite_silicalite4

The corresponding ATOM_KEY would be

A_5101

Here *A* = *O2-#2s*.

Example 4: Bridging oxygen at the aluminosilicate zeolite can be described as

O2-AlSi#2_zeolite_aluminosilicate

Here, [Y] = *AlSi*, which means Al and Si are the neighbours to the two-coordinated oxygen. In other words, this is a bridging O in the zeolite framework.

The corresponding ATOM_KEY would be

A_5103

Where A = *O2-AlSi#2*.

Example 5: To distinguish hydroxyl hydrogen atom (H1) from terminal and the hydrogen atom at the Bronsted oxygen site (H2), as in alumina-silicic acid. The ATOM_TYPE can be expressed as follows.

H+O-Si#1_aluminasilicic for H1,

H+O-AlSi#1_aluminasilicic for H2

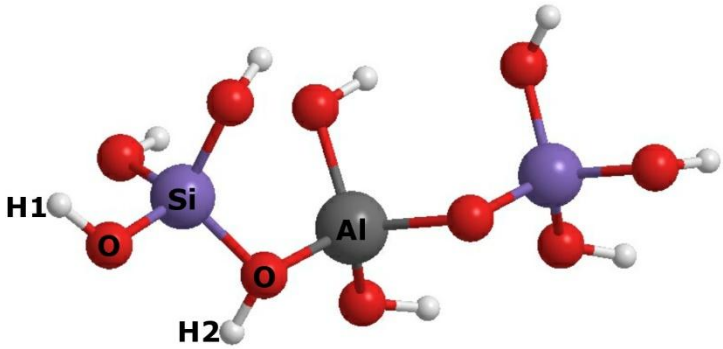


Diagram above shows the positions of the two hydrogen atoms. For H1, [Y] = O-Si, where the O is the neighbouring atom to H1 and Si is the second-nearest atom that is linked to O. For H2, [Y] = O-AlSi, which means the O is the neighbouring atom to H2 and Si and Al atoms are the second nearest atoms that are linked to the O.

The corresponding keys are as follows:

A_5122 for H1 with A = *H+O-Si#1*.

B_5122 for H2 with B = *H+O-AlSi#1*.

13 Third-Party FF Schemes

DL_FIELD can setup FF models derived from third-party sources. This is achieved either by preparing a file format from third-party sources that is recognisable by DL_FIELD, or read in other FF file formats and DL_FIELD will convert these into *udff* files.

The following sections describe how this can be done in DL_FIELD.

13.1 Amber GAFF from Antechamber

The General Amber FF (GAFF) is a useful FF scheme specially designed for organic molecules such as small drug molecules. The force field is already implemented in DL_FIELD (amber16_gaff). However, what is missing is the partial charge values which can be obtained from Amber's *Antechamber* program.

One way to obtain these charges is to run *antechamber* program and produce an output file in the *mol2* format. This section explains how one can use the *Antechamber* software to produce a *mol2* file, which can be read by DL_FIELD to produce Amber GAFF model for DL_POLY simulations.

Let's suppose you want to setup a GAFF model for ascorbic acid (Vitamin C) and the initial input file for the structure is in the PDB format. The following command can be issued in Amber:

antechamber -if pdb -i ascorbic.pdb -fo mol2 -o ascorbic_acid.mol2 -c bcc

This instructs *Antechamber* to produce a *mol2* output file. The *bcc* is the charge model being used, which is the abbreviation for AM1-BCC. The charge model was parameterised to reproduce HF/6-31G* RESP charges. These partial charge values are shown at the last column for each atom in the *mol2* file (see below).

Then, use this *ascorbic_acid.mol2* file as the input configuration in DL_FIELD and specify amber16_gaff scheme (Option **1** in the *control* file). DL_FIELD will automatically recognise the file is compatible with the GAFF and produces the corresponding input files for DL_POLY with the set of charges as shown in the *mol2* file.

Note that the same input file can also be used for other FF schemes, such as OPL2005. In this case, the element symbols will be extracted from 'atom labels' and DL_FIELD will treat the configuration as if it is a *xyz* file and the usual conversion procedure will be used to setup the force field model. ***However, DL_FIELD will ignore the charge values in the mol2 file and determine the partial charges for the molecule according to the OPLS2005 formalism***.

Similar to the *xyz* format, a *dlf_notation.output* file will be produced after each successful conversion.

The example below shows a *mol2* file of an ascorbic acid (vitamin C) molecule, produced by using the Antechamber software, with AM1-BCC charge method for AMBER16_GAFF.

Note that, if setting up using newer AMBER25_GAFF, then use the new bond charge correction for GAFF2:

antechamber -if pdb -i ascorbic.pdb -fo mol2 -o ascorbic_acid.mol2 -c abcg2

```
@<TRIPOS>MOLECULE
****
  20   20    1    0    0
SMALL
bcc


@<TRIPOS>ATOM
     1 C         0.5320    0.6960    0.5130 c3       1 ***      0.140400
     2 C1        1.1130   -0.7070    0.4630 c3       1 ***      0.102100
     3 H         1.1810    1.3910    1.0910 h1       1 ***      0.065200
     4 O        -0.7190    0.6670    1.1730 oh       1 ***     -0.591800
     5 H1        0.3760    1.1160   -0.5050 h1       1 ***      0.065200
     6 H2        0.9650   -1.2210    1.4420 h1       1 ***      0.064700
     7 O1        0.4260   -1.4590   -0.5120 oh       1 ***     -0.599800
     8 C2        2.5880   -0.7270    0.0980 c3       1 ***      0.148300
     9 O2        3.3640   -0.0400    1.0670 os       1 ***     -0.398900
    10 H3        2.7690   -0.2680   -0.9010 h1       1 ***      0.112700
    11 C3        3.1190   -2.1380    0.1410 c2       1 ***      0.041900
    12 C4        4.2130   -0.9780    1.5850 c        1 ***      0.713800
    13 O3        5.0310   -0.7710    2.4570 o        1 ***     -0.549500
    14 H4       -1.1250    1.5360    1.1300 ho       1 ***      0.414000
    15 H5        0.8040   -2.3420   -0.5630 ho       1 ***      0.423000
    16 O4        4.8120   -3.2890    1.4290 oh       1 ***     -0.472100
    17 C5        4.0860   -2.2150    1.0540 ce       1 ***     -0.111100
    18 O5        2.6680   -3.1720   -0.6010 oh       1 ***     -0.467100
    19 H6        3.1950   -3.9540   -0.3620 ho       1 ***      0.452000
    20 H7        5.3850   -3.0020    2.1610 ho       1 ***      0.447000
@<TRIPOS>BOND
    1    1    2 1
    2    1    3 1
    3    1    4 1
    4    1    5 1
    5    2    6 1
    6    2    7 1
    7    2    8 1
    8    4   14 1
    9    7   15 1
   10    8    9 1
   11    8   10 1
   12    8   11 1
   13    9   12 1
   14   11   17 2
   15   11   18 1
   16   12   13 2
   17   12   17 1
   18   16   17 1
   19   16   20 1
   20   18   19 1
@<TRIPOS>SUBSTRUCTURE
    1 ***       1 TEMP          0 ****  ****    0 ROOT
```

13.2 OPLSAA force field from LigParGen to udff

The LigParGen is an FF web server interface (http://zarbi.chem.yale.edu/ligpargen/) that can automatically generate OPLSAA models for organic molecules, such as drug molecules or ligands with proteins and DNA. Users would need to specify the molecules or supply with a configuration file such as PDB, and the server will produce a number of FF files for

different simulation software. More details about LigParGen can be found in the following reference:

LigParGen web server: An automatic OPLS-AA parameter generator for organic ligands. Dodda, L. S.;Cabeza de Vaca, I.; Tirado-Rives, J.; Jorgensen, W. L. *Nucleic Acids Research*, **45**, Issue W1, W331-W336 (2017)

Later DL_FIELD versions, such as 4.10 onwards, can transcribe these FF models into DL_POLY and GROMACS files, via the Charmm's rtf and prm files produced by LigParGen. This is done as follows, by using ascorbic acid as an example:

Step 1: Use ascorbic_acid.pdb (available in the *Examples/* folder) as the input structure for LigParGen server and select the 1.14*CM1A-LBCC charge model.

Step 2: Upon successful run, users are given several Download options. Select and download PRM (parameter) and RTF (residue topology file) files for CHARMM/NAMD.

Step 3: Rename these files as *ascorbic_acid.rtf* and *ascorbic_acid.prm*, respectively. Note: renaming process is not essential, provided DL_FIELD can distinguish rtf and prm files as shown in Step 4.

Step 4: Specify the rtf and prm files in the *dl_f_path* (see below). This will instruct DL_FIELD to read in the FF files and convert them into an *udff* file.

```
# paths
library = lib/
solvent = solvent/
output  = output/

# paths and filenames
control = dl_field.control5

charmm_rtf = ascorbic_acid.rtf    opls2005
charmm_prm = ascorbic_acid.prm  opls2005
```

The relevant input files are highlighted in yellow. The *charmm_rtf* and *charmm_prm* are types of files DL_FIELD is expected to read. Each of the type takes two parameters: the associated filename, and the FF scheme to which it belongs. The latter information will be used by DL_FIELD to setup the *udff* file. For example, for *charmm_rtf*, the filename is *ascorbic_acid.rtf* and the FF scheme is OPLS2005.

Note that although rtf and prm are the original file formats for CHARMM software, in this case, the LigParGen produces FF data for OPLSAA in the rtf (structure and the model charge) and prm (potential parameters) formats. For this reason, the FF scheme must be specified as OPLS2005 in DL_FIELD.

Step 5: Setup the DL_FIELD control file as follows, with the relevant options highlighted in yellow.

```
Control file title.
1       * Construct DL_POLY output files
none        * chemsh interface (for QM/MM), or gromacs
opls2005  * Type of force field require (see list below for choices).
kcal/mol  * Energy unit: kcal/mol, kJ/mol, eV, or K.
normal  * Conversion criteria (strict, normal, loose)
1       * Bond type (0=default, 1=harmonic , 2=Morse)
1       * Angle type (0=default, 1=harmonic, 2=harmonic cos)
none  * Include user-defined information. Put 'none' or a .udff filename
1      * Verbosity mode: 1 = on, 0 = off
ascorbic_acid.pdb   * Configuration file.
none   * Output file in PDB. Put 'none' if not needed.
0 40 molecules  9.0 * Solution Maker: on/off, density, unit, cutoff)
```

When DL_FIELD is run, it will process the rtf and prm files and produce an *udff* file called the *dlf_charmm.udff* file. It contains the information from both the rtf and prm for OPLS2005 FF. ***Since no udff file is specified in the DL_FIELD control file, DL_FIELD will stop execution and will not carry out the FF model conversion, even if the input configuration file is specified***. The following message will be shown on the screen:

```
…
…

Processing Charmm rtf file: ascorbic_acid.rtf...
Completed.

Processing Charmm prm file: ascorbic_acid.prm...
Completed.

The udff file dlf_charmm.udff has been created.
Only file format conversions had been carried out.

Program executed successfully. Thank you for using DL_FIELD.
```

However, file format conversions and FF model set up processes can be combined by carrying out the additional steps as follows:

Step 6: From the rtf file, the ascorbic acid molecule is given the RESIdue name UNK. This would be the same label for the MOLECULE_KEY in *dlf_charmm.udff* file. This label must

```
REMARK PDB file
HETATM    1  CT  UNK     1      0.532   0.696   0.513  1.00  0.00         C
HETATM    2  CT  UNK     1      1.113  -0.707   0.463  1.00  0.00         C
HETATM    3  HC  UNK     1      1.181   1.391   1.091  1.00  0.00         H
HETATM    4  OH  UNK     1     -0.719   0.667   1.173  1.00  0.00         O
HETATM    5  HC  UNK     1      0.376   1.116  -0.505  1.00  0.00         H
HETATM    6  HC  UNK     1      0.965  -1.221   1.442  1.00  0.00         H
HETATM    7  OH  UNK     1      0.426  -1.459  -0.512  1.00  0.00         O
…
…
```

be included in the PDB file (at column 18-21 as shown below), so that DL_FIELD would know which MOLECULE to use when setting up the FF.

Step 7: Insert the udff filename *dlf_charmm.udff* in DL_FIELD control file in Option **6**.

When running DL_FIELD, it will read and transcribe the rtf and prm files into the *udff* files **and** follow by FF model setup.

```
Control file title.
1       * Construct DL_POLY output files
none       * chemsh interface (for QM/MM) or gromacs
opls2005  * Type of force field require (see list below for choices).
kcal/mol  * Energy unit: kcal/mol, kJ/mol, eV, or K.
normal  * Conversion criteria (strict, normal, loose)
1       * Bond type (0=default, 1=harmonic , 2=Morse)
1       * Angle type (0=default, 1=harmonic, 2=harmonic cos)
dlf_charmm.udff  * Include user-defined information. Put 'none' or a .udff filename
1      * Verbosity mode: 1 = on, 0 = off
ascorbic_acid.pdb   * Configuration file.
none   * Output file in PDB. Put 'none' if not needed.
0 40 molecules  9.0 * Solution Maker: on/off, density, unit, cutoff)
```

13.3 CHARMM FF from MATCH server to udff

The MATCH (Multipurpose Atom Typer for CHARMM) is a molecule-matching program developed for CHARMM force fields. The program source is available for download, but the MATCH program can also be accessed via a web server, although it comes with a limited functionality.

MATCH (https://brooks.chem.lsa.umich.edu/index.php?matchserver=submit).

J.D. Yesselman, D.J. Price, J.L. Knight, C.L. Brooks III, 'MATCH: An atom-typing toolset for molecular mechanics force fields', *J. Comp. Chem.* **33**, 189-202 (2012)

Like LigParGen (See Section 13.2), user would need to supply a molecular configuration (completed with all hydrogen atoms) in PDB or mol2 format and the Server will produce the corresponding rtf and prm file for the molecule, archived in a tar file.

User can use the *ascorbic_acid.pdb* file included in the *Examples/* folder as a test.

Once downloaded and untarred, these files can be read by DL_FIELD to produce the corresponding udff file by using the similar procedures as mentioned in Section 13.2. Below shows an example of reading in the necessary files.

```
# paths
library = lib/
solvent = solvent/
output  = output/

# paths and filenames
control = dl_field.control5

charmm_rtf = 1010532/1010332.rtf    charmm36_cgenff
charmm_prm = 1010532/1010332.prm   charmm36_cgenff
```

Note the use of CHARMM36 CGenFF, since the MATCH Server uses the CHARMM's general force field to setup the FF models.

13.4 Conversion from CHARMM PSF file to udff format

DL_FIELD can also read CHARMM's protein structure files (psf) produced from the CHARMM's *psfgen* command. When scanning through psf, DL_FIELD can identify new residues and setup as new MOLECULE templates in a *dlf_charmm.udff* file.

A psf contains complete topological information of a molecular system, including complex systems such as biomolecules that contain ligands. When a psf is generated, the CHARMM program will also produce a matching pdb file.

Both psf and the matching pdb file can be supplied to DL_FIELD, which can setup the corresponding FF files for DL_POLY run.

For example, consider a psf of a biomolecular system consists of a heam-cytochrome (a hemoprotein that contains a heme *c* complex) that connects with several ligating amino acid residues. This means the chemical structures of these ligating residues can be different from normal residues residing in other parts of proteins. Example below shows the content of the *dl_f_path* that instructs DL_FIELD to carry out the psf to udff conversion:

```
# paths
library = lib/
solvent = solvent/
output  = output/

# paths and filenames
control = dl_field.control5

charmm_psf = ../charmm/heme_oxidised.psf charmm36_prot
charmm_pdb = ../charmm/heme_oxidised.pdb charmm36_prot
charmm_prm = ../charmm/heme_ligand.prm charmm36_prot
```

DL_FIELD will scan the psf file and identify any new, modified, or patched residues that are not available in the library. These residues will be setup as new MOLECULE templates in the *dlf_charmm.udff* file.

Below shows part of the output. In this example, LYS60 and CYS122 are written out as new MOLECULEs in the *udff* file. If a matching pdb file is also supplied, then DL_FIELD will write out a modified *dlf_charmm.pdb* file that is compatible with the *udff* file.

If a CHARMM prm file is also supplied, then new parameters will be transferred to the *udff* file.

```
…
Total bond pairs in psf file: 14599
Total improper sets in psf file: 848

*** Analysing psf file...
Total atoms in psf file: 14511
In PSF file: Residue 1 is in fact -NPRO-

Residue LYS60 is a new amino acid patch.
New MOLECULE_TPYE = lysine_residue1  New MOLECULE_KEY = LYS1

In PSF file: Residue 122 is in fact -CYX-

Residue CYS122 is an amino acid with different charges.
New MOLECULE_TPYE = cysteine_residue2  New MOLECULE_KEY = CYS2
…
…
```

If *dlf_charmm.udff* file is pre-defined in a DL_FIELD *control* file, then FF conversion will also take place after the format conversion and reads the input configuration file (see example below).

```
Control file title.
1       * Construct DL_POLY output files
0       * chemsh interface (for QM/MM)
charmm36_prot  * Type of force field require (see list below for choices).
kcal/mol  * Energy unit: kcal/mol, kJ/mol, eV, or K.
normal  * Conversion criteria (strict, normal, loose)
1       * Bond type (0=default, 1=harmonic , 2=Morse)
1       * Angle type (0=default, 1=harmonic, 2=harmonic cos)
dlf_charmm.udff  * Include user-defined information. Put 'none' or a .udff filename
1       * Verbosity mode: 1 = on, 0 = off
dlf_charmm.pdb   * Configuration file.
none   * Output file in PDB. Put 'none' if not needed.
0 40 molecules  9.0 * Solution Maker: on/off, density, unit, cutoff)
```

If *dlf_charmm.udff* is not specified, then DL_FIELD will only carry out format conversions.

Please note the following restrictions:

(a) The rtf and psf files cannot be specified at the same time.

(b) All relevant molecules (for example, the cytochrome and the heme complex) must be grouped in the **same** Molecular Group or segment name and contained within a same block in the psf and matching pdb files. This is because DL_FIELD **assumes no bond** among the molecules from different Molecular Groups (or chain segments).

If this is not the case, then the following error may appear:

```
Error in assign_linked_atom(), for XX910 (910). Current Molecular Group = HA
This error may arise because DL_FIELD detects an improper statement
of which one of the atom belongs to the other (linked) residue that does not belong to the
current Molecular Group.
To solve this error, please ensure both residues are assigned to a same Molecular Group.
```

In the example model of cytochrome-heme *c*, DL_FIELD identifies an improper interaction statement in the psf file that straddles across the heme molecule and a ligating protein residue that does not belong to the same group: The heme belongs to HA, whereas, the protein belongs to PA (not shown in the error statement above).

If this is the case, manually change one of the molecules to the same identifier (normally the complex molecule), in **both** the psf and pdb files.

(c) If there are missing parameters for dihedral sets, they are assumed to be zero. This normally occurs around the ligands and the complex molecule and are not defined in the psf file.

14. Force field models for GROMACS

The following information assumes you understand GROMACS file structures and commands. For more information about GROMACS, please refer to the GROMACS manual.

In DL_FIELD, there is an option to produce FF files for GROMACS software for simulation runs. In most cases, DL_FIELD can setup a runnable set of GROMACS FF files without further modification.

To use this feature, the keyword *gromacs* must be used at Option **B** in the DL_FIELD control file.

Upon successful FF conversions, DL_FIELD will produce the following files:

(1) The system topology file, *gromacs.top*
(2) One or more include topology files, *gromacsX.itp*, where X = 1, 2, 3…
(3) A coordinate file, *gromacs.gro*
(4) A generic MD parameter file, *gromacs.mdp*

DL_FIELD can produce several *itp* files, which correspond to the number of Molecular Groups defined in the configuration file. Each *itp* file contains force field information for each Molecular Group. These *itp* files are included in the *top* file, which defines the ATOM_TYPEs and provides a summary of system composition.

If GROMACS program (gmx) is also installed in the computer, DL_FIELD can also run the program by creating a *tpr* binary file from the GROMACS files mentioned above. For more details, please see Section 4.2.

14.1 Force field settings

GROMACS contains numerous directives and different ways to achieve same FF definitions. To avoid ambiguity, DL_FIELD adopts the following default rules to setup force field models for all FF schemes:

(1) The *nrexcl* is always set to 3. This means non-bonded interactions (vdw and Coulombic) on the 1-2, 1-3 and 1-4 atom pairs will be excluded by default.

(2) The *gen-pair* parameter is always set to *no*. This instructs GROMACS **not to** automatically generate 1-4 atom pairs for non-bonded interactions.

(3) The fudge QQ (1-4 Coulombic scaling factor) and fudge LJ (1-4 vdw scaling factor) defined in the *[defaults]* directive will be set to the chosen FF scheme. These scaling factors are shown as a reference. For a multiple-potential setting, they are set to a default value of 1.0. In both cases, they are most probably not going to be used because the correct scaling factors are also defined in the *[pair]* directive (see below).

The 1-4 interactions, including the associated scaling factors or specific potential parameters, will be explicitly defined under the *[pair]* directive for every pair of 1-4 interacting atoms. If needed, this is where any additional non-bonded interactions that has been excluded due to the *nrexcl* value will be defined.

The vdw parameters specify under the *[pair]* directive can be different and overrides the default values defined under the *[atomtypes]* directive in the *top* file. Here, fudge LJ is introduced via rescaling of the vdw parameters and fudge QQ is explicitly defined.

All other non-bonded interactions, like those of 1-5 and beyond, will be mixed according to the combination rule set in the *[default]* directive. The vdw parameters and charges for these calculations will be extracted from *[atomtypes]* and *[atoms]* respectively.

14.2 Multiple potential systems

If a multiple potential is used, then **only one** Molecular Group can be defined for each different FF scheme. This means the number of *itp* files produce corresponds to the number of different FF schemes employed in the molecular system.

In a normal, single-potential setting, the vdw non-bonded interactions are automatically setup by GROMACS according to the combination rules defined in the *[defaults]* directive. In the case of multiple potential settings, the default combination rules will be ignored. Every possible atom pairs of vdw interactions will be explicitly defined under the *[nonbond_params]* directive, including atom pairs from two different FF schemes. Atom pairs that belong to a same FF scheme will be mixed with the rule set according to that FF scheme, while those from two different schemes will be mixed as per the options selected in the DL_FIELD *control* file.

Please see Section 14.3 below for the restrictions in using multiple potential settings.

14.3 Notes and restrictions

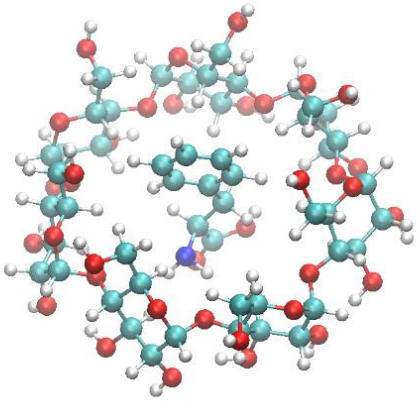
(1) Irrespective of what energy unit is specified in DL_FIELD *control* file, all energy unit will be converted to kJ/mol and nm for the unit distance in the GROMACS files.
(2) FF conversions only work for FF schemes that use LJ 12-6 and Buckingham potentials. For example, PCFF that uses LJ 9-6 form would not work. In addition, Buckingham potentials only work for older version GROMACS (before 2019 versions).
(3) GROMACS imposes a fix-format on the *gromacs.gro* coordinate files. For instance, the atom labels can only display up to five-character in length and the coordinates are specified to three significant figures. An error can occur when running *gmx* if the data format is violated, and DL_FIELD will not check for this.
(4) For multiple-potential models, only systems based on Lennard-Jones 12-6 functions can be successfully setup. GROMACS does not support mixed-function setting. For example, a model consists of both Lennard-Jones and Buckingham potentials for the vdw interactions. DL_FIELD can still produce GROMACS files for such systems, although *grompp* will generate an error when trying to produce a *tpr* file. However, such systems can be run using the corresponding FF files produced for DL_POLY.
(5) The rigid body or the [***RIGID***] directive in DL_FIELD, is not supported for GROMACS. The exception would be for the water models. In this case, the GROMACS directive command [settles] will be used (see also Example case below for water molecules).
(6) Comparison of single-point energies produced between DL_POLY and GROMACS are likely to be different. This is because both packages use different scaling precisions and distance unit. For example, DL_POLY uses Å, whereas GROMACS uses nm as unit distance. Furthermore, system coordinates in GROMACS' *gro* file

are fix-formatted to three significant figures. On the other hand, DL_POLY always express the coordinates in six-decimal precision in the *CONFIG* file.

For instance, consider CHARMM36 CGenFF for carbon dioxide with the equilibrium bond distance of 1.170 Å, and consider an interatomic distance for C-O bond is 1.163 Å. DL_FIELD will convert this distance to nm and truncate to three-significant figure in the *gro* file as 0.116 nm. Subsequent (harmonic) bond energy calculations would give a disparity of about a factor of two.

14.4 Example case: organic molecules

This example illustrates conversion of phenylalanine enclosed in a β-cyclodextrin (structure shown below) and solvates the whole system with ethanol in a cubic box of size 80 Å. All H-containing bonds are also constrained.



The structure file is in the *Examples/* folder as *cyclodextrin_phe.xyz* and illustrates below. The molecular system is separated into two Molecular Groups, CYC refers to the cyclodextrin cage and PHE for the organic molecule enclosed within the cage.

```
170
Cyclodextrin with PHE.
# MOLECULAR_GROUP CYC
C  -6.049000  0.945000  0.569000
C  -6.341000  2.134000  -0.313000
C  -5.113000  3.000000  -0.453000
C  -4.574000  3.388000  0.915000
C  -4.429000  2.153000  1.811000
C  -4.114000  2.550000  3.250000
C  -4.381000  -3.897000  -0.680000
C  -5.277000  -3.152000  -1.645000
C  -5.119000  -1.651000  -1.519000
C  -5.285000  -1.237000  -0.064000
…
…
H  -2.830000  6.347000  -2.095000
H  0.359000  5.982000  -1.647000
H  0.509000  6.438000  4.356000
# MOLECULAR_GROUP PHE
C  -0.830000  0.683000  2.322000
C  -0.114000  -0.185000  3.052000
C  0.873000  -0.882000  2.472000
…
…
```

To carry out this conversion, the following shows the relevant commands in a DL_FIELD *control* file.

```
DL_FIELD control file for cyclodextrin-phe.
1       * Construct DL_POLY output files
gromacs       * Seconday output files (gromacs, chemshell or none).
0       * produce dl_field.udff model file
opls2005  * Type of force field require (see list below for choices).
kcal/mol  * Energy unit: kcal/mol, kJ/mol, eV, or K.
normal  * Conversion criteria (strict, normal, loose)
1       * Bond type (0=default, 1=harmonic , 2=Morse)
1       * Angle type (0=default, 1=harmonic, 2=harmonic cos)
none   * Include user-defined information. Put 'none' or a .udff filename
1      * Verbosity mode: 1 = on, 0 = off
cyclodextrin_phe.xyz   * Configuration file.
none   * Output file in PDB. Put 'none' if not needed.
0 40 molecules  9.0 * Solution Maker: on/off, density, unit, cutoff)
…
…
1       * Constrain bonds? 1 = Yes (see below) 0 = No
…
…
1       * Periodic condition ? 0=no, other number = type of box (see below)
80.0   0.0   0.0 * Cell vector a (x, y, z)
0.0   80.0  0.0   * Cell vector b (x, y, z)
0.0   0.0  80.0 * Cell vector c (x, y, z)
default   * 1-4 scaling for coulombic (put default or x for scaling=x)
default   * 1-4 scaling for vdw (put default or x for scaling=x)
0  300.0  * Include velocity? 1=yes, 0=no and scaling temperature.
1        * Position solute at origin? 1 = yes, 0=no
etoh  2.0 default * Solvate model? none or specify solvent (see below) and distance criteria.
…
…
#########################################################
Atom state specification:  type  Molecular_Group   filter   [value]

CONSTRAIN  PHE    h-bond
CONSTRAIN  CYC    h-bond

#########################################################
```

After conversion, DL_FIELD produces three itp files: one for CYC, one for PHE and the other for the ethanol solvent (ETOH). They are included in the topology file, *gromacs.top*, as follows:

```
;
; Gromacs system topology (top) file.
; Produced from DL_FIELD v4.11
;

[ defaults ]
; nbfunc   comb-rule   gen-pairs   fudgeLJ fudgeQQ
1      1          no         0.500000     0.500000

[ atomtypes ]
; atom_type  at.num   mass   charge   ptype   c6   c12
CAO         6  12.01150    0.000   A  2.030504e-03  3.732606e-06
CT          6  12.01150    0.000   A  2.030504e-03  3.732606e-06
OAL         8  15.99940    0.000   A  2.624389e-03  2.420782e-06
OAS         8  15.99940    0.000   A  1.393695e-03  8.290022e-07
OS          8  15.99940    0.000   A  1.393695e-03  8.290022e-07
HC          1   1.00797    0.000   A  1.225781e-04  2.992630e-08
HO          1   1.00797    0.000   A  7.845000e-09  1.225781e-16
CA          6  12.01150    0.000   A  2.344876e-03  4.693426e-06
HA          1   1.00797    0.000   A  1.008475e-04  2.025617e-08
CT1         6  12.01150    0.000   A  2.030504e-03  3.732606e-06
C           6  12.01150    0.000   A  4.886845e-03  1.358990e-05
NT          7  14.00670    0.000   A  3.674381e-03  4.745346e-06
O           8  15.99940    0.000   A  2.363857e-03  1.589906e-06
OH          8  15.99940    0.000   A  2.074092e-03  1.512013e-06
H           1   1.00797    0.000   A  7.845000e-09  1.225781e-16

#include "gromacs1.itp"
#include "gromacs2.itp"
#include "gromacs3.itp"

[ system ]
; Title
Control file, for version 4.11

; System composition
[ molecules ]
; Molecular_group   species #
CYC     1
PHE     1
ETOH     5212
```

14.5 Example case: water molecules

There are two ways to setup an FF model for water molecules: constrained bonds and use of SHAKE/RATTLE algorithms in molecular simulations. The other method is to setup water as rigid molecules. DL_FIELD can produce FF files for DL_POLY for both methods. However, DL_FIELD can produce FF files for GROMACS using the bond constrain method. For rigid model, DL_FIELD will use the [settles] directive for GROMACS.

To setup constrained water molecule models (say, TIP3P model), the following [***CONSTRAIN***] directive must be used in a DL_FIELD *control* file and assign to Molecular Group that contains water molecules as follows:

```
…
…
0       * Tether atoms? 1 = Yes (see below)  0 = No
1       * Constrain bonds? 1 = Yes (see below) 0 = No
0       * Apply rigid body? 1 = Yes (see below) 0 = No
1       * Periodic condition ? 0=no, other number = type of box (see below)
…
…
#############################################################
Atom state specification:  type  Molecular_Group   filter   [value]
…
…
CONSTRAIN  W1 rigid_water
…
…

#############################################################
```

Note the term 'rigid_water' is the filter key for DL_FIELD to impose bond constrains and not to confuse with the [***RIGID***] directive. For other filter keys, please see Option **23** in Section 4.

After running DL_FIELD for GROMACS, the following itp file is produced:

```
[ moleculetype ]
; name    nrexcl
W1   3

[ atoms ]
; nr   type   resnr   residu    atom    cgnr    charge    mass
   1      OT    2 TP3O      O     1  -0.8340000   15.9994
   2     H3P    2 TP3O     H1     2   0.4170000    1.0080
   3     H3P    2 TP3O     H2     3   0.4170000    1.0080

[ bonds ]
; no bond

[ pairs ]
; ai   aj   funct   fudge_QQ  charge_i   charge_j   sigma  epsilon

[ constraints ]
; Bond constraints
    2     3   1     0.1514
    2     1   1     0.0957
    3     1   1     0.0957
```

Although this itp file can be used for simulation run, GROMACS has implemented the SETTLE algorithm specifically for water models, which is computationally more efficient. To use this algorithm, DL_FIELD will replace the *[constrains]* GROMACS directive with *[settles]*. This only works if the rigid body option (Option **24**) is switched on in the *control* file.

15 Force field models for LAMMPS

LAMMPS (Large-scale Atomic/Molecular Massively Parallel Simulator) is a general-purpose molecular dynamics software. Developed by Sandia National Laboratories, it is widely used to simulate atomistics and mesoscales models and contain a rich, albeit with large numbers, of commands and directives for a variety of simulation conditions and study purposes.

The following information assumes you have at least some basic understandings of LAMMPS file structures and commands. For more information about LAMMPS, please refer to the LAMMPS manual.

In DL_FIELD, there is an option to produce FF files for LAMMPS simulation runs. In most cases, DL_FIELD can setup a runnable set of LAMMPS FF files without further modification.

To use this feature, the keyword *lammps* must be used at Option **B** in the DL_FIELD control file.

Upon successful FF conversions, DL_FIELD will produce the following files:

(1) The system input file, *lammps.in*
(2) One or more data files contain structures and FF topologies, *lammpsX.data*, where X = 1, 2, 3...

DL_FIELD can produce several *lammpsX.data* files, where *X* correspond to the number of Molecular Groups specified in the configuration file. Each *data* file contains force field information for each Molecular Group. These *data* files are included in the *lammps.in* file, with all offsets automatically defined.

If LAMMPS program (*lmp*) is also installed in the computer, DL_FIELD can also run the program straight after FF model files are setup. For more details, please see Section 4.3.

15.1 Force field and parameter settings

LAMMPS contains numerous commands and different ways to achieve same FF definitions. To maintain consistencies, DL_FIELD adopts the following default rules to setup force field models for all FF schemes:

(1) Always use full, with energy unit and reconvert (if needed) to kcal/mol.
(2) Always use *interaction_style* hybrid, for example, *bond_style* hybrid harmonic, then explicitly define interaction style in the Interaction *Coeffs* commands in *lammps.data* file.
(3) *Kspace_style* command always use *pppm 0.0001*. That is, Particle-Particle Particle-Mesh (PPPM) methods, an optimized alternative to Ewald summation with *O*(N log N) scaling.
(4) Due to reason (3), when a rigid body is introduced for a molecule, intra-interactions are excluded based on the *special_bonds* command in the *input* file without the use of *neigh_modify* command.
(5) Both vdw and coulombic real space cut off are set to 12.0 angstrom. This value can be changed in the *pair_style* command in *lammps.in.*
(6) If bond constraints are applied, DL_FIELD defines these bonds by using the SHAKE algorithm with the following parameters as defaults: tolerance limit set to 10^{-5} and 20 as the maximum iterations for every timestep. This is defined in *lammps.in* file in *fix* directives.

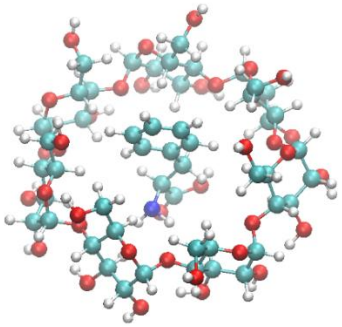
15.2 Restrictions

The following lists restrictions in using LAMMPS. Some of these restrictions will be removed in the subsequent future software releases.

(1) Multiple potential setting is not available to LAMMPS.
(2) FF schemes, such as CHARMM for proteins, that make use of CMAP feature, is not available to LAMMPS.
(3) The current version (4.13) does not support the following FF schemes: MISC_FF and INORGANIC FFs. DL_FIELD will return an error when one of these schemes are selected.
(4) Imposing FREEZE and TETHER states for atoms are not available to LAMMPS.

15.3 Example case

This example illustrates conversion of phenylalanine enclosed in a β-cyclodextrin (structure shown below) and solvates the whole system with ethanol in a cubic box of size 80 Å. All H-containing bonds are also constrained. Same example is used for GROMACS.



The structure file is in the *Examples/* folder as *cyclodextrin_phe.xyz* and portion of the file contents are shown below. The molecular system is separated into two Molecular Groups, CYC refers to the cyclodextrin cage and PHE for the organic molecule enclosed within the cage.

```
170
Cyclodextrin with PHE.
# MOLECULAR_GROUP CYC
C  -6.049000  0.945000  0.569000
C  -6.341000  2.134000  -0.313000
C  -5.113000  3.000000  -0.453000
…
…
H  -2.830000  6.347000  -2.095000
H  0.359000  5.982000  -1.647000
H  0.509000  6.438000  4.356000
# MOLECULAR_GROUP PHE
C  -0.830000  0.683000  2.322000
C  -0.114000  -0.185000  3.052000
```

To carry out this conversion, the following shows the relevant commands in a DL_FIELD *control* file.

```
DL_FIELD control file for cyclodextrin-phe.
1       * Construct DL_POLY output files
lammps       * Seconday output files (gromacs, chemshell or none).
0          * Write out dl_field.udff model file.
opls2005  * Type of force field require (see list below for choices).
kcal/mol  * Energy unit: kcal/mol, kJ/mol, eV, or K.
normal  * Conversion criteria (strict, normal, loose)
1       * Bond type (0=default, 1=harmonic , 2=Morse)
1       * Angle type (0=default, 1=harmonic, 2=harmonic cos)
none   * Include user-defined information. Put 'none' or a .udff filename
1      * Verbosity mode: 1 = on, 0 = off
cyclodextrin_phe.xyz   * Configuration file.
none   * Output file in PDB. Put 'none' if not needed.
0 40 molecules  9.0 * Solution Maker: on/off, density, unit, cutoff)
…
…
1       * Constrain bonds? 1 = Yes (see below) 0 = No
…
…
1       * Periodic condition ? 0=no, other number = type of box (see below)
80.0   0.0   0.0 * Cell vector a (x, y, z)
0.0   80.0  0.0  * Cell vector b (x, y, z)
0.0   0.0  80.0 * Cell vector c (x, y, z)
default   * 1-4 scaling for coulombic (put default or x for scaling=x)
default   * 1-4 scaling for vdw (put default or x for scaling=x)
0  300.0  * Include velocity? 1=yes, 0=no and scaling temperature.
1       * Position solute at origin? 1 = yes, 0=no
etoh  2.0 default * Solvate model? none or specify solvent (see below) and distance criteria.
…
…
############################################################
Atom state specification:  type  Molecular_Group   filter   [value]

CONSTRAIN  PHE    h-bond
CONSTRAIN  CYC    h-bond

############################################################
```

After conversion, DL_FIELD produces three data files (*lammpsX.data*, *X*=1 to 3): one for CYC, one for PHE and the other for the ethanol solvent (ETOH). They are indicated in the *lammps.in* input file, as follows:

Note that ATOM_KEYs are shown as remarks in the *pair_coeff* section, to indicate atom pairs id references. They are defined in the form *AB#X* where *AB* is the ATOM_KEY and *X* is the data file number to which the ATOM_KEY belongs.

For example, *pair_coeff* atoms 1 and 2 (highlighted below) refers to the vdw interaction between CAO and CT atoms that belong to the molecular structure defined in *lammps1.data.*

```
# LAMMPS input script, produced by DL_FIELD 4.12
# Title: Control file title. For DL_FIELD 4.2.
# Input structure: test_structures/cyclodextrin_phe.xyz

units         real      # in kcal/mol
dimension     3         # 3D simulation
atom_style    full      # molecular system with charges
boundary      p p p     # Normal periodic boundary
timestep      0.5       # in fs

bond_style    hybrid harmonic
angle_style   hybrid harmonic
dihedral_style  hybrid opls
improper_style  hybrid cvff
pair_style    hybrid lj/cut/coul/long 12.000000
kspace_style   pppm 1.0e-4
special_bonds   lj 0.0 0.0 0.500000 coul 0.0 0.0 0.500000

read_data lammps1.data &
  extra/atom/types       15  &
  extra/bond/types       15  &
  extra/angle/types      22  &
  extra/dihedral/types   19  &
  extra/improper/types    3  &
  extra/improper/per/atom 3 &
  extra/bond/per/atom    1
read_data lammps2.data add append offset 7 6 13 14 0
read_data lammps3.data add append offset 18 17 30 29 3


# Define Group-ID for atoms from the respective Molecular Groups.
group CYC type 1 2 3 4 5 6 7
group PHE type 8 9 10 11 12 13 14 15 16 17 18
group ETOH type 19 20 21 22

group bond_constraints union CYC PHE ETOH
fix shake_1 bond_constraints shake 0.000010 20 1 b &
  3 6 8 10 16 17 18 21

pair_coeff    1    1  lj/cut/coul/long   0.066000   3.500000 # CAO#1 CAO#1
pair_coeff    1    2  lj/cut/coul/long   0.066000   3.500000 # CAO#1 CT#1
pair_coeff    1    3  lj/cut/coul/long   0.105925   3.304542 # CAO#1 OAL#1
pair_coeff    1    4  lj/cut/coul/long   0.096125   3.185906 # CAO#1 OAS#1
pair_coeff    1    5  lj/cut/coul/long   0.096125   3.185906 # CAO#1 OS#1
pair_coeff    1    6  lj/cut/coul/long   0.044497   2.958040 # CAO#1 HC#1
…
…
```

16 Example Structures

The *Examples/* directory contains several sample structures. Look at the content of each file and convert these structures. Inspect the output files and see how DL_FIELD works.

You can insert the structure filenames in a DL_FIELD control file and run DL_FIELD. Alternatively, you can use predefined control files in the *control_files/* folder.

The *control_files/* directory contains a collection of *control* files named according to the example numbers (shown below). To convert these example structure, edit the *dl_f_path* and make sure the control file is pointed to the right location. For example, to convert the Example 5a structure (see below), edit *dl_f_path* and define the control file component as:

control = control_files/example5a.control

Then, run DL_FIELD. This control file reads in the *dodecylsulphate.pdb* file to produce the corresponding *dl_poly.CONFIG* and *dl_poly.FIELD* files in the *output/* directory.

You can change the options in each control file and inspect the outcome of the FF files produced. For example: different force field schemes, add solvents, etc. Note that not all force fields can work with every structures. An error will occur if certain information about a structure is not available for a chosen force field scheme.

With a correct set of options in the *control* file, DL_FIELD can generate FF files for DL_POLY for all example structures. This is not necessary the case for GROMACS and LAMMPS because not all features in DL_FIELD have been implemented just yet.

16.1 Example Files: PDB Format

16.1.1 Single FF models

(1a) *ethanol.pdb*
Contains a single ethanol molecule and an ion pair Na^+ and Cl^-, in a periodic cubic box of size 60x60x60 $Å^3$. DL_FIELD creates two Molecular Groups: ETOH and ION in force field files, as indicated in *ethanol.pdb*.

(2a) *sod1.pdb*
A superoxide dismutase protein structure. The protein consists of dimer orientated approximately 180 degree rotation relative to each other. Each monomer is given a Molecular Group names of SOD1 and SOD2 respectively. Notice that the protein contains histidine residues in various charge states. DL_FIELD can decide this automatically and select the appropriate parameters.

After setting up force field for the protein, the *control* file then solvates the system with TIP3P CHARMM water model and automatically add counter ions (sodium ions).

(3a) *6pti.pdb*
This is bovine pancreatic trypsin, an example protein structure illustrated in the Amber 9 manual. The control file uses CHARM36_prot as the FF scheme. Edit the control file and use other FF schemes such as opls2005 or opls_aam and see what happens. It demonstrates the ease of switching to different FF model schemes.

(4a) *b-All-a-Glc-OMe.pdb* (AMBER FF only)

A disaccharide methoxy-terminated structure. This is β-D-allopyranose-[1,3]-α-D-glucopyranose-1-OMe. DL_FIELD recognises the standard Glycam naming convention for the carbohydrate structure. The *control* file also instructs DL_FIELD to solve the system with TIP4P water model.

(5a) *dodecylsulphate.pdb*
A single simple membrane molecule, constructed by using some other molecular packages such as Chem3D. Note that the residue name has to change to SDS in the pdb file, which is the residue key for the molecule. The MOLECULE template can be found in the *.sf* file for OPLS2005 FF.

(6a) *ethanol_auto1.pdb, ethanol_auto2.pdb* and *alcohols.pdb* (CHARMM22_prot only)
These are examples of alcohol structures that make use of the auto-CONNECT feature. The *udff* file called *alcohol.udff* must be used alongside with these structures. Please read the remarks included in each file.

When the *udff* file is read, DL_FIELD converts the structures by using the ***MOLECULE*** ethanol template defined in the *udff* file, instead of using the standard ***MOLECULE*** ethanol template that is pre-defined in the library's *.sf* file (for CHARMM22_prot).

The difference between *ethanol_auto1.pdb* and *ethanol_auto2.pdb* files is that the ethanol molecules in the former file are grouped with different residue id, whereas, the latter file lumped both molecules as a single residue with deliberate mixing of the atoms between them. This is to demonstrate DL_FIELD can convert both configurations, unpicking the mixed atoms in the latter file into separate molecules.

The file *alcohols.pdb* contains two different types of alcohols but uses the same MOLECULE_KEY (residue key), ROH as defined in the *alcohol.udff* file. This example file illustrates the use of the auto-CONNECT feature to construct a general MOLECULE template that applies to a class of molecules.

The example control files (*control_files/example_6a.control*), can be reused for all three PDB structures. Change the input filenames in Option **8** (line 11) of the *control* file and see what happens.

(7a) *ethanol_shell.pdb* (CHARMM22_prot only)
An example use of the core-shell model in the organic force field. Here, the hydroxyl oxygen is assigned to core and shell components to mimic the polarizability effect of the atom. The *alcohol.udff* file must be used to convert this structure, which contains the ***MOLECULE*** ethanol_sh template that is needed for the conversion. The residue label ESH would be needed in the pdb file, so that DL_FIELD will look for the corresponding MOLECULE template defined in the *udff* file.

(8a) *methanol.pdb* (CHARMM22_prot only)
A single methanol molecule. The corresponding *udff* file (*alcohol.udff*) illustrates the overriding of the original ***MOLECULE*** methanol template in the *.sf* library file.

DL_FIELD reports the override in the dl_field.output file as follows:

```
- MOLECULE_TYPE 'methanol' has been overrided by UDFF file:
  Standard .sf file: key = MEOH  mass = 32.042000
  User redefinition: key = MeOH  mass = 32.042000
  Note: The MOLECULE_KEY has changed. Make sure you have matched KEY in your
configuration file.
```

Note that the residue label is redefined as 'MeOH'. This new residue label must also use in the pdb file.

Repeat DL_FIELD run without the *udff* file and see what happens. You need to edit the *example8a.control* file to do this (put *none* in Option **6** or line 10).

(9a) *mgo.pdb and mgo_shell.pdb* (INORGANIC_binary_oxide force field)
Example use of the INORGANIC force field using the self-CONNECT feature. The *mgo.pdb* is a rigid ion model, whereas, *mgo_shell.pdb* is the core-shell model of magnesium oxide. You need to edit *example9a.control* to change the input filename.

(10a) caco3.pdb (INORGANIC_ternary_oxide force field)
An example use of a more complicated force field for inorganic systems. This is aragonite (calcium carbonate mineral) in water (SPC). The structure is converted by using the ***MOLECULE*** calcium_carbonate1 template, which contain the auto-CONNECT and ***ANGLE ONLY*** features for DL_FIELD to automatically set up bonds on the carbonate ions only. To convert this structure, the periodic boundary condition (Option **25**), has to set to *auto*. This will instruct DL_FIELD to read the cell parameter information (CRYST1) in the PDB file.

(11a) *alkane.pdb* (CHARMM19 united atom model)
A C18 alkane chain, making use of the MOLECULE alkane defined in the *DLPOLY_CHARMM19.sf* file. The MOLECULE alkane has the auto-CONNECT feature, which can be used to convert any united-atom alkanes.

16.1.2 Multiple FF models

The following examples illustrate the use of multiple potential schemes in a molecular system. These files are just some examples to show the capability of DL_FIELD. The FF implementations are correct but that does not mean the subsequent model set up is valid!

To convert the following structures, set the force field scheme require, Option **1**, to *multiple* in the *control* file. This instructs DL_FIELD to read the input files and look for the ***POTENTIAL*** directives. In addition, Options **18** and **19** (lines 22 and 23) must be defined, to indicate the type of mixing rules between the different potential schemes in the model.

(12a) *multiple_potential_1.pdb* (AMBER, OPLS2005 and CHARMM22_prot - multiple)
Apply three different potential schemes on five methanol molecules: The first molecule is assigned to the AMBER, the following three molecules to the OPLS2005 and the last one to the CHARMM22_prot. The vdw parameters (sigma and epsilon) between two different force fields are derived using the Fenders-Halsey mixing scheme.

(13a) *multiple_potential_2.pdb* (CHARMM22_prot and INORGANIC_binary_oxide FF - multiple)
A simple example of mixed organic-inorganic system. The single ethanol molecule is assigned to CHARMM22_prot and the magnesium oxide cluster is assigned to rigid ion FF model from INORGANIC_binary_oxide. The PDB file shows that the ethanol molecule is assigned to a Molecular Group called 'ORG', whereas, Molecular Group is undefined for the magnesium oxide. For this reason, DL_FIELD assigns the inorganic model to a default Molecular Group called 'not_define' in the *dl_poly.FIELD* file.

Note that vdw parameters for organic-inorganic atoms pairs were not defined. However, these parameters can be defined by making use of the ***VDW_FIX*** directive.

An *udff* file (*vdw_fix.udff*) is included in the *Examples/* directory to illustrate the use of the ***VDW_FIX*** directive, for Examples (12a) and (13a). Repeat the conversions but this time

include the *udff* file and see what happens. To do this, insert the path *Examples/vdw_fix.udff* in Option **6** (line 10) of the control file (*example13a.control*).

When the ***VDW_FIX*** is applied to the Example 13a structure, the vdw parameters for the atom pair HA…CT3 are reassigned to that of the Morse potential in the CHARMM22_prot FF. Additionally, the vdw atom pair OH1…Mg3 (mixed organic-inorganic) is now assigned with a set of the Lennard-Jones parameters. Without the ***VDW_FIX***, DL_FIELD would have left it blank for users to manually insert the parameters.

Inspect the *dl_poly.FIELD* files produced, with and without the *udff* file and note the differences.

The use of different CHARMM FF components in a model can be demonstrated by revisiting the *sod1.pdb* file (Example (2a)). Inspection of the file shows that two ***POTENTIAL*** schemes have been assigned: The SOD1 Molecular Group has been assigned to CHARMM22_prot, while SOD2 Molecular Group has been assigned to the CHARMM36_prot force field. When a force field scheme is specified in Option 1, these directives will be ignored. When setting the Option **1** to *multiple*, then DL_FIELD will assign the CHARMM22 force field to the first protein monomer (SOD1) and CHARMM36 to the second protein monomer (SOD2), according to the ***POTENTIAL*** directives specified in the PDB file.

(14a) *ionic_liquid_species.pdb* (OPLS_CL_P)
Contain two ionic liquid components, fluoroalkylimidazolium cation with tetrafluoroborate anion. Atom labels are expressed in DL_F Notation in the *CONFIG* and *FIELD* files.

(15a) *nucleic.pdb* (CHARMM36_nucl)
Example structure of DNA, which consists of adenine and thymine base pairs and –OH terminations at the end of strands at the 3- and 5- positions of the ribose units.

16.2 Example Files: *xyz* Format

The following examples illustrate the use of the *xyz* file format. All ATOM_TYPEs are expressed in the standard DL_F Notation. This is shown in *dl_field.output* file under the heading:

ATOM_TYPEs mapping (DL_F notation):

A separate file, *dlf_notation.output* in the *output/* directory, is also created and contains user atomic structure expressed in DL_F Notation. Try Option **12** (line 15) to interchange atom labels into either force field-specific format and DL_F Notation. Notice the difference in atom labels in *dl_poly.FIELD* and *dl_poly.CONFIG* files.

(1b) *amides.xyz*
An example system contains primary, secondary and tertiary amides, expressed in DL_F Notation. Inspect the *dlf_notation.output* in the *output/* folder.

(2b) *alkyl_ammonium.xyz*
An ammonium compound, together with a chloride anion. Note: the cation, N^+, located within the molecule does not need the positive sign. Whereas, the *free* chloride ion, Cl, does. The structure is solvated with water (SPC/E model), from the *control* file.

(3b) *favipiravir.xyz* (OPLS2005 FF)

This is an antiviral drug against influenza and the drug molecule is solvated with dimethyl sulphoxide from the *control* file.

(4b) *dynemicin_a.xyz* (PCFF)
This is an anti-cancer drug, dynemicin A.

(5b) *mgo.xyz* (inorganic_binary_oxide)
This is the same structure as that of *mgo.pdb* (Example 9) but is expressed in the *xyz* format. For inorganic systems, both the ***MOLECULE_KEY***, which indicates the type of the inorganic material, and the ***MOLECULAR_GROUP***, to which the material belongs, must be explicitly defined in the file. Note that, element symbols can be used, instead of the ELEMENT_KEYs as in the *mgo.pdb* file.

Initially, ***MOLECULE_KEY*** MO3 is used, which is a rigid-ion model. One can use the same configuration file to setup a core-shell model by simply changing the ***MOLECULE_KEY*** to MO1. DL_FIELD will automatically add the necessary core (or shell) components into the system and set up the force field.

(6b) *caco3.xyz* (inorganic_ternary_oxide)
This is the same structure as that of *caco3.pdb* (Example 10a) but is expressed in the *xyz* format. The system is divided into two Molecular Groups: *A1*, which is assigned to calcium carbonate, and *WAT*, which is assigned to water molecules. The ***MOLECULE_KEY***s *CC1* and *SPC* are assigned to each group, of molecules, respectively. Same potential scheme is used for both groups of molecules (***POTENTIAL*** inorganic_ternary_oxide). The keyword 'multiple' must be used for the FF scheme in the *example_6b.control* file. Note that element symbols can be used, instead of the ELEMENT_KEYs as would be the case in the *caco3.pdb* file.

(7b) *multiple_potential_3.xyz* (OPLS2005 and CVFF - multiple)
Illustrate multiple potential capability in *xyz* format. The example configurations consist of several methanol molecules. The positions of the ***POTENTIAL*** directives in the file indicate the extent of the force field schemes CVFF and OPLS2005. The CVFF methanol molecules are divided into three Molecular Groups: GRP1, GRP2 and GRP3, whereas the OPLS2005 methanol molecule belongs to Molecular Group GRP4. Inspect the *dl_poly.FIELD* and *dl_poly.CONFIG* files and see how they correspond to the input *xyz* file.

(8b) *multiple_potential_4.xyz* (OPLS2005 and inorganic_binary_oxide)
Illustrate multiple potential capability in *xyz* format with mixed inorganic-organic potential schemes. The structure is same as the *multiple_potential_2.pdb* file in the Example 13a. For the organic part, the ethanol molecule is assigned to the OPLS2005 force field, whereas, the inorganic part is assigned to inorganic_binary_oxide force field. The ***MOLECULE_KEY*** directive must also be defined for the inorganic force field, to indicate the type of material. For organic force field, the directive ***MOLECULE_KEY*** has no effect, and the molecule will be automatically determined by DL_FIELD.

(9b) *multiple_potential_5.xyz* (inorganic_binary_oxide and amber16_gaff)
This example illustrates the use of a template-base force field in *xyz* format. DL_FIELD will automatically identify the molecules, rearrange, and match with the MOLECULE template in the library files. Moreover, DL_FIELD will detect the use of core-shell model on the inorganic material and append the model automatically.

(10b) *ionic_liquid_species.xyz* (OPLS_CL_P)

Another example to show the use of a template-base force field in *xyz* format. The file contains a guanidinium cation with dicyanamide anion. The atom labels are expressed in the standard CL&P force field in the *CONFIG* and *FIELD* files.

(11b) *cyclodextrin_phe.xyz* (OPLS_2005)
This example illustrates concurrent production of DL_POLY and GROMACS FF files. The structures are separated into two Molecular Groups: CYC and PHE, with correspond to β-cyclodextrin and phenylalanine molecules respectively. The *control* file instructs DL_FIELD to solvate the system with ethanol molecules. This results in three itp files (*gromacsX.itp*, where *X* = 1, 2 and 3) being produced for GROMACS: one for CYC, one for PHE and the other is for the solvent, ETOH.

Alternatively, if *lammps* is inserted into the control file, then LAMMPS output files will be generated.

16.3 Example File: *mol2* Format

The *mol2* is the latest file format first made available in version 4.5. Only a subset of features is made available for the *mol2* configuration files. For instance, *mol2* files do not allow the setting up of a multiple potential models. However, the use of *.mol2* is one of the convenient ways to set up Amber GAFF force field models, a popular general force field for general organic molecules such as those of small drug molecules. The *.mol2* file can be readily produced using the Amber's *antechamber* program suite.

(1c) *ascorbic_acid.mol2* (AMBER_GAFF and OPLS2005)
This example file was produced from a starting PDB structure using the Amber's *antechamber* program and charges were assigned using the BCC-AM1 method. DL_FIELD will detect if the atom keys contained in the file belong to Amber GAFF. It will not carry out the conversion process. Instead, the atom keys and charges will be used, *per se*, and setup the force field models for DL_POLY.

If the force field scheme is changed to OPLS2005 (Option **1**) and run DL_FIELD again, then DL_FIELD will realise the atom keys in the *mol2* file do not belong to that of OPLS2005. In this case, normal conversion process according to the *xyz* procedures will be carried out, including the charges. DL_FIELD ***will not*** take the charge values from the *mol2* file.

In both instances, a *dlf_notation.output* file will be produced, showing the actual chemical types for each atom in the system.

In addition, the option to write out the udff user model file is also switched on. This instructs DL_FIELD to create a *dl_field.udff* file, which contains all force field information of the system. This file can be read in by DL_FIELD to create the FF model files without referring to the library files.

17 FAQ

1. DL_FIELD has reported a successful conversion of my configuration file, but I notice that not all molecules in my file have been converted.

Answer: DL_FIELD will terminate when the END keyword is encountered in your PDB file. These keywords may sometime appear in the middle of your file. Remove them and rerun DL_FIELD.

2. DL_FIELD fails to convert my protein file and report the following error:

'Error: c2 s1 s1 angle parameter not found'

What happens?

Answer: Identify the atom index numbers that give rise to this error and refer them to your file. This error could possibly arise because DL_FIELD detect a possible disulphide linkage between cysteine residues and yet they are in reduced form (-SH). Remove the thiolate hydrogen from both residues and rerun DL_FIELD.

3. DL_FIELD reports the following type of error:

Matching of target atom *a* = *element* failed. Program exits. config_pos=*some number*

****

Fail to match atoms for *molecule* of id=X. Program exit.

Answers: This only occurs in matching of MOLECULE templates in PDB files. There are few possibilities to these types of error:

(a) The connectivity information (***CONNECT*** directive) mismatches with the bond connectivities of the user's configuration. A particular pair of atoms within a user's molecule may be too far apart.

Turn on verbosity mode from *dl_field.control* file and rerun DL_FIELD. Look into *dl_field.output* file. The offending molecule gives rise to this error is usually located at the end of the file. Check for the user config. neighbour list information and compare with the CONNECT statements of the molecule.

If there is a mismatch of neighbour list information, identify the particular missing pair and increase the bond length correspondingly. For instance, if DL_FIELD failed to identify C-O bond (neighbour pair) in the molecule, increase the C_O threshold value (#define C_O ....) in *dl_field.h* program.

Alternatively, a poorly or non-equilibrated structure may also give rise to this error due to having two non-bonded atoms being too close to each other and DL_FIELD misidentified them as a bonded entity.

(b) Wrong element symbols. If elements are not explicitly defined in *element symbol* columns 77-79 of the user's PDB file then DL_FIELD will attempt to extract the element symbols from the *atom_label* (columns 13-16). Quite often these columns

contain other atom labels. While some measures have been programmed to resolve these, it is not failed safe.

4. DL_FIELD fails to recognise some of the amino acid residues such as those with various charge states such as histidine. Why?

Answer: One possible common mistake is the use of potential-specific notation that works in one potential scheme but the notation is not recognised in the other potential scheme. For instance, a delta-protonated histidine is denoted as 'HSD' in CHARMM, while this is denoted as 'HID' in AMBER. It is advisable to use the standard 'HIS' notation and let to determine the charge state and assign the parameters according to the potential scheme chosen.

5. When comparing with other programs, I can't get the coulombic or vdw energy in agreement. Why?

Answers: The coulombic energy component in DL_FIELD includes both the long-range interatomic charge-charge interactions as well as the 1-4 intra-interactions. DL_FIELD does not calculate the pair energy beyond the cut off specified. For this reason, the coulombic energy component should be used as a reference only. To improve the accuracy of the energy, try to use large cut off values.

However, discrepancies can still arise due to slight variations of fundamental constants. Subsequently, slightly different coulombic constants being defined for scaling the coulombic energy in different programs. In DL_FIELD, this constant is displayed in the energy component of *dl_field.output* file, which can be different from those of DL_POLY programs, or indeed, any other MD programs.

The coulombic constant can be adjusted by redefining the fundamental constants (electric permmitivity, elementary charge value etc) located in the *dl_field.h* header file. Nevertheless, such adjustments are a matter of artistic refinement that should not significantly influence the outcome of a simulation.

In the case of vdw, the energy variation is largely due to two factors: (a) rounding errors of the vdw parameters used and (b) the way cut off distance being treated. DL_FIELD does not calculate pair-energy beyond the cut off specified. For this reason, the vdw energy component should be used as a reference only.

In DL_POLY MD package, the vdw cut off correction may be treated differently compare with other programs. However, the 1-4 vdw interactions are usually in good agreement due to the fact that the 1-4 distances are usually within a typical cut-off range specified.

To improve the accuracy of the energy, try to use large cut off values so that all possible atom pairs are included.

6. I have set up the necessary molecular models and potential parameters but I still can't get DL_FIELD to convert the structure.

Answer: There can be many reasons for this but one common mistake is that the information in the user PDB file is not in the proper location columns. Consult Chapter 6 to make sure the information in the PDB file is in the correct columns and order.

7. I have defined new parameters in the *udff* file, but DL_FIELD seems to read this information incorrectly.

Answer: Before you insert your own parameters, make sure you have consulted the standard *.par* file for the potential scheme of interest. The sequence and the number of parameters can be very different among the different schemes.

The sequence and the number of parameters appear in *udff* file must follow strictly that of the corresponding *.par* of a given potential scheme. For instance, AMBER requires six parameters to define a set of dihedral bond. All six parameters must be listed in the *udff* file and in the same order. Two of the parameters, namely the 1-4 vdw and coulombic scaling factors are listed in AMBER, whereas, they are not listed for the other potential schemes.

8. DL_FIELD fails to convert my *xyz* structure and gives an error something like 'unknown CG'. What happen?

Answer: It means DL_FIELD cannot recognise certain arrangement of the atoms in your molecule. Email me and I may be able to include the missing *CG* into DL_FIELD. Another possibility is that DL_FIELD failed to make correct determination the bond connectivity. Try to use a loose conversion in Option **3** and run DL_FIELD again.

End of DL_FIELD version 4.13 user manual

C W Yong, June 2026

Please quote the following reference in your publication:

C W Yong, 'Descriptions and Implementations of DL_F Notation: A Natural Chemical Expression System of Atom Types for Molecular Simulations', *J. Chem. Inf. Model.* **56**, 1405–1409 (2016)