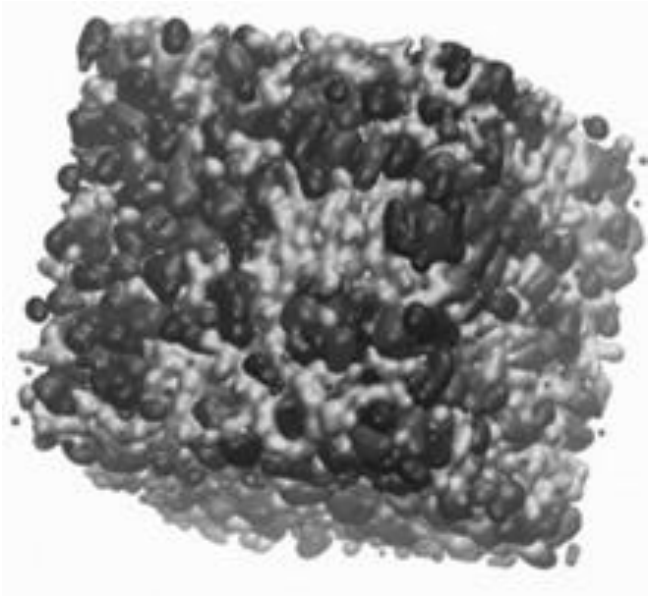


DLPGEN v.3.1.1

Carlos E. S. Bernardes

CENTRO DE QUÍMICA ESTRUTURAL
INSTITUTE OF MOLECULAR SCIENCES
DEPARTAMENTO DE QUÍMICA E BIOQUÍMICA
FACULTY OF SCIENCES, UNIVERSITY OF LISBON

22/6/2023

DLPGEN VERSION 3.1.1

Carlos E. S. Bernardes
cebernardes@ciencias.ulisboa.pt

ACKNOWLEDGMENTS

This work was supported by Fundação para a Ciência e a Tecnologia (FCT), Portugal (projects 2021.03239.CEECIND, PTDC/QUI-OUT/28401/2017, LISBOA-01-0145-FEDER-028401, UIDB/00100/2020, and UIDP/00100/2020).

** INDEX **

1. DESCRIPTION

DLPGEN is a Fortran program that produces input files for DL_POLY, GROMACS, LAMMPS, and CHARMM. The program can create simulation boxes with the molecules arranged in a crystalline structure or dispersed in an expanded matrix. To build crystalline materials, data from single-crystal X-rays are required (i.e., unit cell parameters, symmetry operations, and molecule-reduced coordinates). For this purpose, the program reads COORD files, that can be retrieved from the Cambridge Structural Database,[1] or can be manually built using any other data source (e.g., CIF files). As an alternative to these options, PACKMOL[2] can also be automatically called from DLPGEN to create simulation boxes for liquids.

If you find this software useful for your research, please cite:

C.E.S. Bernardes; J. Chem. Inf. Model. (2022) 62, 1471-1478.
https://doi.org/10.1021/acs.jcim.1c01431.

The version currently available is working, however, it is not free from bugs. Thus, it is recommended to check the output files before performing the simulations. Any problems or questions do not hesitate to contact me.

DLPGEN is free software, distributed under the terms of the GNU General Public License as published by the Free Software Foundation and included in the source code documentation. This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY, without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

In no event, the author will be liable to you for damages, including any general, special, incidental, or consequential damages (including but not limited to arising out of the use or inability to use the program, to loss of data or data being rendered inaccurate, or losses sustained by you or third parties, or a failure of the program to operate with any other programs), even if the author has been advised of the possibility of such damages.

2. INSTALLING AND RUNNING

After downloading the program, the file can be extracted by using:

```
$ tar -xvf dlpgen_3_X.tar
```

The program is installed from the "src" directory by typing:

```
$ cd dlpgen_3_X_XX/src
$ make cygwin
```

or

```
$ make linux
```

The first option allows installing the program on Windows computers with CYGWIN, while the second can be used to install it under Linux, MAC-OS, or using the Windows Subsystem for Linux (WSL). Both options require the GFORTRAN compiler. Please note that the program was not tested with other compilers. After this process, the program will be placed in the "bin" folder inside the program directory. It is advised to copy the program to a local *bin* directory (e.g., /usr/local/bin or ~/bin). Alternatively, it is possible to edit the *bin* location in the *makefile* before compilation, by changing the "BINROOT" option.

The program runs by typing:

```
$ dlpgen [Arguments]
```

The following arguments can be used with the program:

-chrm	Create input files for CHARMM.
-cor	Use "*.cor" file to generate input files for a crystal structure, i.e., converts *.cor files to *.gen.
-conf	Make a configuration file.
-cryst	Reads symmetry operations in an IFCRYST section in the GEN file, to create a solid structure.
-datalmps [FILE]	Reads the position of the atoms in the simulation box from a LAMMPS data file.
-dens [VALUE]	Set a target density value (g/cm3) of the simulation box. This option requires PACKMOL.[2]
-dlp	Create input files for DLPOLY (Default).
-drude	Read Drude information in an IFDRUDE section of a GEN file.

-extra	Set extra calculations in the LAMMPS input files, to compute average energies, temperatures, and simulation box sizes.
-f [FILE]	DLPGEN **P**rogram **C**ontrol **F**ile (Default: "PCF").
-gmx	Create input files for GROMACS.
-gro [FILE]	Use GROMACS *.gro files to create the initial system configuration (useful to add DRUDE particles to a previously equilibrated configuration).
-h	Help!
-lmps	Create input files for LAMMPS.
-ljmix	Explicitly calculate the mixing LJ parameters for all interactions in the GROMACS topology.top file.
-nobad	Do not compute **b**onds, **a**ngles, or **d**ihedrals.
-noconnect	Do not compute **connect**ivity.
-noconst	Do not compute **const**raints.
-nocheck	Do not check the simulation box integrity.
-noctl	Do not write standard CONTROL or *.MDP files.
-pack	Use Packmol[2] to create a simulation box using the box dimensions set in the PCF file (this argument is not required if *-dens* is used).
-pdb	Convert *.pdb files to *.gen.
-pdb_cfg [File]	Use *.pdb file as the initial simulation box configuration.
-poly	Build a Z-matrix for a polymeric chain.
-print_fld	Print Force-Field file.
-revcon	Uses a DL_POLY, REVCON file from a previous run as an initial simulation box (useful to add DRUDE particles to a previously equilibrated configuration).
-tol	Packmol tolerance Value (Default: 2.5).
-v	Verbose mode.
-ve	Verbose mode with steroids!
--version	Program version information.

-xyz	Convert *.xyz files to *.gen.
-xyz_zmat [File]	Builds a molecule from a Z-matrix using reference coordinates for the first three atoms. The coordinates are recorded in FILE.
-zmat	Convert a Z-matrix (*.zmat) into a *.gen file.
-14excl	Compute 1-4 interactions for shell-shell and shell-core interactions (if DLPOLY Classic is to be used).

If the program runs with no additional arguments, the **P**rogram **C**ontrol **F**ile named "PCF", which must be in the pattern folder, will be used to create input files for DL_POLY. Different named PCF files can be selected as:

```
$ dlpgen –f file
```

To create input files for other MD packages, e.g., GROMACS, use:

```
$ dlpgen -gmx
```

Like any program, DLPGEN is not free from errors, particularly while setting input files for new molecules. Thus, it is recommended to check the program's behavior. For this use, the arguments "-v" or "-ve" run the verbose mode and get access to detailed information during the program execution.

The first step of the program is the establishment of the connectivity of the atoms for each molecule based on the data in the *GEN file (see below). In principle, any atom can be recognized. The **program is case sensitive**, so that, an atom named "CS", will be recognized as a carbon atom while "Cs" will be considered as a Cesium atom. By default, connectivity criteria in the code recognize several bonds in molecules automatically, particularly for organic compounds. If a bond for a given pair of atoms is not available, this data will be prompted to the user, and the result recorded in the file:

```
$ ~/.dlpgendatabase
```

which can be edited if necessary. If the requested interaction makes no sense, a zero value should be used to ignore it.

When the program runs, the established connectivity is recorded in a file with termination *badin (e.g., molecule.badin). If necessary, it is possible to edit this file and run the program as:

```
$ dlpgen –noconnect
```

This argument allows bypassing the routine that computes the atomic connectivity, preserving the previously edited file. A similar procedure can also be used to edit the bonds, constraints, angles, dihedrals, impropers, and drude particles, which are written in a file with termination *badout (e.g., molecule.badout). In this case, DLPGEN should be run as:

```
$ dlpgen –noconnect -nobad
```

or just

```
$ dlpgen -nobad
```

The folder "examples" included with the program source code, provides several examples of how to use and set the input files for DLPGEN. These examples include:

1 – Input files for ethane using a polarizable force field.

2 - Input files for butane from a z-matrix.

3 - Input files for butane from a PDB file.

4 - Generate input files for a crystal structure of benzoic acid from a COR file.

5 - Generate input files for a 4'-hydroxyacetophenone (HAP) hydrate. In this case, the initial data in a COR file was converted into a *gen file. Because this hydrate contains 1.5 water molecules and one HAP in the unit cell asymmetric unit, the coordinates in the converted *gen* file for water and HAP were used to obtain three different files (given in the mols directory). In each file, the appropriate symmetry operations were included for each molecule.

6 – Obtain a CONFIG file for liquid water using Packmol and assuming a density of 0.9 $g \cdot cm^3$.

7 – Obtain a CONFIG file for liquid water using Packmol and a previously defined simulation box.

8 - Create input files for LAMMPS, GROMACS, and CHARMM.

Input files for several molecules (e.g., ionic liquids) can also be found and downloaded from https://bit.ly/2Lb0xAz. Video tutorials with step-by-step descriptions on the use of DLPGEN are also available at https://bit.ly/3qwDlNp.

3. SIMULATION BOX & THIRD-PARTY SOFTWARE

DLPGEN can produce all input files to run DLPOLY, GROMACS, LAMMPS, and CHARMM. To set the initial system configuration, unless a crystal structure will be produced, the program uses small cubic boxes and puts one molecule in each box. Thus, to avoid the superposition of the molecules, the box dimensions set in the PCF file must be sufficiently large to contain the molecules. As a result, an expanded simulation box is produced. As an alternative, the more efficient procedure implemented in PACKMOL can be used. For this, PACKMOL must be downloaded from:

http://www.ime.unicamp.br/~martinez/packmol/home.shtml

compiled and the executables placed in a folder included in the operating system PATH (e.g., /usr/local/bin or ~/bin).

4. INPUT FILES

DLPGEN requires three main files to run: i) one containing the information to build the topology file and the initial simulation box; ii) the molecule reference coordinates, and iii) a file containing the force field parameterization (library file).

4.1 *PROGRAM CONTROL FILE (PCF)*

The following example shows how to set the PCF file for DLPGEN. This file contains all necessary information to run the program, i.e., the system composition, files folder location, and simulation box parameters.

```
# TITLE
Project Name

# FILES LOCATION

DIR             "mols/"

# MOLECULES IN THE SYSTEM (NAMES WITHOUT EXTENSIONS)

NUMMOLS         2

#               MOL NAME        No MOLECULES
MOLS_NAME       molecule_a      192
                Molecule_b      192

# SIMULATION BOX

a_cell          4.583
b_cell          8.825
c_cell          17.888
alfa            90.000
beta            90.770
gama            90.000

# no. cells repetitions
x_cell          8
y_cell          4
z_cell          2

# Force-field (e.g. opls.lib or cl_p.lib)

ff-file         "/lib/opls.lib"
drude_model     dyna

PACKMOL 1 fixed 0. 0. 0. 0. 0. 0.
```

The available directives for the main input file are:

Directive*	***Meaning***
# TITLE project title	Job title.
DIR "directory"	Directory path containing the input files and auxiliary output files generated during the program run.
NUMMOLS *n*	Number, *n*, of molecular types.
MOLS_NAME *NAME n*	*Name* and number, *n*, of molecules. This name will be used to locate the molecular files in the **DIR** folder path.
a_cell val	Length, *val*, of the crystal structure unit cell along the *a* axis or, the length of the simulation box along the *x* axis.
b_cell val	Length, *val*, of the crystal structure unit cell along the *b* axis or the length of the simulation box along the *y* axis.
c_cell val	Length, *val*, of the crystal structure unit cell along the *c* axis or, the length of the simulation box along the *z* axis.
alfa val	Angle α (*val*) of the unit cell/simulation box.
beta val	Angle β (*val*) of the unit cell/simulation box.
gama val	Angle γ (*val*) of the unit cell/simulation box.
CELL *a b c* α β γ	Unit cell/simulation box dimensions. This entry can be used to replace the six directives above.
x_cell n	Number, *n*, of unit cells to be stacked along the x axis (use this option to stack several small crystal unit cells, to build a large simulation box. The default is 1, for the simulation of liquids)
y_cell n	Number, *n*, of unit cells to be stacked along the y axis (use this option to stack several small crystal unit cells, to build a large simulation box. The default is 1, for the simulation of liquids)
z_cell n	Number, *n*, of unit cells to be stacked along the z axis (use this option to stack several small crystal unit cells, to build a large simulation box. The default is 1, for the simulation of liquids)

ff-file "file"	Path to the file containing the force field parameterization.
drude_model model	Drude/shell model to be considered while writing the FIELD file. "model" can be "dyna" or "relx", for the dynamic and relaxed models, respectively. For more details regarding these two models, please see the DL_POLY manual.
mix_rule LB	Use Lorentz-Berthelot mixing rules (i.e., uses the arithmetic mean for sigma). If not included, the default is the use of geometric means.
PACKMOL *n entry*	Set constraints for a given molecular type while using Packmol. ***n*** position of the molecular type to which constraints will be used. ***entry*** constraint defined as in the Packmol manual (https://bit.ly/3r0OQvE).

***** in bold are the key directives and the text in normal font represents a value/text that must be entered by the user.

Remarks

The *NAME* entry in **MOLS_NAME**, identifies each molecule and is used by DLPGEN to search for the *xyz, *pdb, *zmat, *cor, *.gen, *.badin, and *badout files (see details below) for each specie in the folder **DIR**.

a_cell, **b_cell**, **c_cell**, **alfa**, **beta**, **and gamma**, or the **CELL** directive are used to set a crystal unit cell dimensions or the simulation box size. The previous directives can be ignored if the **-dens** argument (see page 4) is used. The options above can also be used to build an expanded simulation box. In this case, a box with dimensions that can contain each molecular species should be specified.

If the output files aim at the simulation of a solid phase, the unit cell dimension obtained from the single crystal structure should be used as input. To replicate the unit cell along the x, y, and z-axis, **x_cell**, **y_cell**, and **z_cell** directives should be used, respectively. For example, consider the orthorhombic crystal structure with unit cell *a* = 4.583 Å, *b* = 8.825 Å, and *c* = 17.888 Å, by setting:

```
a_cell          4.583
b_cell          8.825
c_cell          17.888
alfa            90.000
beta            90.000
gama            90.000

# or:
# CELL 4.583 8.825 17.888 90.0 90.0 90.0
```

```
# no. cells repetitions
x_cell          8
y_cell          4
z_cell          2
```

In this way, DLPGEN will create a simulation box with sides x = 36.664 Å, y = 35.300 Å, and z = 35.776 Å. This allows, therefore, the creation of well-proportioned simulation boxes of crystal structures, which can accommodate large cutoffs. For liquids ignore these directives or use 1 in all parameters.

4.2 *COORDINATES FILE*

This file should be named with **MOLS_NAME**, as defined in the **PCF** file, followed by **.gen** (e.g., molecule.gen). It can contain three parts, of which only the first one is mandatory. Part I contains the atomic coordinates of the molecule that will be used to build the initial simulation box and to determine the connectivity. Part II contains additional information to create the starting configuration (e.g., symmetry operations in the crystal structure) and topology data (e.g., Drude particles, bonds, angles, and dihedral). Part III gives a representation of the atomic coordinates as a PDB file, for visualization with, e.g., RasMol, VMD, or Mercury (note that it is possible to open *gen files directly with these programs. However, this may not work if **IF** functions are defined in the file). Part I and III can be automatically generated by DLPGEN from a z-matrix, a Cambridge structural coordinates file (COORD), a *xyz, or a *pdb file (see below). If a COORD file is used (see details about these files in section 4.5), the symmetry operations of the crystal structure will be placed in the GEN file and, by default, DLPGEN will generate a simulation box that corresponds to a crystal structure.

To convert, for example, an *xyz file into *gen, type:

```
$ dlpgen -xyz
```

This procedure will look for files with the extension *.xyz that start with the names defined in the **MOLS_NAME** directive (e.g., molecule.xyz). It is likely that, while preparing the initial input files, the atom names do not match those in the force field library. To help in the assignment process, the atoms labels in Part III are changed to include their relative position (see below). This allows the easy identification of each atom using a visualization program (e.g., Mercury).

** Part I **

The first line on this part contains the number of atoms in the molecule. The following lines contain the atom's details in five columns: A- Atom number; B- Atom name (4 characters); C, D, and E with the x, y, and z coordinates, respectively.

- Example of Part I -

```
#  A  B    C                  D                  E
   17
    1 O1  0.336435025653E+01 0.118784500000E+01 0.484444921474E+01
    2 O2  0.503266700571E+01 0.215594750000E+01 -0.11649618316E+01
    3 N3  0.216665334516E+01 0.306668750000E+01 0.451527958226E+01
    5 C5  0.353020834190E+01 0.219654250000E+01 0.269960403706E+01
    6 C6  0.450288508798E+01 0.128668500000E+01 0.231228957567E+01
.......
```

** PART II **

This part is optional and includes additional information needed to build the topology files and the initial simulation box (e.g., symmetry operations, extra bonds, etc.). Each input is preceded by a key (**SYMM**, **CONSTRAIN**, **RIGID**, **BONDS**, **ANGLE**, **DIHED**, or **IMPROP**) followed by the corresponding number of entries (see example below). In the case of the constraints and bonds, it is possible to define if the additional records will be used during the connectivity analysis (the program routine that establishes the molecule angles and dihedrals) or simply added to the obtained result (keywords **noappend** and **append**, respectively). In the case of the additional angles and dihedrals, they will be simply appended to the connectivity data. For rigid units, the first number represents the number of atoms in the unit followed by their atomic positions in the molecule (column A in part I). Each value must be separated by one space and no spaces should be given in the line before the number of atoms in the rigid unit. Also, note that the use of rigid units is limited to DL_POLY. If symmetry operations are included, they follow the nomenclature used in the Cambridge Structural Database coordinate file format (section 4.5).

- Example of Part II -

```
# SYMMETRY OPERATIONS
SYMM        2
-1.0  0.0  0.0 0.00000   0.0 -1.0  0.0 0.00000   0.0  0.0 -1.0 0.00000
 1.0  0.0  0.0 0.00000   0.0 -1.0  0.0 -.50000   0.0  0.0  1.0 -.50000

# RIGID UNITS
RIGID 1
4 2 4 7 9

# EXTRA BONDS (at1, at2)
BONDS   5  append
1    8
1    9
1    10
1    11
1    12

# EXTRA CONSTRAINS (at1, at2)
CONSTRAINS   1  noappend
10   12


# EXTRA ANGLE (at1, at2, at3)
ANGLE  5
1    2    3
1    2    4
1    2    5
1    2    6
1    2    7
```

```
# EXTRA DIHEDRAL (at1, at2, at3, at4)
DIHED   5
5    2    1    8
5    2    1    9
5    2    1    10
5    2    1    11
5    2    1    12


# EXTRA IMPROPER (at1, at2, at3, at4)
IMPROP 1
1    2    3    4


# Drudes (position, polarizability, k, Charge, Thole)
DRUDES 2 DRUDMASS 0.1
    1    0.641     4184.00   0.00000   2.60
    2    0.641     4184.00   0.00000   2.60


# VIRTUAL SITES
VS2  1
5   1   2   1    0.7439756

VS3  1
5   1   2   3   1  0.7439756  0.128012

VS4  2
5   1   2   3   4   2   1.0   0.9   0.105
7   1   2   3   4   2   1.0   0.9   0.999

VSN  1
5   1   1   2   3   4
```

The Drude model details, start with the line:

```
DRUDES n DRUDMASS m
```

where *n* is the number of entries and *m* is the mass of the Drudes/Shells. Each line contains the properties of each particle:

- 1st: number of the atom to which the Drude/Shell will be appended.
- 2nd: atomic polarizability, α.
- 3rd: spring constant, k.
- 4th: Shell/Drude charge.
- 5th: atomic Thole factor.

The spring constant has the function form:

$$U = (k/2) * r^2 \quad (1)$$

where r is the distance between the atom core and the particle. The charge **or** the force constant of the harmonic oscillator in (1) must be provided. If one of these properties (charge or force constant) is set to zero, it will be computed using the relation:

$$q_{drude} = -\sqrt{4 \cdot \pi \cdot \varepsilon_o \cdot \alpha \cdot k} \qquad (2)$$

This is illustrated in the example above, which requests the program to calculate the charges considering the atomic polarizability and the provided force constant.

The virtual site entries are only used to create input files for GROMACS and CHARMM. Each virtual site is set with the tag **vs2**, **vs3**, **vs4**, or **vsn** for virtual sites of type 2, 3, 4, or n, respectively, followed by the number of entries. Each line must follow the formats described in GROMACS and CHARMM manuals. In this case, the program will append the previous lines to the corresponding topology file.

It is possible to define entries that should only be read when the input files aim at a specific MD program. For example, the use of rigid bodies is implemented only for DL_POLY. Thus, a rigid unit entry can be set in the GEN file in an IF section, that is ignored when GROMACS, LAMMPS, and CHARMM input files are created:

```
IFDLP
  RIGID 1
  5 1 2 3 4 5
ENDIF
```

In the example above, "IFDLP" states that all the following entries are only used for DLPOLY input files. Each section should end with "ENDIF". Capital letters should be used to define these entries. Analogously, for GROMACS, LAMMPS, and CHARMM options, use "IFGMX", "IFLMPS" and "IFCHRM", e.g.:

```
IFGMX
      # EXTRA IMPROPER (at1, at2, at3, at4)
      IMPROP 2
      2    1     5     4
      2    1     3     4
ENDIF
```

Similarly, it is also possible to use the IF section for DRUDE particles and symmetry operations. For the first case use "IFDRUDE" while for the second "IFCRYST":

```
IFDRUD
   # Drudes (position, polarizability, k, Charge)
   DRUDES 6 DRUDMASS 0.3
   1     0.987     2092.00   0.00000
   2     1.482     2092.00   0.00000
   3     0.987     2092.00   0.00000
   4     1.482     2092.00   0.00000
   5     1.482     2092.00   0.00000
   6     1.854     2092.00   0.00000
ENDIF
```

and

```
IFCRYST
  # SYMMETRY OPERATIONS
  SYMM        2
  -1.0  0.0  0.0 0.00000   0.0 -1.0  0.0 0.00000   0.0  0.0 -1.0 0.00000
   1.0  0.0  0.0 0.00000   0.0 -1.0  0.0 -.50000   0.0  0.0  1.0 -.50000
ENDIF
```

To use the information inside the latter "IF" sections, the arguments "-drude" and "-cryst" should be included when the program is initiated (see section 2). Please note that any information outside an "IF" section will be considered by the program.

Finally, in the case of the TIP4P water models,[3,4] for LAMMPS and CHARMM, it is necessary to include:

```
TIP4P     0.1577   -1.1794
```

In the latter example, it is stated that the molecular type corresponds to a TIP4P model, the distance between the oxygen atom and the charge position is d(O-M) = 0.1577 Å, and the corresponding charge is -1.1794 e (these values correspond to the model TIP4P/ICE). Additionally, the atoms list must follow the order O, H, H, M.

** PART III **

As in the case of part II, it can be omitted. It contains the coordinates of the atoms in the form of a PDB file for use with e.g., RasMol, VMD, or MERCURY. Note that this file can be directly opened with the previous programs since they will ignore all information in the initial two sections (except if IF sections are present in the file). This section and part I are automatically generated by DLPGEN when it reads and converts the initial input files.

- Example of Part III -

```
ATOM      1  O1                  3.364   1.188   4.844
ATOM      2  O2                  5.033   2.156  -1.165
ATOM      3  N3                  2.167   3.067   4.515
```

As shown in the example above, the atom names are modified, in this case, to contain the atomic symbol of the atoms and their relative position in Part I. This allows the easy identification of each atom in Part I when they need to be named according to the information in the library file.

4.3 *FORCE FIELD LIBRARY FILE*

The force field file contains all the information about the parametrization required to create the topology files for the different MD programs (e.g., charges, LJ, bonds, angles, and dihedral potentials). DLPGEN closely follows the OPLS force fields. However, other models, including coarse-grained, can also be used.

Each section in the file must start with the keyword ATOMS, BOND, ANGLE, DIHEDRAL, and IMPROPER, in this order, and finish with 'END'. There is no limit to the number of entries in each section, but their presence is necessary even if no data is included. At the end of the file, two optional sections (MIX_VDW and VDW_nm) can also be included, allowing to change the mixing rules applied to the van der Waals potentials.

- Example of LIBRARY file -

```
UNITS kJ
LJfudge   0.5
Qfudge    0.5

ATOMS
Name  Nam_Cod   Mass        Charge      Epsilon       Sig (A)
CBT   CBT       12.011      0.3500      0.27614       3.500
FBT   FBT       18.998      -0.160      0.22200       2.950
SBT   SBT       32.066      1.0200      1.04600       3.550
OBT   OBT       15.999      -0.530      0.87900       2.960
NBT   NBT       14.007      -0.660      0.71100       3.250
END ATOMS

BOND            k          b0
CBT     FBT     3698.00    1.323
CBT     SBT     1950.00    1.818
OBT     SBT     5331.00    1.437
NBT     SBT     3137.00    1.570
END BOND

ANGLE                   K          theta0
FBT     CBT     FBT     781.00     107.1
FBT     CBT     SBT     694.00     111.7
CBT     SBT     OBT     870.00     102.6
OBT     SBT     OBT     969.00     118.5
NBT     SBT     OBT     789.00     113.6
END

DIHEDRAL                        V1/A     V2/f      V3/m     V4/m    Code
FBT     CBT     SBT     OBT     0.00000  0.00000   1.4510   0.000    1
OBT     SBT     NBT     SBT     0.00000  0.00000   -0.015   0.000    1
FBT     CBT     SBT     NBT     0.00000  0.00000   1.3220   0.000    1
CBT     SBT     NBT     SBT     32.7730  -10.420   -3.195   0.000    1
*       CA      CA      *       0.00000  6.50000   0.0000   0.000    1
END
```

```
IMPROPER                          V1       V2        V3
NA      CR      NA      HA        0.00000  9.20000   0.0000
*       CR      *       *         0.00000  9.20000   0.0000
END

MIX_VDW                 Epsilon       Sig (A)
ClCW       Cl-          1.01786       2.92035
ClCW       I-           1.12497       3.16207
END

VDW_nm                Eo       ro    n     m
CNT       CT          0.4490   3.85  11    10
CNT       C1          0.8340   3.61  10    9
END
```

The force field file can start with:

```
UNITS [KEY]
```

where "KEY" can be "kJ", "kcal", and "eV". If omitted, "kJ" (per mole) will be considered. In the case of the **distances**, all values should be given in **Angstroms**. Independently of this, the following units for energy and distance will be set for the different MD programs:

DL_POLY	$kJ \cdot mol^{-1}$, Å
GROMACS	$kJ \cdot mol^{-1}$, nm
LAMMPS	$kcal \cdot mol^{-1}$, Å
CHARM	$kcal \cdot mol^{-1}$, Å.

The next lines should be:

```
LJfudge   0.5
Qfudge    0.5
```

which gives the 1-4 van der Waals ("LJfudge") and Coulomb ("Qfudge") interactions scaling factor. If this information is omitted, a value of 0.5 will be assumed during the preparation of the input files.

In the ATOMS section, the first line after the entry is not read by the program. Each line contains six values (see example above): two names that identify (ID) the atoms, the atomic mass (in au.), charge (in e), and the 12-6 Lennard-Jones (LJ) parameters:

$$U = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] \quad (3)$$

where ε and σ are the LJ parameters. In the previous equation r is the distance.

The first atom ID in each entry ("Name" in the example above) gives the atom name in the *.gen file (column B, see section 4.2) and the second ("Nam_Cod"), gives the codename used to find the bonds, angles, and dihedrals parameterization in the following sections. This allows, for example, to define several types of methyl carbon atoms (e.g., CS or CT) with e.g., different charges or Lennard-Jones parameters, and use only

one entry for the bonds, angles, and dihedrals potentials involving these atoms.

The BOND and ANGLE sections contain the parametrization assuming harmonic oscillators:

$$U(r) = k/2 * (r - r_0)^2 \quad (4)$$

and

$$U(\theta) = k/2 (\theta - \theta_0)^2 \quad (5)$$

respectively. Each line starts with two and three "Nam_Cod" labels, that form a given bond and angle in the molecule, followed by the force constant and equilibrium distance, r_0, or angle, θ_0.

In the case of the dihedrals, two potential functions can be used:

$$U(\phi) = V_1/2(1+\cos(\phi)) + V_2/2(1-\cos(2\phi)) + V_3/2(1+\cos(3\phi))+ V_4/2(1-\cos(4\phi)) \quad (6)$$

and

$$U(\phi) = A(1+\cos(m*\phi-f)) \quad (7)$$

where ϕ is the torsional angle, V_1, V_2, V_3, and V_4 are the coefficients of the Fourier series, A is the potential energy, m is the cosine frequency, and f is the cosine phase. The potential parameters in each line of this section should be preceded by four "Nam_Cod" names that define the dihedral. In the last column of each line, an integer number identifies the potential function to be considered: "1" for equation (6) and "2" for equation (7). The wildcard "*" can also be used to represent any type of atom. Please note that, as in the OPLS-AA force field, by default, DLPGEN will set a scale factor of 0.5 for the 1-4 electrostatic and van der Waals interactions. Finally, for the improper dihedrals, equation (6) should be used with $V_1 = V_3 = 0$, and $V_2 > 0$.

The two final sections can be used to change the LJ mixing rules for specific atom pairs. These entries are optional and can be included in the file in any order after the IMPROPER section. "MIX_VDW" allows specifying the Lennard-Jones sigma value, σ_{ij}, and well-depths, ε_{ij}, for a given atom pair. For this, after "MIX_VDW" the atom "Name" in the pair interaction should be included, followed by the corresponding σ_{ij} and ε_{ij} values.

It is also possible to use an n-m potential function for a given atom pair:[5]

$$U_{ij} = \frac{E_0}{n-m}\left[m\left(\frac{r_0}{r_{ij}}\right)^n - n\left(\frac{r_0}{r_{ij}}\right)^m\right] \quad (8)$$

In this case, after the section named "VDW_nm", in each line, two entries with the atoms "Name" in the interaction should be followed by E_0, r_0, n, and m. Note, however, that this option is currently only implemented for the MD programs DL_POLY and LAMMPS.

COARSE GRAIN (CG) MODELS

The current version of the program also supports coarse grain models. This option is **only implemented for GROMACS**. The library file has the same structure as the previous one but at the beginning the key "CGFF" should be included and the LJfudge and Qfudge values changed to 1 (as in the case of the Martini force fields). The key "CGFF" leads to the following changes when creating the input files:

- Only bonded bids will be set in the exclusion list.
- The angle function will be changed to:

$$U(\theta) = k/2\ [(\cos(\theta) - \cos(\theta_0)]^2 \quad (9)$$

- Dihedral entries will be ignored.
- Improper functions will be set as

$$U(\theta) = k/2\ (\theta - \theta_0)^2 \quad (10)$$

so that the Improper section in the LIB file must be set as:

```
IMPROPER                        Theta_0       k
ci     cj     ch     cl         90.0          100.0
END
```

where the four initial entries refer to the atom names of the improper dihedral defined using the "Nam_Cod" entries. The following real numbers are the equilibrium angles and constant of the harmonic oscillator.

As described above, DLPGEN searches for bonds formed between the atoms considering the first two letters of the atom names. Because in the CG model groups of atoms are considered instead, each atom's name should start with a lowercase letter. This avoids any possible mistakes by the program, by assuming atoms instead of beads. Connectivity can then be set for automatic search by providing the appropriate cut-off distances when requested or defining bonds/constraints in the GEN file (recommended). In the latter case, if the option "noappend" is used, DLPGEN will compute the angles of each molecule. The definition of improper dihedrals must be performed manually in the GEN file (see details in section 4.2).

Setting CG models also produces other important changes in the NPT.mdp file generated for GROMACS. The main changes are: (i) the timestep set to 10 fs, which can be increased to 20 ps or 25 ps after the first run; (ii) to account for the electrostatic interactions beyond the cut-off, the reaction field method is set; (iii) compressibility assumed in the barostat is increased by one order of magnitude.

4.4 *Z-MATRIX FILE*

To use a Z-matrix as starting point, a file named "molecule.zmat" should be placed in folder **DIR**, set in the PCF file. Details about the structure of a Z-matrix can be found elsewhere. This file should start with the number of atoms in the molecule followed by the connectivity. The first column in each line contains the atom number (for guidance) and can be omitted. An example is given below:

```
5
1    CT
2    CS    1       1.529
3    CS    2       1.529          1       109.5
4    HC    1       1.09           2       109.5          3       60.0
5    HC    1       1.09           2       109.5          3       -60.0


# or

5
CT
CS    1       1.529
CS    2       1.529          1       109.5
HC    1       1.09           2       109.5          3       60.0
HC    1       1.09           2       109.5          3       -60.0
```

An important feature of preparing input files from a Z-matrix is the possibility to control the molecular orientation. This is useful if a molecule needs, for example, to be orientated in space inside a crystal unit cell. For this, the x, y, and z coordinates of the first three atoms of the molecule in the Z-matrix should be given in a separate file (e.g., ref_file.xyz) as follows:

```
0.533003719003E+01  0.366096273430E+01  0.267653960000E+01
0.353881787658E+01  0.353512271833E+01  0.660524440000E+01
0.129629437485E+01  0.716038615732E+01  0.475910340000E+01
```

Finally, the program should be run as:

```
$ dlpgen -xyz_zmat ref_file.xyz
```

4.5 *Z-MATRIX FILE FOR POLYMERIC CHAINS*

DLPGEN can generate a polymeric chain based on information provided for monomers. This requires a Z-matrix containing the information about (at least) the 3 first monomers in the chain. The program then expands the Z-matrix to create a polymer with the required size and produces the corresponding GEN file. It is important that in the initial Z-matrix atom names already correspond to those in the library file. This process is not foolproof and requires the following steps:

1) Create a Z-matrix for the three initial monomers in the chain. It is important that: (i) the first monomer is built with the atoms numbered from 1 to N. (ii) The next monomers build sequentially one at a time. (iii) The last atom of the Z-matrix should be labeled X. Please see the example in **Figure 1** for polystyrene. As a reference, for the last C-X bond use a distance of 1.2 Å.

2) Using the Z-matrix obtained in 1) build a file with termination *poly, e.g., "FILE_POLY.poly", with the following structure:

```
no_monomer    20              # Number of monomers.

monomer   21  53              # Position of the atoms that limit the
                              # monomers to be repeated

terminal  2                   # Atom names in the last monomer of the
   50   CT                    # chain.
   53   HC
end

zmat     53                        # Initial Z-matrix
CT
HC   1      1.07000000
HC   1      1.07000000    2     109.47120255
HC   1      1.07000000    3     109.47125080    2      -119.99998525
CTtA 1      1.54000000    4     109.47121829    2      -120.00000060
HC   5      1.07000000    1     109.47120255    4        59.99999036
…
HC  50      1.07000000   37     109.47120255   34       -60.00000740
HC  50      1.07000000   37     109.47120255   34        60.00000740
X   50      1.20000000   37     109.47123134   34       180.00000000
end
```

The *.POLY file must have four distinct entries, which start with the following keys in bold:

no_monomer *f*	Number, ***f*** (integer), of monomers in the chain. Note that DLPGEN will consider that the initial Z-matrix already contains 3 monomers and depending on ***f*** being even or odd, the final polymeric chain size may differ by one unit.
monomer f1 f2	The number of atoms that limit the monomers to be repeated in the Z-matrix expansion. **f1** and **f2**

		(integers) are the atomic positions of the 1st and last atoms of the repeating unit.
terminal	***f***	In the last monomer, the atom names should be changed to properly close the chain. Thus, different atom names can be defined. In this section, ***f*** is the number of atoms whose names should be changed. This entry is followed by ***f*** lines containing the atom position and the name to be used in the last monomer.
zmat	***f***	Z-matrix containing ***f*** number of atoms.

3) Finally run DLPGEN as:

```
$ dlpgen -poly
```

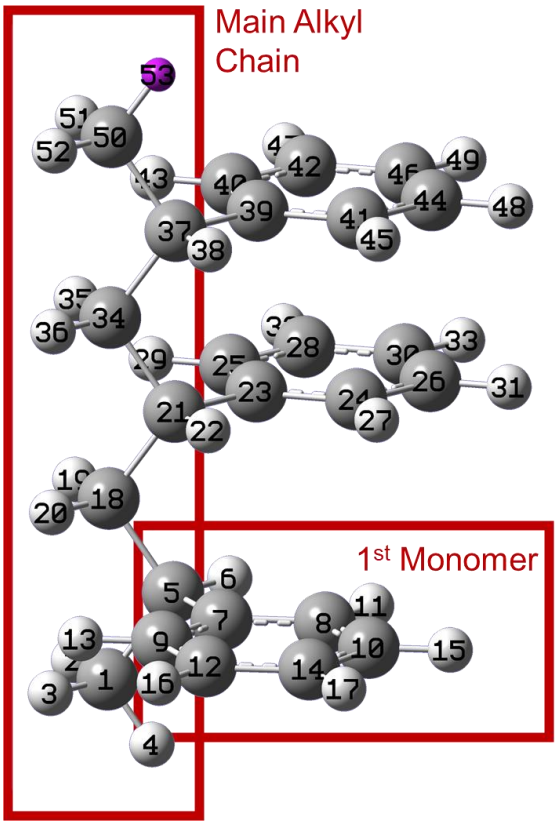
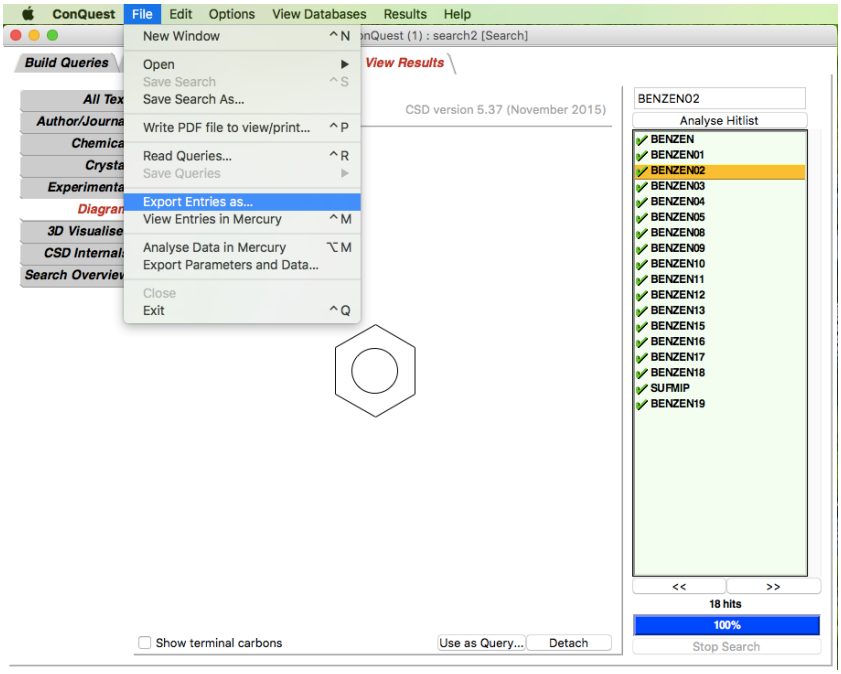


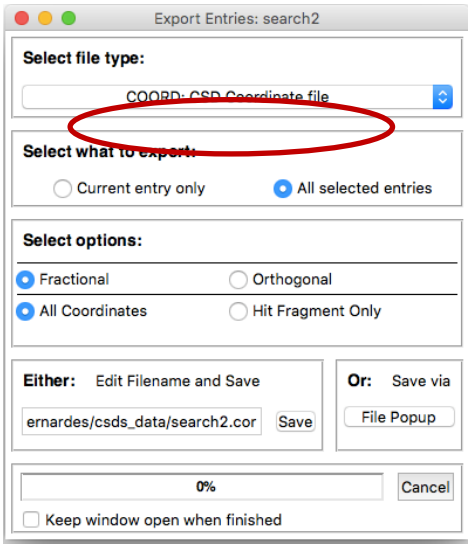
Figure 1. Example of the numbering scheme to construct a polymeric chain for polystyrene.

4.6 *CAMBRIDGE STRUCTURAL DATABASE COORDINATES FILE (COR)*

The COR file should be named with "MOLS_NAME" as defined in the PCF file, followed by **.cor** (e.g., molecule.cor), and placed in folder **DIR**. The file has the same format used with the Cambridge Structural Database (CSD) coordinate file, but without the header and with an 'END' entry after the reduced coordinates (see example below). It can be obtained using the program CONQUEST from CSD. For this, export a selected structure using the "Export Entries as" in the "File" menu:



Select the file type as "COORD: CSD Coordinate File" and choose the file name:



Finally, edit the obtained file as follows:

- Example of a COORD file -

```
CELL       4.583   8.825  15.888  90.000  90.770  90.000
SYMM      1.  0.  0. 0.00000   0.  1.  0. 0.00000   0.  0.  1. 0.00000
SYMM     -1.  0.  0. 0.00000   0.  1.  0. 0.50000   0.  0. -1. 0.50000
SYMM     -1.  0.  0. 0.00000   0. -1.  0. 0.00000   0.  0. -1. 0.00000
SYMM      1.  0.  0. 0.00000   0. -1.  0. -.50000   0.  0.  1. -.50000
O_2          0.74830   0.13460   0.30494     1555001
OHP          1.09470   0.24430  -0.07333     1555002
N_2          0.48600   0.34750   0.28422     1555003
C_2          0.66860   0.24090   0.25753     1555004
HA           1.05400   0.07300   0.18730     1555017
END
```

Note that, in some cases, the symmetry operations in the COR file may be recorded as follows:

```
SYMM     -1. -. -. -.00000  -. -1. -. -.00000  -. -. -1. -.00000
```

DLPGEN is not able to read these entries. The program will run normally without warnings or errors, but the conversion of the *cor file to *gen will be incomplete. To avoid this problem, add zero values after "-.", as shown below:

```
SYMM     -1. -.0 -.0 -.00000  -.0 -1. -.0 -.00000  -.0 -.0 -1. -.00000
```

It is also possible to create a *cor file manually if, for example, only a crystallographic input file (CIF) is available. Consider, for example, the CIF file for Anhydrous sodium tetrachloroaurate(III) with reference "2018864" in the Crystallography Open Database. The unit cell dimensions are given in the entries:

```
_cell_angle_alpha                90
_cell_angle_beta                 104.58
_cell_angle_gamma                90
_cell_length_a                   11.277
_cell_length_b                   11.234
_cell_length_c                   20.584
```

and convert to:

```
CELL       11.277  11.234  20.584  90.000  104.58  90.000
```

In turn, the symmetry operations need to be converted into a matrixial form. This information normally appears in the CIF file as follows:

```
_symmetry_equiv_pos_as_xyz
x,y,z
-x+1/2,y+1/2,-z+1/2
-x,-y,-z
x-1/2,-y-1/2,z-1/2
…
```

which converts to:

```
SYMM      1.  0.  0. 0.00000   0.  1.  0. 0.00000   0.  0.  1. 0.00000
SYMM     -1.  0.  0. 0.50000   0.  1.  0. 0.50000   0.  0. -1. 0.50000
SYMM     -1.  0.  0. 0.00000   0. -1.  0. 0.00000   0.  0. -1. 0.00000
SYMM      1.  0.  0. -.50000   0. -1.  0. -.50000   0.  0.  1. -.50000
…
```

Finally, the reduced coordinates can be directly copied to the *cor file. Consider the example:

```
_atom_site_label
_atom_site_type_symbol
_atom_site_fract_x
_atom_site_fract_y
_atom_site_fract_z
_atom_site_occupancy
_atom_site_U_iso_or_equiv
Cl15 Cl-1 0.6111 0.5006 0.7985 1 0.0
Cl8 Cl-1 0.4853 0.0966 0.3647 1 0.0
Cl5 Cl-1 0.3868 0.3084 0.4407 1 0.0
…
```

Then the corresponding reduced coordinates are:

```
Cl15      0.6111   0.5006   0.7985
Cl8       0.4853   0.0966   0.3647
Cl5       0.3868   0.3084   0.4407
…
```

Note that if the data contains errors, often given in parenthesis, this information must be removed.

5. OUTPUT FILES

Table 1 gives a list of all main files created by DLPGEN. For each program, all files listed should be used in the simulations. The only exception is the file CONFIG.pdb, which aims only at the visualization of the produced simulation box if required. The files CONTROL, NPT.mdp, in.lmp, and input_chrm.inp, contain the simulation settings and closely follow the recommendations for the corresponding program. Thus, the user can (**should**) edit the information contained in these files to properly set the calculations. In the case of in.lmp and input_chrm.inp for LAMMPS and CHARMM, respectively, due to the large complexity of these files, the first lines have a non-standard section that can be easily changed, e.g., to set the number of steps, cutoff, and timestep of the simulations.

Several other auxiliary files are also built during the execution of DLPGEN. These files are stored in the folder **DIR** set at the PCF file. They can be edited and used while running the program, to correct any eventual problem found while running DLPGEN.

5.1 *CONNECTIVITY FILE*

The connectivity file is automatically generated by DLPGEN from the coordinates included in *gen files. It is placed in the FILES LOCATION directory **DIR** preserving the core file name and with the extension *badin, and contains the bonds identified by the program. It is possible to edit this file if, for example, the algorithm that determines the connectivity between atoms misses a bond. In this case, to avoid the recalculation of the bonds, the program should be run as:

```
$ dlpgen –noconnect
```

5.2 *TOPOLOGY FILE*

The topology file is automatically generated by DLPGEN, from *badin files. It is placed in the FILES LOCATION directory **DIR** with the extension *badout. This file contains the bonds, constraints, angles, dihedrals, impropers, and (if applicable) drudes 1-4 exclusions. As in the latter case, it is possible to edit this file to solve any issue found (e.g., the algorithm missed an angle, dihedrals, or improper for a given molecule). This is likely to occur if the molecule contains, for example, nitrogen atoms. In this case, the identification of improper dihedrals is based on the angles formed by the dihedral X-N-X-X. If the angle is close to 0 or 180°, then an improper will be considered by the program. Otherwise, it is ignored. If the verbose mode is used (argument "-v"), a warning message is printed by the program with the result of the analysis.

Table 1. Output files produced using DLPGEN.

PROGRAM	ARGUMENT	FILES
DL_POLY	-dlp	FIELD CONTROL CONFIG CONFIG.pdb
GROMACS	-gmx	topology.top CONFIG.gro NPT.mdp CONFIG.pdb
LAMMPS	-lmps	in.lmp data.lmp pair.lmp CONFIG.pdb topology.inp
CHARMM	-chrm	input_chrm.inp config.crd parameters.prm topology.rtf CONFIG.pdb

6. TIPS

As with any other program, DLPGEN can be used in several different ways depending on the user's needs and starting point. The main idea in the case of DLPGEN is to use *gen files that contain all necessary data to prepare the simulation (see section 4.2). These files can be obtained using the program itself or, e.g., for ionic liquids, alcohols, and small neutral molecules, downloaded from:

http://webpages.ciencias.ulisboa.pt/~cebernardes/dlpgen_prog/Software_dlpgen_ff_files.html

To create your whom *.gen files you can use the following procedure:

1. Convert your initial PDB file (or other supported input file) to *.gen, by using:

```
$ dlpgen -pdb
```

   In this case, a file named "PCF" with the simulation system details (e.g., files to be used) should be present in the pattern folder (see section 4.1). The command above will convert the initial *.pdb file to *.gen and stop.

2. Open the converted *.gen file (placed in **DIR**) and make the necessary changes to match the atom names with those in the library file. To help in the assessment process, open the same *.gen file with, e.g., VMD, RasMol, or MERCURY. This action will load the *.pdb section in the *gen file. At this point, the atoms labels displayed by the visualization software will contain the atoms type (e.g., "C" for carbons and "O" for oxygens) and their position in the *gen file. For example, if the second atom in the list is carbon, the corresponding label is "C2" and, if the third atom is oxygen, is named "O3". Mind that all-atom names in the PDB section of the *gen file are neglected by DLPGEN, thus, no relation between these names and those in the force field or library file exists.

3. Run the program using:

```
$ dlpgen
```

4. It is likely to obtain error messages at this stage. This may happen because some of the required force field parameters (i.e., atom names, bonds, angles, dihedrals, and improper dihedrals) are not present in the library file. The error will include information about what is missing. For example, if the parameters for the bond between carbon (CTO) and oxygen (OH) atoms, at positions 5 and 8 are missing, the following error message is displayed:

```
Parameters missing for Bond CTO OH
between atoms       5       8
Write parameters in the library file for bond CT OH
```

As can be observed in the example, the error message contains the position of the atoms in the *gen file, and the code name "Nam_Cod" are used to ID each atom in the LIB file (recall section 4.3). To avoid guessing all parameters needed to build your force field, use and abuse this ability of the program.

5. Use VMD, RasMol, or MERCURY as described in point 2 to identify the missing parameters.

6. After running the program successfully, check if everything is correct (e.g., the number of bonds, angles, dihedrals, and the identification of improper dihedrals). Use the verbose mode if necessary. Finally, set the parameters to obtain the final input files for your MD program. For example, ask the program to use PACKMOL or request input files for DL_POLY, GROMACS, LAMMPS, or CHARMM.

7. EXAMPLES

As starting point to use the program please look at the tutorials available online:

https://webpages.ciencias.ulisboa.pt/~cebernardes/dlpgen_prog/Software_dlpgen_tutorials.html

Additionally, with the program source files is a folder containing examples of using the program, which includes:

1. Generate input files for ethane using a polarizable force field.
2. Generate input files for butane from a z-matrix.
3. Generate input files for butane from a PDB file.
4. Generate input files for a crystal structure of benzoic acid from a COR file.
5. Generate input files for a 4'-hydroxyacetophenone (HAP) hemihydrate.
6. Generate input files for liquid water using PACKMOL, and assuming a given system density of 0.9 g cm^3.
7. Generate input files for liquid water using PACKMOL, using a previously defined simulation box.
8. Create input files for LAMMPS, GROMACS, or CHARMM.
9. Create a polymeric chain of polystyrene.

To run these examples, check the instructions in the README file included in the examples folder.

8. RELEASE NOTES

Version 3.1.1

This version contains the following changes:

- Ability to create polymeric chains.
- Support for Coarse-Grained models, like the Martini force field (Beta release for GROMACS users).
- Bug fixes.

Version 3.0.05

This version contains the following changes and bug fixes:

- Updated user manual.
- The Drude spring constant, k, is now defined following the conventional harmonic oscillator equation (i.e., k/2).
- Fixed problem when LAMMPS input files are prepared using simulation boxes containing TIP4P water molecules, previously equilibrated with DL_POLY and GROMACS.
- Fixed problem when the same molecule is used more than once in the same topology file of GROMACS.
- Other minor fixes and warnings messages improved.

9. REFERENCES

1. Allen, F. H. The Cambridge Structural Database: a Quarter of a Million Crystal Structures and Rising. *Acta Crystallogr. B* **2002**, *B58*, 380-388.

2. Martínez, L.; Andrade, R.; Birgin, E. G.; Martínez, J. M. PACKMOL: A Package for Building Initial Configurations for Molecular Dynamics Simulations. *J. Comput. Chem.* **2009**, *30*, 2157-2164.

3. Abascal, J. L. F.; Vega, C. A General Purpose Model for the Condensed Phases of Water: TIP4P/2005. *J. Chem. Phys.* **2005**, *123*.

4. Abascal, J. L. F.; Sanz, E.; Fernandez, R. G.; Vega, C. A Potential Model for the Study of Ices and Amorphous Water: TIP4P/Ice. *J. Chem. Phys.* **2005**, *122*.

5. Clarke, J. H. R.; Smith, W.; Woodcock, L. V. Short-Range Effective Potentials for Ionic Fluids. *J. Chem. Phys.* **1986**, *84*, 2290-2294.

For any questions or problems, please contact me.

GOOD SIMULATIONS!